SECONDS FIRST!

A Thesis Dedicated to Secondary Structure Elements

DISSERTATION

zur Erlangung des Grades eines

Doktors der Naturwissenschaften

der Technischen Universität Dortmund
an der Fakultät für Informatik
in Zusammenarbeit mit der Fakultät für Chemie und Chemische Biologie

von

Tobias Brinkjost

Dortmund

2019

technische universität dortmund
fakultät für informatik
fakultät für chemie und chemische biologie

Tobias Brinkjost
Lehrstuhl XI – Algorithm Engineering
Fakultät für Informatik
Technische Universität Dortmund
Otto-Hahn-Str. 14
44227 Dortmund

Tag der mündlichen Prüfung: 18.12.2019

Dekan: Prof. Dr.-Ing. Gernot A. Fink

Gutachter/Gutachterinnen: Prof. Dr. Petra Mutzel, Prof. Dr. Daniel Rauh, Prof. Dr. Sven Rahmann

“And therefore as a stranger give it welcome. There are more things in heaven and earth Horatio, Than are dreamt of in our philosophy.”

Hamlet, Act 1, Scene 5

Acknowledgements

It is a pleasure to thank those who made this thesis possible and blame all disbelievers (q.e.d).

I am deeply grateful to Prof. Dr. Petra Mutzel and Prof. Dr. Daniel Rauh for their help, encouragement, confidence, and support in any situation, their inspiration, and many fruitful discussions. I am especially in debt to Prof. Dr. Petra Mutzel for funding. In this regard, I also would like to thank Dr. Stefan Dissmann for his support, many inspiring on- and off-topic discussions, and the opportunity to fulfill teaching responsibilities for his lectures. I would like to offer my special thanks to Prof. Dr. Stefan M. Kast for his constructive comments on the SCOT publication and his support during the last months. I want to thank Dr. Oliver Koch for the opportunity to work on this project. I am grateful to Prof. Dr. Günter Rudolph for acting as my mentor and to Dr. Markus Schürmann for his precious advice. I wish to thank Gundel Jankord for her constant commitment and optimism.

It was a pleasure to work on this project in close collaboration with Christiane Ehrt and so it is a pleasure to express my gratitude to her, for all the good and bad times we have been through, for her joy and patience, day by day and desk next to desk, for her thoroughly proof reading of this thesis, and for having shared at least the same fascination for the project as I have done.

I owe a very important debt to my colleagues Sarah Albers, Christiane Ehrt, Lina Humbeck, Julia Jasper, Dr. Dennis Krüger, Dr. Mauro Nogueira, Dr. Hitesh Patel, Dr. Jette Pretzel, and Anna Rudo for the collaborative, inspiring, and family-like atmosphere in any situation. For sure, science can be a tough row to hoe and sometimes we had to go to hell and back to accomplish our goals. But it always felt like an adventurous joyride with colleagues like you. Let's go bowling!

I would like to thank my family for their support and their faith in me through my entire life and in particular, I must acknowledge Heidrun Haselau, without whose love, patience, and encouragement I would not be where I am today.

Last but not least, to all the people who are not named individually but also contributed each in his or her personal way, from secretaries, facility managers, people who wore a smile, train drivers that arrived on schedule, train drivers that caused creative breaks by not arriving on schedule, . . . , to friends who were patient with me while I was busy doing or talking about all that weird science stuff: thank you, the drinks are on me!

Collaboration

This thesis was created under the supervision of Prof. Dr. Petra Mutzel and Prof. Dr. Daniel Rauh at the Department of Computer Science and the Faculty of Chemistry and Chemical Biology of the TU Dortmund University between February of 2013 to February of 2019.

Parts of this thesis were created in close collaboration with Christiane Ehrt. All contributions that are only briefly introduced herein are referenced at the corresponding positions, i.e., the ESOM training and turn dihedral angles clustering in Section 3.3.2, the selection of a hydrogen bond criterion and its parameter optimization in Section 3.5.2.1, and the analysis of the automated pocket detection approaches in Section 5.6.8.

Apart from that, the creation of the datasets described in Section 2.3 and the evaluations presented in Sections 4.6, 5.6.4, and 5.6.5 are based on her work.

List of Publications

- C. Ehrt, T. Brinkjost, and O. Koch. “Impact of Binding Site Comparisons on Medicinal Chemistry and Rational Molecular Design”. In: *Journal of Medicinal Chemistry* 59.9 (2016), pp. 4121–4151. DOI: 10.1021/acs.jmedchem.6b00078
- H. Patel, T. Brinkjost, and O. Koch. “PyGOLD: a python based API for docking based virtual screening workflow generation”. In: *Bioinformatics* 33.16 (2017), pp. 2589–2590. DOI: 10.1093/bioinformatics/btx197
- J. Jasper, L. Humbeck, T. Brinkjost, and O. Koch. “A novel interaction fingerprint derived from per atom score contributions: exhaustive evaluation of interaction fingerprint performance in docking based virtual screening”. In: *Journal of Cheminformatics* 10.1 (2018), p. 15. DOI: 10.1186/s13321-018-0264-0
- C. Ehrt, T. Brinkjost, and O. Koch. “A benchmark driven guide to binding site comparison: An exhaustive evaluation using tailor-made data sets (ProSPECCTs)”. In: *PLOS Computational Biology* 14.11 (2018), pp. 1–50. DOI: 10.1371/journal.pcbi.1006483
- C. Ehrt, T. Brinkjost, and O. Koch. “Binding Site Comparison – Software and Applications”. In: *Encyclopedia of Bioinformatics and Computational Biology*. Ed. by S. Ranganathan, M. Gribskov, K. Nakai, and C. Schönbach. Oxford: Academic Press, 2019, pp. 650–660. ISBN: 978-0-12-811432-2. DOI: 10.1016/B978-0-12-809633-8.20196-9
- C. Ehrt, T. Brinkjost, and O. Koch. “Binding Site Characterization – Similarity, Promiscuity, and Druggability”. In: *Med. Chem. Commun.* (2019). In submission
- T. Brinkjost, C. Ehrt, O. Koch, and P. Mutzel. “SCOT: Rethinking the Classification of Secondary Structure Elements”. In: *Bioinformatics* (2019). DOI: 10.1093/bioinformatics/btz826

Abstract

The visualization of a protein is hardly imaginable without secondary structure elements (SSEs). But SSEs play a by far more important role in the field of chemical biology, apart from the creation of *fancy* protein images. They are essential in structure-based analyses due to their impact on secondary structure prediction and structural protein alignment.

However, their proper classification is a challenging issue. There are more than 30 tools available that underline the subjective character of the classification of SSEs. But only two tools have dominated this field of research for decades. Why is that? What are the advantages of hydrogen bond-based methods, despite the fact that they are often unable to assign left-handed helices, PPI-helices, or bent structures?

We have developed SCOT, a novel multipurpose software that incorporates the benefits of a multitude of approaches for the classification of helices, strands, and turns in proteins. To our knowledge, it is the very first method that not only captures a variety of rare and basic SSEs (right- and left-handed α-, 3_{10}-, 2.2_7-, plus right-handed π-helices, PPII helices, and β-sheets) in protein structures, but also their irregularities in a single step, and provides proper output and visualization options.

SCOT combines the benefits of geometry-based and hydrogen bond-based methods by using hydrogen bond and geometric information to gain insights into the structural space of proteins. Its dual character enables robust classifications of SSEs without major influence on the geometric regularity of the assigned SSEs. In consequence, it is perfectly suited to automatically assign SSEs for subsequent helix- and strand-based protein alignments with methods such as LOCK2. This is especially supported by our elaborate kink detection. All of these benefits are clearly demonstrated by our results. Together with the easy to use visualization of assignments by the means of PyMOL scripts, SCOT enables a comprehensive analysis of regular backbone geometries in protein structures.

The high number of available secondary structure assignment methods (SSAMs) hampers a straight forward selection of the most suitable one for a certain application. In addition, relying on the most frequently cited tool must not necessarily result in an optimal choice. Thus, we have developed SNOT to fill the gap of a tool that provides a multitude of objective and rational criteria for the comparison and evaluation of different SSAMs. It provides exhaustive information on geometrical parameters, residue statistics, the consistency with respect to protein flexibility and

quality, SSE overlaps and sequence coverage, and the consent of two classifications.

We used SNOT to compare SCOT to DSSP, STRIDE, ASSP, SEGNO, DISICL, and SHAFT. The results point toward SCOT's unique features as a solitary multipurpose SSAM with optimal performance for numerous challenges: the support of commonly observed and rare SSEs, the comprehensive assignment of turn types, the elaborate kink detection, the geometric consistency of the SSEs, the robustness with respect to structure quality and protein flexibility, and its superior suitability for SSE-based protein structure alignments. Our analyses of alternative π- and PPII-helix assignments indicate challenges which we try to address with our methodology.

There are also hints toward a correlation between the SSEs of a protein and its function. Koch and Waldmann proposed that similar arrangements of SSEs in the neighborhood of a ligand binding site (*ligand-sensing cores*) can recognize similar scaffolds in disregard of the overall fold.

We have developed SLOT to discover these unrevealed similarities which are solely based on SSEs. Its graph-based methodology is able to mimic the geometry of SSEs by using a flexible multi-point representation instead of a straight vector. These points are used to capture the geometry of a protein, i.e., the arrangement of SSEs, in distance matrices. This unique representation enables the comparison of protein structures regardless of their SSEs' directions or SSE sequence. An optimized algorithm to determine the MCS of two given graphs is used to calculate their structural similarity and to ensure fast runtimes. What sets SLOT further apart from its 40 competitors is that it can be used with any external secondary structure classification.

Our exhaustive evaluation highlights the benefits of SCOT for the use with SLOT and also covers our optimizations of the comparison algorithm. It additionally questions the applicability of the concept of *ligand-sensing cores*.

In the end, SCOT, SNOT, and SLOT were the beacons on our journey to answer the question: Are there similar *ligand-sensing cores* or undiscovered structural similarities solely based on SSEs?

Abbreviations

APT1	acyl protein thioesterase 1
ASSP	Assignment of Secondary Structure in Proteins
AUC	area under the (ROC) curve
BDA	bending angle
CATH	Class Architecture Topology Homology
CATH-ID	CATH database identifier
DSSP	Define Secondary Structure of Proteins
DISICL	DIhedral-based Segment Identification and CLassification
ECOD	Evolutionary Classification Of protein Domains
ESOM	emergent self-organizing maps
FASSE	fraction of aligned secondary structure elements
LSD1	lysine-specific demethylase 1
MAO	human monoamine oxidase
MCES	maximum common edge subgraph
MCIS	maximum common induced subgraph
MCS	maximum common subgraph
NMR	nuclear magnetic resonance
ROC	receiver-operating-characteristic
PDB	RCSB Protein DataBank
PDB-ID	PDB identifier
PPII	polyproline II
PSSC	Protein Structure Similarity Clustering
RAID	redundant array of inexpensive disks
RMSD	root-mean-square-deviation
SCOT	Secondary structure Classification On Turns
SHAFT	Secbase automated Helix Assignment adapted From Turns
SLOT	Secondary structure Layer One Two
SNOT	Secondary structure Numeric Observation Tool
SSAM	secondary structure assignment method
SSCM	secondary structure comparison method
SSD	solid-state drive
SSE	secondary structure element
STRIDE	STRuctural IDEntification

Vtor	virtual torsion angle

Contents

"All men dream: but not equally. Those who dream by night in the dusty recesses of their minds wake in the day to find that it was vanity: but the dreamers of the day are dangerous men, for they may act their dreams with open eyes, to make it possible."

Thomas Edward Lawrence

1 Introduction

Proteins are the fundamental elements of chemical biology and their computational analysis has become an interdisciplinary and lively field of research since huge – and due to high-throughput techniques [8] still growing – amounts of (protein) data are available through protein databases [9]. From the chemical point of view, proteins are polymers composed of 20 different amino acids joined by peptide bonds and organized in four levels, i.e., the primary, the secondary, the tertiary, and the quaternary level. The three-dimensional structure determines the functions of the protein which are, among others, binding, catalysis, switching, and acting as structural elements. The binding of other proteins, ligands, and antibodies is the main mechanism behind biological pathways and, thus, it is of particular interest in drug design.

From the computational point of view, various different data structures are used to represent the structure of a protein to make them assessable and comparable with respect to the question of structural similarity.

For either perspective, there would be a certain loss in the fascination for proteins, if scientists were not able to visualize their structures to enhance their ideas of the protein of interest, to find structural similarities, to identify binding sites, to follow the binding of a ligand over time in MD simulations, . . . , or just to be attracted by their aesthetics. The majority of these visualizations, three-dimensional visualizations in particular, represent proteins by their secondary structure elements (SSEs), i.e., α-helices, β-strands (and turns connecting these). Thus, our idea of proteins is – whether consciously or unconsciously – highly biased by SSEs.

In addition to the visualization of proteins, there are other applications in which SSEs play an important role, such as protein structure or fold prediction [10], protein structure comparison and alignment [11], and secondary structure prediction [12]. Furthermore, correlations between SSEs and a protein's function have also been observed [13, 14]. However, these examples focus on very specific proteins and functions.

In 2005, Koch and Waldmann proposed a more generally applicable correlation [15]. Their concept of *ligand-sensing cores* postulates that similar spatial arrangements of SSEs constituting the binding site can recognize similar molecular scaffolds in disregard of the overall fold. Due to its general definition and applicability, it is of special interest for the discovery of novel small molecule modulators of protein function for interesting new and unexplored targets with a special focus on chemical biology. To date, there is no automated approach available to address this particular challenge.

All of these aspects underline the relevance and influence of SSEs as a global (protein) player. Unfortunately, there cannot be a correct classification of SSEs [16] in general. However, the provision of an SSE annotation to each protein in the world's largest publicly available resource of protein structures, i.e., the RCSB Protein DataBank (PDB) [9], by default and their automated assignment by many applications, e.g., UCSF Chimera [17], disguise this fact and impede a discussion on their assignment and relevance. It also fosters the idea of a consent although a multitude of different secondary structure assignment methods (SSAMs) and interpretations of a reasonable SSE classification exists.

We created three tools to face the challenge of finding similar SSE arrangements in general and *ligand-sensing cores* in particular in its entirety:

- SCOT – Secondary structure Classification based On Turns (see Chapter 3)
- SNOT – Secondary structure Numeric Observation Tool (see Chapter 4)
- SLOT – Secondary structure Layer One Two (see Chapter 5)

SCOT

SCOT is an SSAM using hydrogen bonds, geometric properties (Cα–Cα distances), as well as dihedral angles (based on turn clustering), and is inspired by the SHAFT classifier [18]. It supports the classification of helices, β-strands, and turns. It reads and writes files in the well-established PDB file format [19]. SCOT utilizes a hierarchical assignment of protein structural elements starting with the assignment of turns. Since most of the publicly available protein structures in the PDB do not contain information on hydrogen atoms, we use the algorithm by McDonald and Thornton [20] to assign them artificially. For the determination of the hydrogen-bonded *normal* and *reverse* turns, we utilize the DREIDING [21] instead of the established DSSP (Define Secondary Structure of Proteins) [22] hydrogen bonding criterion. For the *open* turns, we determine the Cα–Cα distance between the first and the last residue of the turn. The detected turns are then clustered according

to their dihedral angles. We use a dataset of more than 3,500 protein structures from the PDB with distinct sequences by the use of the PISCES sequence culling server [23]. The dihedral angles of the classified turns of these proteins are then clustered by emergent self organizing maps (ESOMs) [24], with up to more than 1,000,000 neurons for a single class, resulting in a variety of distinct turn clusters of similar backbone conformations.

The next layer of the hierarchical assignment of SSEs is dedicated to the classification of sheets and strands. We have developed three algorithms to assign sheets and strands. The final one determines the hydrogen bond contacts for all residues of an input protein structure. Using these contacts, we build a strand graph consisting of sequence regions of consecutive parallel or anti-parallel hydrogen bonding patterns. The edges are labeled with the hydrogen bonds they represent connecting different strands. Thus, each strand, and its length in particular, is implicitly defined by the hydrogen bond information stored at the labels of its vertex' incident edges. We then determine a merge blocking fingerprint based on specific turns which are usually located between succeeding strands within the same sheet. Using this fingerprint, we merge consecutive strands whose gap is not indicated as blocked by this fingerprint. Each connected component of the graph represents a sheet, each of which consisting of at least two strands. To cope with the circularity of β-barrels and to guarantee a deterministic assignment of sheets and strands, we use a priority queue to extract the sheet and strand information out of the graph. During this step, we also determine kinks based on the Cα–Cα distances in segments of length 4 in a strand. If this distance falls below a pre-defined threshold, a kink is defined. The additional information about kinks is added to the REMARK section of the output PDB file to be conform to the PDB file format.

The final layer of the hierarchical assignment of SSEs deals with the classification of helices. We have developed five different algorithms for this purpose. The final one classifies right-handed (α, 3_{10}, π), left-handed (α, 3_{10}), and ribbon (polyproline II, 2.2_7) helices. Each of these three groups is processed separately. In each such group and for each class of a helix (e.g., α), the turn overlaps of all sequence positions of the corresponding turn (i.e., *normal* of length 5 and class 1) are determined. Plus, we also determine the turn overlaps of the corresponding open turns for all helix classes within one group. These are used for the extension of our helices. Based on the class-specific and extension overlaps, we define three layered helices consisting of a core, a hull, and an extension. Each such helix is created whenever we detect a segment of succeeding helix-specific turn overlaps of a minimum number of overlaps and segment length. This is the core of the helix. The hull is defined as all neighboring residues with an overlap of at least 1. The extension is defined according to the core but based on the extension overlaps. We then split and block these helices whenever the Cα–Cα distance of a sequence segment of length 4 within a helix exceeds a predefined threshold. After that, we merge consecutive and overlapping helices and determine their classification based on the sequence coverage and turn overlaps for each involved helix class. The dominant class is taken as the final helix class. We also determine a helix class Purity based on these overlaps to reflect the dominance of a helix' class. We finally assign kinks within cores and hulls based on minima in the corresponding turn overlaps. We also assign classes to kinks to reflect the different geometrical regions a helix can consist of (e.g., 15 for a kink between an α (1) and 3_{10} (5) core). The information about kinks and class Purity are added to the REMARK section.

SNOT

SNOT is a tool to evaluate the performance of SSAMs. We parse files in the PDB file format and extract a multitude of parameters for each type and class of SSEs, a group of SSEs (e.g., right-handed helices), and for the protein in total. For instance, we calculate the Twist, the Rise, and the Radius within all sequence segments of length 4 in a helix. We also provide the φ, ψ, and ω dihedral angles for all residues. The consistency of an SSE classification is reflected by the Tanimoto and weighted Tanimoto similarities based on binary fingerprints for a structure ensemble of one protein. Such fingerprints are also used to compare two SSE assignments to obtain the consensus of different classifiers. Finally, we provide information about the distribution of residues within different SSEs. We calculate the relative frequency and the conformational parameter according to Chou and Fasman [25], and perform the d significance test according to Wilmot and Thornton [26].

We used SNOT to compare different SSAMs, such as SCOT, DSSP, and SHAFT, to pick a suitable classifier that fulfills our requirements to search for common *ligand-sensing cores* with SLOT.

SLOT

The comparison of protein binding sites using the SSE information alone – the so called search for common *ligand-sensing cores* – is the aim of SLOT. In contrast to existing protein structure comparison methods based on secondary structure information (SSCMs), such as TM-align [11] or SSM [27], we perform a two step comparison. At first, the overall protein structure is compared. If the obtained score is below a predefined threshold – the overall folds are not significantly similar –, we compare the binding sites of the proteins in a second step. We have developed and evaluated six different ways of modeling the SSEs of an entire protein structure or a protein's binding site. Our different modeling algorithms mainly utilize undirected labeled graphs, representing the geometry and orientation of helices and strands directly or indirectly, but also on histograms representing the distribution of distinct turn types. Each way of modeling can be used for both steps: to represent the SSEs of the entire protein or of a protein's binding site. We use a modified version of the maximum clique detection algorithm by Tomita et al. [28], an optimization of the algorithm by Bron and Kerbosch [29], to determine the maximum common subgraph (MCS) based on a modular product graph, a technique described by Levi [30]. The determination of the MCS is known to be an NP-hard optimization problem [31]. To address this challenge, our different ways of modeling try to minimize the size of the input graph or use a variety of information to discriminate the compatibility of labels, both to reduce the size or the density of the modular product graph. Finally, we derive the similarity of two given input graphs mainly from the size of their corresponding MCS. For each graph, we provide output files for the visualization by external programs, such as PyMOL [32]. Furthermore, the matching obtained from the MCS can also be exported and visualized. We evaluate each way of modeling by placing five proteins, known to share a common *ligand-sensing core*, in a sequentially non-redundant set of protein structures. This set of non-redundant structures was also used for the classification of turns during the development of SCOT. The five proteins with common *ligand-sensing cores* were MAO-A, MAO-B, and LSD1 [33], and ATP1 and the dog

gastric lipasein [34]. We discuss the applicability of the concept of *ligand-sensing cores* on the basis of our results obtained for this example of a common *binding site fold*.

With these tools at hand, we set out to find structural similarities in proteins solely based on SSEs.

This thesis is organized as follows: Chapter 2 provides the definitions and notations used throughout this thesis. It also contains the description of the creation of all datasets used herein and information about the environment used for the development of our tools. The following three chapters are dedicated to our developed tools. Chapter 3 introduces our SSE classification tool SCOT. Chapter 4 is dedicated to our SSAM evaluation tool SNOT. Chapter 5 describes our search for similar spatial arrangements of SSEs by the introduction of SLOT. Each of these three chapters contains an individual introduction, state of the art, methodology, results, discussion, and outlook. Finally, Chapter 6 provides a conclusion.

"Victory awaits him who has everything in order — luck, people call it. Defeat is certain for him who has neglected to take the necessary precautions in time; this is called bad luck."

Roald Amundsen

2 Preliminaries

2.1 Definitions

The interdisciplinary scope of this thesis requires definitions in the field of chemical biology (see Section 2.1.1) as well as computer science (see Section 2.1.2). The following definitions are used in this thesis. Equations (such as for the comparison of fingerprints) are given in the chapters and sections of their application.

2.1.1 Chemical Biology

Amino acids are the fundamental elements of proteins and form their primary structural layer. Here, we concentrate on the elements of the secondary structural layer (SSEs), i.e., helices, β-sheets, and turns. They are defined on backbone hydrogen bond interactions. Thus, the side-chains that are the individual characteristic of the amino acids, are not import for the definition of SSEs.

Definition 2.1.1 (Amino acid/Residue). Amino acids consist of a Cα atom to which an amino group (-NH_2), a carboxyl group (-COOH), a hydrogen atom, and a side-chain (R) is bound (see Figure 2.1). The side-chain is specific to each amino acid. The main-chain, also referred to as backbone, consist of the N, Cα, C, and O atoms and is identical for all amino acids. The chain-trace backbone atoms correspond to the main-chain backbone atoms without O atoms.

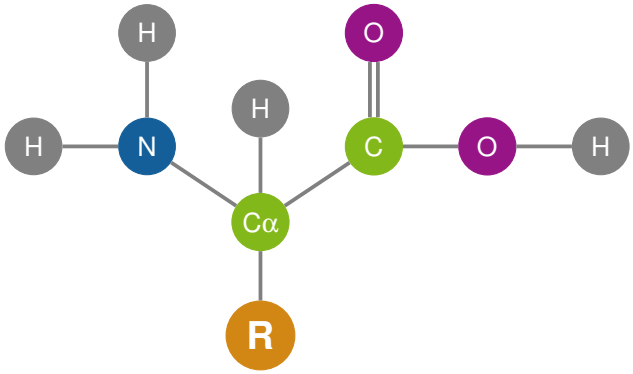


Figure 2.1: Visualization of the common structure/atoms of all amino acids. R indicates the position of the amino acid-specific side chain.

Definition 2.1.2 (Standard amino acids/Residues). Standard protein residues are: Ala, Arg, Asn, Asp, Cys, Gln, Glu, Gly, His, Ile, Leu, Lys, Met, Phe, Pro, Ser, Thr, Trp, Tyr, Val.

Definition 2.1.3 (Protein). Proteins are linear polymers composed of amino acids joined by peptide bonds between their carboxyl groups.

Definition 2.1.4 (Protein structural layers). The hierarchical and structural layers of proteins are:

1. Primary: amino acid sequence joined by peptide bonds
2. Secondary: backbone hydrogen bond interactions forming helices [35], β-sheets [36], turns, and loops
3. Tertiary: folded secondary structure to domains or folds
4. Quaternary: folded tertiary structure to chains

2.1.2 Computer Science

A graph is an abstract mathematical model, which represents objects (vertices) and their relations (edges). In general, a graph is defined as follows:

Definition 2.1.5 (Graph). An undirected graph $G = (V, E)$ is an ordered pair where V is a finite set of vertices and E is a finite set of edges between vertices $E \subseteq \{\{u, v\} | u, v \in V \wedge u \neq v\}$. We say directed graph, if $E \subseteq (V \times V)$ is a finite set of edges where each edge is an unordered pair (u, v) of vertices $u, v \in V$. In a complete graph all vertices are interconnected $E = \{\{u, v\} | u, v \in V \wedge u \neq v\}$ or $E = \{(u, v) | u, v \in V, u \neq v\}$, respectively. A labeled graph $G = (V, E, l_V, l_E)$ assigns additional information to vertices and edges by the use of the labeling functions l_V and l_E.

In this thesis, the labeling functions l_V and l_E of a labeled graph $G = (V, E, l_V, l_E)$ are given by informal descriptions. Therefore, we do not explicitly state them in our notations of labeled graphs

and use the general notation $G = (V, E)$. In addition, we also refer to a directed graph as an undirected graph if the condition $(u, v) \in E \Rightarrow (v, u) \in E$ with $u, v \in V$ holds true.

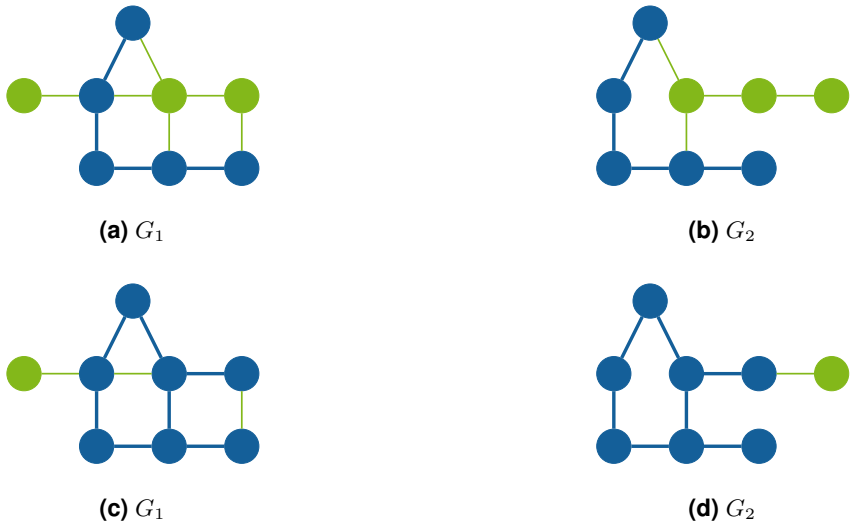
Whenever graphs or other data structures are used to represent elements of the input, the question concerning their similarity arises. Having said that, one way to reflect the similarity of two (or more) graphs is to determine common substructures or subgraphs. The biggest or maximum common subgraph denotes the maximum structural similarity.

Definition 2.1.6 (Subgraph). A graph $G' = (V', E')$ is a subgraph of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. It is said to be (vertex-)induced by V' if $E' = (V' \times V') \cap E$ holds.

For the following definitions, let $G = (V, E)$, $G_1 = (V_1, E_1)$, and $G_2 = (V_2, E_2)$ be graphs.

Definition 2.1.7 (Subgraph isomorphism). G is subgraph-isomorphic to a graph G_1 if there exists an injection $\phi : V \to V_1$ in such a way that $\forall u, v \in V : (u, v) \in E \Rightarrow (\phi(u), \phi(v)) \in E_1$.

Definition 2.1.8 (Common subgraph). G is said to be a common subgraph of the graphs G_1 and G_2 if G is subgraph-isomorphic to G_1 and G_2.



(a) G_1 **(b)** G_2 **(c)** G_1 **(d)** G_2

Figure 2.2: Example for the maximum common (vertex-) induced and edge subgraph. Example of the maximum common (vertex-)induced subgraph (MCIS) (a, b) and maximum common edge subgraph (MCES) (c, d) for two given graphs G_1 and G_2. The vertices and edges of the maximum subgraphs are highlighted in blue whereas all other vertices and edges are highlighted in green. Figures according to Raymond and Willett [37].

Definition 2.1.9 (Maximum common subgraph). The maximum common subgraph (MCS) G_{mcs} of graphs G_1 and G_2 is a common subgraph of G_1 and G_2 for which $|G'| \leq |G_{mcs}|$ holds for all common subgraphs G' of G_1 and G_2. Note that the MCS is not necessarily unique.

An MCS can either be a maximum common (vertex-)induced subgraph (MCIS) or a maximum common edge subgraph (MCES). In an MCIS the number of vertices is maximized whereas in an

MCES the number of edges is maximized. Figure 2.2 gives an example for both variants. Throughout this thesis, we solely use MCISs. Thus, both terms (MCS, MCIS) are used synonymously and refer to MCIS.

The determination of the MCS of two graphs is NP-hard in general [31] and can be reduced to the problem of finding the largest clique (see Definition 2.1.10) in an appropriately defined modular product graph (see Definition 2.1.11), also known as compatibility, correspondence, or association graph. This procedure was first described by Levi [30].

Definition 2.1.10 (Clique). A clique C in an undirected graph $G = (V, E)$ is a subset of vertices $C \subseteq V$ such that every two distinct vertices are adjacent.

Definition 2.1.11 (Modular product graph). In a modular product graph $G_P = (V_P, E_P)$ of two labeled graphs $G_1 = (V_1, E_1, l_{V_1}, l_{E_1})$ and $G_2 = (V_2, E_2, l_{V_2}, l_{E_2})$, the set of vertices $V_P \subseteq V_1 \times V_2$ contains a vertex (v_1, v_2) with $v_1 \in V_1$ and $v_2 \in V_2$ if $l_{V_1}(v_1)$ is compatible to $l_{V_2}(v_2)$. The set of edges E_P contains an edge connecting the vertices $(u_1, u_2), (v_1, v_2) \in V_P$ if either $e_1 = (u_1, v_1) \in E_1$ and $e_2 = (u_2, v_2) \in E_2$ and $l_{E_1}(e_1)$ is compatible to $l_{E_2}(e_2)$, or $e_1 \notin E_1$ and $e_2 \notin E_2$. The latter condition does not come into action if G_1 and G_2 are complete graphs. The compatibility of vertices or edges are strongly application-depended and, therefore, have to be defined individually.

It can be easily demonstrated that a clique in a modular product graph G_P corresponds to the common subgraph of two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$. A clique contains only vertices that represent a pair of compatible vertices $u_1 \in G_1$ and $v_1 \in G_2$ each. All vertices $v \in C$ have to be adjacent to each other, because their underlying compatible vertices $u_1, v_1 \in V_1$ and $u_2, v_2 \in V_2$ are either connected by compatible edges $e_0 = (u_1, v_1)$ E_1 and $e_1 = (u_2, v_2) \in E_2$ or not adjacent. Both criteria result in an edge connecting the corresponding vertices of C.

Definition 2.1.12 (Circle in a graph). A circle in a graph $G = (V, E)$ is a path of edges $e_1, \ldots, e_k \in E$ with $e_1 = (v, w)$, $e_k = (u, v)$, and $v, w, u \in V$ by which v is reachable from itself.

Definition 2.1.13 (Graph density). The density D of a graph $G = (V, E)$ relates its number of edges to the maximum possible number of edges (complete graph of the same size).

$$D(G) := \begin{cases} 0 & |V| < 2 \\ \dfrac{2 \cdot |E|}{|V| * (|V| - 1)} & \text{otherwise} \end{cases} \tag{2.1}$$

Definition 2.1.14 (Tree). An undirected graph $T = (V, E)$ is called a tree if T is connected and does not contain any circles. In a rooted tree, τ is called the root of T. A child $c \in V$ of a vertex $v \in V$ has v as direct predecessor on the distinct path $e_1, \ldots, e_p \in E$ from τ to c: $e_1 = (\tau, w), \ldots, e_k = (v, c)$ with $w \in V$. Any vertex $v \in V$ without children is called a leaf. Herein, we only use rooted trees.

Definition 2.1.15 (Fingerprint). A fingerprint F is a vector of length $|F| \in \mathbb{N}$. A binary fingerprint consists of bits $b_1, \ldots, b_{|F|}$ with $\forall i \in \{1, \ldots, |F|\} : b_i \in \{0, 1\}$. An integer fingerprint consists of integer values $\forall i \in \{1, \ldots, |F|\} : b_i \in \mathbb{N}$. The initial values of a fingerprint are *false* or 0, respectively. We use the terms a bit is marked or unmarked synonymously to $b_i = 1$ or $b_i = 0$, respectively.

Definition 2.1.16 (Queue). A (priority) queue Q is an ordered data structure in which elements E are ordered by keys K. It provides the following operations: $\mathrm{push}(k, e)$ adds element $e \in E$ with key $k \in K$ to Q. $\mathrm{pop}()$ removes and returns the first element with respect to the order of the keys. $\mathrm{contains}(k)$ returns whether an element with key k in Q exists. $\mathrm{remove}(e)$ removes element e from Q.

Definition 2.1.17 (Histogram). A histogram H consists of a set of keys K and a counter function $c : k \to \mathbb{N}$. For each $k \in K$, the counter function $c(k)$ returns the counter value for k. $c(k)$ returns 0 if $k \notin K$. For a number $n \in \mathbb{N}$, $c(k) := n$ sets the counter of $k \in K$ to n.

SSEs are the golden thread through this thesis and, therefore, appear in every main chapter. Graphs are used in Chapter 3 as a pure data structure and in Chapter 5 more extensively as the basis for the determination of structural similarities in proteins. In this context, the MCS (see Definition 2.1.9) is used to represent this structural similarity. Trees (see Definition 2.1.14) are used in Chapter 3 as a data structure reflecting the organization of strands in a β-sheet. Finally, fingerprints (see Definition 2.1.15) are of special importance in Chapters 3 and 4 to model sequence coverages of SSEs, for instance.

2.2 Notations

We use the following notations throughout this thesis.

- Let P be a protein, C be a chain of a protein, and S be an SSE, then $|P|$, $|C|$, or $|S|$ denote the sequence length of the protein, chain, or SSE respectively.
- All residues r_i have (internal) sequence numbers i from $1, \ldots, |C|$ without insertion codes. If we refer to actual sequence numbers in concrete protein structures, we mention this explicitly.
- All hydrogen bonds used herein are intra backbone hydrogen bonds. We use the notation $hb^{+}_{i,j}$ for a hydrogen bond between the hydrogen (H) atom (donor) of residue r_i and the (O) atom (acceptor) of residue r_j. We use the notation $hb^{-}_{j,i}$ for the same hydrogen bond but in acceptor to donor direction. We use $hb^{\pm}_{i,j}$ for any hydrogen bond regardless of its direction.
- A sequence segment s defines a segment of consecutive sequence positions with $s.front, s.back \in \{1, \ldots, |C|\}$ and $s.front \leq s.back$ defining the first and the last sequence position of s.
- A residue's name is referred to by the three letter abbreviations, such as Ala for alanine.
- An SSE type refers to a primary SSE, such as helix or β-sheet (respectively strand).
- The dihedral angles ϕ_i, ψ_i, and ω_i correspond to the dihedral angles of residue r_i at sequence position i (see Figure 2.3). ϕ_i is calculated based on the C atom of r_{i-1} and the N, Cα, and C atoms of residue r_i. ψ_i and ω_i are calculated analogously on their respective four surrounding atoms.

- A PDB file or the PDB file format refers to the standard/fixed column width representation and explicitly not to the also available XML or CIF file formats.
- PDB (and SSAM) file line prefixes, such as `REMARK` or `HELIX`, are written in monospace font.
- @pdb denotes the identifier of one element/protein structure of the RCSB PDB [9]. 1gos@pdb refers to the protein structure with PDB identifier (PDB-ID) 1gos. We use 1gosA@pdb to refer to the chain with chain identifier A of structure 1gos. We use 1j8kA2@pdb to refer to model 2 of chain A of the nuclear magnetic resonance (NMR) structure 1j8k.
- @cath refers to entries of the CATH database [38] analogously.
- All command line options (e.g., `-e`) and flags (e.g., `--write-fingerprints`) are written in monospace font.
- Throughout this thesis, we use the color coding for SSEs depicted in Table 2.1.
- All protein structure illustrations were created using PyMOL [32].

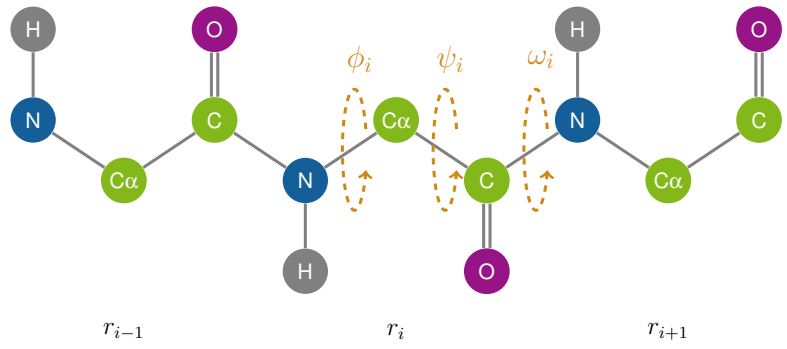


Figure 2.3: Visualization of the dihedral angles ϕ_i, ψ_i, and ω_i of residue r_i.

Helices									Sheets	Kinks
RH α	RH 3_{10}	RH π	RH Mixed	LH α	LH 3_{10}	PPII	2.2_7	Termini		

Table 2.1: Secondary structure color coding. Secondary structure color coding for right- and left-handed (RH, LH), polyproline II (PPII), and 2.2_7-helices, helix termini, strands, and kinks used in protein figures and for the PyMOL [32] export scripts provided by SCOT. This Table is reproduced by permission of Bioinformatics (2019) [7].

2.3 Datasets

The term dataset used herein refers to a subset of protein structures from the publicly available RCSB Protein DataBank (PDB) [9]. A subset can either contain entire protein structures or extracted and separated protein chains or models.

2.3.1 PDB

2.3.1.1 2018

The dataset PDB 2018 copy corresponds to a copy of the entire PDB from the 27th of March 2018. It contains 138,384 protein structures and 385,303 protein chains. The protein files were downloaded utilizing our `PDBFTP` script (see Section 6.4.1, appendix). We use the option `-r` with `ANISOU` and `COMPND` to remove all anisotropic temperature factors and names of standalone drugs or inhibitors. We also set the flag `-lower-pdbid` to change the PDB-ID to its lowercase representation. In NMR structures all models except the first one were removed by evoking the flag `--clean-models`. The splitting of protein files to separate chain files was done with our `PDBChainSplitter` script (see Section 6.4.2, appendix). All ligands were copied to those chains to which any of their atoms is within an at most 3 Å distance.

2.3.1.2 2017

For some of the datasets we used a copy of the PDB from the 20th of March 2017 containing 133,670 protein structures and 369,676 protein chains. These files were downloaded and generated as described for the 2018 copy (see Section 2.3.1.1).

2.3.2 X-Ray Representatives (ESOM Training)

The PDB is a highly redundant dataset. A BLAST [39] sequence similarity clustering revealed approximately 67,000 clusters with 100 % sequence identity. Therefore, we created a dataset of representative structures for the classification of turns and the analysis of SSEs in proteins. All publicly available X-ray structures (2017 copy of the PDB) were split into separate files containing single protein chains. Each chain file contains the global (chain-unspecific) information (e.g., `HEADER` or `REMARK` lines) and the chain-specific information (e.g., `SEQRES` or `HELIX` lines). These chains were filtered based on a resolution threshold of 2 Å and a maximum R-factor of 0.25. Subsequently, the PDB-IDs of the 17,978 filtered structures were submitted to the sequence-culling server PISCES [23]. An identity threshold of 35 % led to a final sequence-diverse dataset of 3,597 X-ray structures of high quality.

2.3.3 Non-Redundant Set of Structures with Left-Handed Helices

SCOT was used to assign the SSEs of all entries in the 2018 copy of the PDB dataset. Structures with at least one left-handed α- or 3_{10}-helix with a length of at least four residues were identified. These structures were sequence-culled using the PISCES [23] methodology with an identity threshold of 35 %. This procedure was restricted to structures with a resolution of 3 Å or better, an R-factor of at least 0.3, and 40 to 10,000 residues per chain.

2.3.4 Quality Dependency Dataset

The dataset of Konagurthu et al. [40], which comprises 15 randomly selected protein structures at different resolutions, was used to analyze the methods' sensitivity to structure quality (see Table 2.2). Instead of using the superseded structures originally taken as low resolution structures, we decided to replace the obsolete and superseded entries by structures that are still available in the PDB [9]. This procedure ensures the use of still relevant structures with a low resolution. We tried to maintain a similarly high resolution difference between low resolution and high resolution structures. Ligands and buffer components were deleted and the structures were aligned and renumbered according to the alignment positions to assess the consensus. Residues that were only found in one structure were removed for comparison purposes. These preparation steps were performed using MOE 2015 [41].

	Low resolution (R)			High resolution (R)		
Structure	**PDB-ID**	**Chains**	**R** / Å	**PDB-ID**	**Chains**	**R** / Å
Hemerythrin-like domain (Myohemerythrin)	2awc (1mhr)	A	2.2 (2.9)	3agt (2mhr)	A	1.4 (1.3)
Dimeric hemoglobin	1nwn (1sdh)	A,B	2.8 (2.4)	3sdh	A,B	1.4
Glutathione reductase	1grh (1grs)	A	3.0 (3.0)	3grs	A	1.5
Adenine glycosylase (Endonuclease III)	1wef (1abk)	A	1.9 (2.0)	1kg2 (2abk)	A	1.2 (1.6)
Lysozyme	1bhz (2lzh)	A	3.9 (6.0)	2zq3	A	1.6
Nitrosomonas cytochrome (Pseudomonas cytochrome)	3zow (151c)	A	2.4 (2.0)	4jcg (351c)	A	1.6 (1.6)
Calmodulin fragment TR2C	1yru (1trc)	B	2.5 (3.6)	1fw4	A	1.7
Ferredoxin reductase	4af6 (1fnr)	A,B	2.9 (2.6)	3mhp (1fnd)	A,B	1.7 (1.7)
Chitinase A (Endochitinase)	3aro (1baa)	A	2.2 (2.9)	3arx (2baa)	A	1.16 (1.8)
Ferrochelatase	1ld3	A	2.6	1doz	A	1.8
Glutamate dehydrogenase	1aup	A	2.5	1bgv	A	1.9
Concanavalin A	4k20 (4cna)	A,B	3.4 (2.9)	5cna	A,B	2.0
Anti-influenza virus antibody 1F1 (Bence-Jones protein)	4gxu (1bjl)	M,N	3.3 (2.9)	4gxv (3bjl)	H,L	1.5 (2.3)
Phosphofructokinase	1mto (5pfk)	A,B,C,D	3.2 (7.0)	4i7e (6pfk)	A,B,C,D	2.0 (2.6)
Serine protease inhibitor	1psi (1qlp)	A	2.9 (2.0)	3ne4 (2psi)	A	1.8 (2.9)

Table 2.2: Structures of the quality dependency dataset. Structures of the quality dependency dataset to analyze the impact of X-ray structure quality on the SSE assignment with different SSAMs. PDB-IDs in brackets denote structures from the original dataset of Konagurthu and co-workers [40] which were replaced for our analyses. This Table is extracted from [7].

2.3.5 Consistency NMR Ensembles

The dataset of NMR structure ensembles was generated to evaluate the consistency of different SSAMs. NMR ensemble structures from the 2017 copy of the PDB dataset were split into separate model files utilizing our `NMRModelSplitter` script (see Section 6.4.3, appendix). Each model file

contains all global information but only the model-specific information on missing residues (REMARK 465 lines) and atom coordinates (ATOM lines). Ensembles with less than 10 models or structures with less than 50 residues were excluded. This filtering step led to 8,800 ensembles adding up to altogether 175,161 structures. Additionally, the dataset was pruned by excluding ensembles that show high structural deviations as expressed by root-mean-square-deviation (RMSD) values. To this end, all structures within one ensemble were aligned with TM-score [42]. The RMSD of the sequence-based structure alignment of the complete structures was used for the subsequent filtering procedure. All ensembles showing an RMSD of at least 10 Å were excluded from the dataset. The remaining ensembles were sequence-culled using the PISCES web server [23]. Standard settings were applied and the identity threshold was set to 35 %. This led to a final dataset comprising 2,856 NMR ensembles (56,189 models).

2.3.6 Consistency X-Ray Ensembles

The second ensembles dataset was built from all available X-ray structures (single chains, similar to the X-ray representatives dataset) of the PDB 2017 copy with a resolution of at least 2 Å and an R-factor below or equal to 0.25. Their corresponding ATOM entry-based sequences were extracted neglecting structures with less than 50 residues. A sequence deconvolution was achieved with the help of USEARCH [43] and the resulting unique sequences were compared to all sequences of the dataset to identify structures with a sequence identity of 100 %. Subsequently, all representative sequences for which more than 10 corresponding PDB structures were found were included in the dataset. We found 104 single chain proteins with at least 10 other structures of identical sequences leading to a dataset of 2,098 structures. This dataset was subsequently sequence-culled to obtain a representative dataset. Using a sequence identity threshold of 35 %, the dataset was sequence-culled with PISCES [23]. The final dataset comprises 84 X-ray ensembles (1,584 structures).

2.3.7 Domains

2.3.7.1 CATH

We designed two datasets to evaluate the impact of SSE assignments on the quality of protein alignments. To this end, we analyzed all domains in the CATH database (Class Architecture Topology Homology) [38] (2018/09/24) with a resolution of at least 2.5 Å and including at least 70 residues. Domains belonging to the same topology or superfamily class were sorted ascending according to the resolution and we retained only the first member per S95 cluster. All domains with an identical topology or superfamily were grouped. The first two domains of each topology or superfamily cluster with at least two members were finally chosen as pairs for subsequent SSE-based domain comparisons. The resulting two datasets contain 1,152 domain pairs of the same superfamily and 393 domain pairs with identical topologies. The topology pairs dataset contains structures with sequence identities ranging from 7.5 % to 53.6 % and of 95.4 % for one pair.

In contrast, the superfamily dataset comprises structures with sequence identities from $8.5\,\%$ to $100\,\%$.

2.3.7.2 ECOD

The ECOD dataset (Evolutionary Classification Of protein Domains) [44] is based on a hierarchical evolutionary classification of protein domains with an emphasis on distantly related homologs. For our subset of the ECOD dataset, we extracted all sequences from the PDB files of the ECOD version develop2016 (2018/12/11). This version contained $141{,}551$ proteins and $639{,}479$ domains. We compared all sequences of the first structure per X group against all others of X group with the help of USEARCH [43] and the information from the ECOD domains file. For each structure, the partner with the lowest sequence identity was selected resulting in one pair per X group and a total of 232 domain pairs. We modified parts of the corresponding PDB files which were relevant for our parsing procedure and, thus, had to comply with the PDB file format. All `ATOM` lines corresponding to ligands were renamed to `HETATM` lines. `TER` lines were added to terminate chains.

2.3.8 Ligand-Sensing Cores

2.3.8.1 LSD1

This datasets consists of three proteins, namely, human monoamine oxidase (MAO) A and B (2bxr@pdb and 1gos@pdb) and lysine-specific demethylase 1 (LSD1, 2ejr@pdb). In 2012, Willmann et al. proposed a shared *ligand-sensing core* in all three proteins [33]. This led to the discovery of Namoline as an LSD1 inhibitor for the impairment of prostate cancer cell growth.

2.3.8.2 APT1

Another common *ligand-sensing core* was published in 2010 by Dekker et al. [34]. It consists of the human acyl protein thioesterase 1 (APT1, 1fj2@pdb) and the dog gastric lipase (1k8q@pdb), and it led to the discovery of Palmostatin B as an APT1 inhibitor.

2.3.8.3 LSC Query and Target

We created a query and a target dataset for the search for common *ligand-sensing cores*. The query dataset consists of split protein chains of the LSD1 (see Section 2.3.8.1) and APT1 (2.3.8.2) datasets. The target dataset is the union of the query dataset and the X-ray representatives dataset (see Section 2.3.2) and contains $3{,}614$ protein chains. In addition to this chain-based dataset, we also created a pocket-based dataset by the use of P2Rank [45, 46]. In all query and target chains pockets were identified using default settings. The `*.atm` files of the three highest ranked pockets per chain were used resulting in 27 query and $10{,}818$ target pockets.

2.4 Development Environment

The following environment was used for the development and the evaluation of all introduced tools, namely, SCOT, SNOT, and SLOT described in Chapters 3, 4, and 5 respectively:

- **Workstation**
  Intel® Xeon® CPU E5-2690
  32 cores @ 2.9 GHz
  128 GB DDR3 RAM
  3 HDs in RAID level 0
  Scientific Linux 7.2
- **GCC 5.3.1**
- **C++ 11**
  No additional libraries, STL only
- **Python 2.7.5**

All tools are written in C++ 11 solely using STL libraries and without the requirements for additional external libraries. However, OpenMP is used for parallelization purposes, but a serial execution fallback is supported in case this library is not available. We avoided the use of additional libraries and chose to implement all algorithms, data structures, and in- and output procedures by ourselves, to provide a maximum grade of versatility and the possibility for tailor-made optimizations.

2.5 Availability

All datasets except the copies of the PDB are available at
`https://ls11-www.cs.tu-dortmund.de/people/brinkjost/datasets-phd-thesis.tar.gz`

The (Linux) executables for SCOT, SNOT, and SLOT are available after publication and on request at `https://this-group.rocks`

"There must be a beginning of any great matter, but the continuing unto the end until it be thoroughly finished yields the true glory."

Sir Francis Drake

3

SCOT | Classifying Secondary Structure Elements

3.1 Introduction

The first publications addressing the automated assignment of secondary structure elements (SSE) date back into the 1970s [47]. The impact of automated secondary structure assignment methods (SSAMs) reaches from secondary structure prediction over secondary structure-based protein alignment to the assignment of protein domains. The high number of already published algorithms for the classification of helices, sheets, and turns in proteins points toward the most challenging issue in secondary structure assignment: we cannot strive for the correct answer when it comes to secondary structure assignment [16]. DSSP [22] is the oldest still available method for the automated identification of regular helical and extended backbone structures in proteins. It is still actively developed and maintained by the CMBI (Centre for Molecular and Biomolecular Informatics). As one of the predominantly applied methodologies for a huge number of structure-based modeling approaches, it is usually applied to assign the SSEs in newly released protein structures. Various modified versions exist as part of structural alignment and visualization tools [17, 32]. Strikingly, it is still the most cited tool for secondary structure assignment although at least thirty alternative approaches have been developed (see Table 3.1).

So, what are the criteria that prompt scientists to prefer one tool over another? Besides consensus

Method	DH	HB	GO	Year	Citations	Citations/Year	Add. M./A.
PSSC [48]	●	●		2014	7	1.75	
SHAFT [18]	●	●		2011	2	0.29	
DSSP – PPII [49, 50]	●	●		2011	36	5.14	
STRIDE [51]	●	●		1995	1,524	66.26	
KAKSI [52]	●		●	2005	81	6.23	●
SEGNO [53]	●		●	2005	34	2.62	
DISICL [54]	●			2014	12	3.00	
PROSS [55]	●			1999	149	7.84	
Chen et al. [56]		●	●	2009	n/a	n/a	
Levitt & Greer[47]		●	●	1977	463	11.29	
beta-Spider [57]		●	●	2005	15	1.15	
DSSPcont [58]		●		2003	65	4.33	
SECSTR [59]		●		2002	140	8.75	
PROMOTIF [60]		●		1996	885	40.23	●
SSTRUC [61, 62]		●		1990	210	7.50	
DSSP [22]		●		1983	9,848	281.37	
RaFoSA [63]			●	2016	n/a	n/a	
SACF [64]			●	2016	2	1.00	
ASSP [65]			●	2015	8	2.67	
Kneller & Hinsen [66]			●	2015	1	0.33	
PCASSO [67]			●	2014	2	0.50	●
SST [40]			●	2012	12	2.00	
SABA [68]			●	2011	7	1.00	
PMML [69]			●	2011	2	0.29	
PROSIGN [70]			●	2008	9	0.90	
PALSSE [71]			●	2005	29	2.23	
Taylor et al. [72]			●	2005	15	1.15	
Zhang & Skolnick [11]			●	2005	1,038	79.85	●
VoTAP [73]			●	2004	41	2.93	
STICK [74]			●	2001	35	2.06	
XTLSSTR [75]			●	1999	74	3.89	
P-SEA [76]			●	1997	81	3.86	
YASSPA (GETSSE) [77]			●	1997	296	14.10	●
P-CURVE [78]			●	1989	106	3.66	
DEFINE_STRUCTURE [79]			●	1988	308	10.27	
SKSP [80]	●	●	●	2007	14	1.27	
CONSENSUS/TCM [81]	●	●	●	1993	109	4.36	

Table 3.1: SSAM citation counts. Overall and annual citation counts for published SSAMs grouped by their underlying methodology based on dihedral angles (DH), hydrogen bonds (HB), or geometry (GO). A feature can be fully (●) supported. A (●) indicates that it is not applicable. β-Spider uses only the contact energy (●). The Web of Science (2018/10/11) Core Collections were used to evaluate the number of citations per SSAM. The column Add. M./A. indicates if the publication of the respective SSAM also contains additional methods or analyses leading to citations that must not be attributed to the published SSAM.

approaches [80], the only possibility to find the most suitable method is a close examination of the individual strengths and weaknesses of different approaches, their availability, and their general applicability toward the type of structural elements under investigation. The definition of helix termini, kinks, and the impact on secondary structure-based protein alignments are further quality criteria. Different approaches have been developed to tackle some but not all of these aspects. Therefore, we set out to combine different aspects of SSE assignments to cope with all of the current challenges using a single method. These aspects are covered by dedicated sections in Chapter 4 underlining the strengths of SCOT concerning the assignment of right- and left-handed helices including rare helix classes, β-sheets and extended conformations in general, structure quality and flexibility, and its huge impact on the SSE-based alignment quality.

This chapter is organized as follows: Section 3.2 describes the state of the art of already available and published SSAMs. These are grouped based on the type of data utilized for the assignment (hydrogen bonding patterns, dihedral angles, geometric properties). It also gives examples of some of these methods which were used for comparison purposes (see Section 4.6). Section 3.3 motivates SCOT itself and its unique characteristics, and describes its methodology including major interim development milestones. More precisely, this section covers the classification of turns, helices, and β-sheets, and the parsing and writing of Protein DataBank (PDB) files. Section 3.4 depicts aspects of our reimplementation of the SHAFT algorithm. Section 3.5 gives insights into the motivation of the evolutionary steps of our classification algorithms and related results. Finally, Section 3.6 discusses SCOT in general and motivates open challenges.

This chapter focuses on the development and evaluation of SCOT itself. Chapter 4 introduces criteria for the comparison, evaluation, and differentiation of SSAMs and therefore covers the detailed evaluation of SCOT with respect to other SSAMs.

3.2 State of the Art

The already available and published SSAMs can be crudely divided based on the type of data utilized for the assignment (hydrogen bonding patterns, dihedral angles, geometric properties). In the following, we will give examples of some of these methods which were used for comparison purposes in Section 4.6. We will outline the basic assumptions underlying the algorithms to highlight the major differences.

3.2.1 Hydrogen Bonding

3.2.1.1 DSSP

DSSP (Define Secondary Structure of Proteins) [22] utilizes dynamic programming for its SSE assignments starting with the detection of hydrogen bonds. An electrostatic interaction energy is calculated to identify distinct hydrogen bonds with an energy cut-off of $-0.5\,\mathrm{kcal/mol}$. This cut-off

was chosen to address errors in coordinates and to support bifurcated hydrogen bonds. These hydrogen bonds are used to identify turns of lengths 4 to 6 and bridges. The turns are used to form minimal helices and bridges to form ladders and, subsequently, β-sheets.

A minimal helix consists of two consecutive turns of identical length. For instance, two consecutive turns of length 4 starting at residues r_i and r_{i+1} define a minimal helix of type 4 spanning residues r_{i+1} to r_{i+4}. Overlapping minimal helices define the longer helices, i.e. α-, 3_{10}-, and π-helices based on minimal helices of types 5, 4, and 6 respectively.

A bridge between two residues r_i and r_j is assigned if there are hydrogen bonds $hb^-_{i-1,j}$ and $hb^+_{i+1,j}$ or $hb^+_{i,j-1}$ and $hb^-_{i,j+1}$ (parallel), or $hb^+_{i,j}$ and $hb^-_{i,j}$ or $hb^-_{i-1,j-1}$ and $hb^+_{i+1,j+1}$ (anti-parallel). Consecutive bridges of identical type form a ladder. One or more multiple ladders connected by shared residues form a β-sheet. Single residue ladders are not regarded as a β-sheet and reported separately. Irregularities in β-sheets are supported by the incorporation of β-bulges. A β-bulge connecting two (perfect and consecutive) ladders or bridges allows on one side of the ladder a gap of non-hydrogen bonded residues of up to 4 instead of 1.

In the case of overlapping SSEs, a hierarchical classification is applied to obtain unique residue assignments. A further development of the algorithm (DSSP 2.0) was realized by adjusting this helix type assignment hierarchy [82]. Consequently, π-helices (which were previously rarely classified) are detected and referred to as α-bulges.

Left-handed helices can be assigned based on the per residue (r_i) chirality information which results from the evaluation of the Virtual torsion angle (Vtor) of the Cα atoms of residues r_{i-1}, r_i, r_{i+1}, and r_{i+2}.

The information of bends is provided for the entire protein chain. A bend at residue r_i is defined whenever the angle at its Cα atom spanned by the Cα atoms of residues r_{i-2} and r_{i+2} exceeds a threshold of 70°.

3.2.2 Hydrogen Bonding and Dihedral Angles

3.2.2.1 STRIDE

Addressing the issue of the highly permissive nature of the hydrogen bond definition used by Kabsch and Sander [22], Frishman and Argos proposed the method STRIDE (STRuctural IDEntification) [51] for secondary structure assignment. The algorithm incorporates a novel, more restrictive, hydrogen bond definition and also geometric constraints based on the residues' dihedral angles. Multiple parameter (weights, statistical residue occurrences, thresholds) optimization steps were performed to ensure a high correspondence to experimentalist-derived classifications formerly used for protein structures published in the PDB. For this purpose, they filtered all X-ray and NMR structures of the PDB to obtain a (sequence) redundant set consisting of 226 proteins chains with manually assigned SSE annotations by the authors of the respective structures (`HELIX` and `SHEET`

lines).

Similar to the DSSP algorithm, STRIDE defines minimal helices based on consecutive turns of the same length. In contrast however, minimal helices are not limited to exactly two consecutive turns but may contain more, which makes a subsequent merging of minimal helices superfluous. Furthermore, each involved turn has to fulfill a condition based on optimized weight factors and residue occurrences, both with respect to values typical for this type of helix. The terminal residues of a (minimal) helix are only considered as part of this helix if their dihedral angles and residues occurrences are typical for this type of helix.

The identification of β-strands is based on the identification of bridges. In addition to the four hydrogen bonding patterns introduced by DSSP and a new pattern that involves three hydrogen bonds, conditions based on weight factors and residue occurrences, both with respect to values typical extended conformations, have to be fulfilled to form a bridge. Neighboring bridges are merged if there is at most 1 intervening residue on one strand and are at most 4 intervening residues on the other strand between both bridges.

The application of the method leads to the identification of three helix classes (α, 3_{10}, π) and strands (no β-sheet affiliation) as well as turn structures in proteins. The methodology is incorporated in the SSE analysis tool of VMD [83].

3.2.2.2 SHAFT

Based on a comprehensive assignment of turn types in proteins, SHAFT (Secbase automated Helix Assignment adapted From Turns) [18] was developed as an alternative helix class assignment. It relies on a dihedral angle-based turn classification (*normal*, i.e., hydrogen-bonded turns, and *open* turns, i.e., turns of different lengths whose Cα–Cα distances are equal to or less than 10 Å). Continuous stretches of overlapping turns of identical types are used to assign α-, 3_{10}-, π-, and γ-helices, but a strand definition is missing. Residue assignments to more than one helix class are avoided by a hierarchically applied helix class assignment. Nonetheless, overlapping helices are assigned.

The applicability of the above algorithms is restricted to protein structures with known backbone atoms. In case of Cα-only structures, as obtained for crystal structures of low resolution or some electron microscopy models [68], these methods cannot be applied unless the complete backbone is reconstructed based on the Cα trace with further methods, e.g., SABBAC [84]. However, only 368 (0.27 %) out of 138,384 structures of the 2018 copy of the PDB are Cα-only structures. Another weak point is the sensitivity of more restrictive hydrogen bond definitions toward structural perturbations and model building artifacts due to a low resolution or a high flexibility of some protein regions. These and other flaws of hydrogen bonding-based assignment software led to the development of novel tools which are based on geometric criteria and/or dihedral angles.

3.2.3 Geometry

3.2.3.1 ASSP

The method ASSP (Assignment of Secondary Structure in Proteins) [65] employs geometric criteria to assign SSEs. It is an extension of the software HELANAL-Plus [85] which assigns helical protein segments. In ASSP, the path traversed by the Cα atoms is used to classify continuous stretches in proteins as α-helices, 3_{10}-helices, π-helices, left-handed (α-, 3_{10}- and π)-helices, β-strands, and PPII helices.

3.2.4 Dihedral Angles

3.2.4.1 DISICL

For DISICL (DIhedral-based Segment Identification and CLassification) [54] assignments, the protein is divided into segments of two residues. Based on torsion angle region definitions, the residue segments are divided into altogether 18 structural classes. The final assignments are based on the pairing of the corresponding regions. We use a set of DISICL-derived classes to assign α- , 3_{10}-, π-, and PPII helices as well as left-handed helices. The latter are not further differentiated to left-handed α-, 3_{10}-, or π-helices. Additionally, β-strands are assigned using the β-strand and β-cap definitions.

3.2.5 Geometry and Dihedral Angles

3.2.5.1 SEGNO

The program SEGNO [53] combines geometric parameters (e.g., distances and curvature-defining angles) with the information on dihedral angles. The software assigns α-, 3_{10}-, and π-helices, β-strands, and PPII helices, as well as mixed helices as combinations of α-, 3_{10}-, and/or π-helices. Length constraints (minima in the number of required residues) for the different types are applied to ensure the classification of continuous structural stretches. Additionally, a Ramachandran outlier detection [86] leads to the exclusion of residues from the secondary structure assignment.

3.3 SCOT

While purely hydrogen bond-based methods might fail to identify geometrically regular continuous stretches of SSEs and are not applicable to regular conformations which are not stabilized by main-chain hydrogen bonds (PPII), geometry- and dihedral angle-based methods might overestimate the number of stable SSEs found in proteins. We have developed SCOT (Secondary structure

Classification On Turns) to find an acceptable compromise between both approaches (hydrogen bond and geometry). Repetitive stretches of hydrogen-bonded and *open* turns of different lengths, which are classified based on their dihedral angles and geometric distance criteria, were used to develop and benchmark a novel alternative for a reliable and consistent assignment of SSEs in proteins. Inspired by the work of Koch and Cole [18], we incorporate the knowledge of hydrogen-bonded and non-hydrogen-bonded turns to assign multiple helix classes. The assignment of helices and β-strands are based on both, turn data and four-residue segment Cα–Cα distances. The latter are mainly applied to avoid irregularities in the assigned SSEs. SCOT was developed focusing on bottlenecks, such as geometric uniformity, stability, consistency across structure ensembles, and the incorporation of rare SSEs.

We have developed SCOT from scratch without using external libraries to be able to have our hands on all aspects of the assignment. Having said that, we had to to develop procedures for the parsing and writing of PDB files described in Sections 3.3.1 and 3.3.5. The algorithms to assign SSEs are presented in the same order in which they are realized by SCOT. In Section 3.3.2 we describe the assignment of *normal*, *reverse*, and *open* turns which are the basis for all other SSEs assigned thereafter. In Section 3.3.3 our three algorithms to assign β-sheets are presented. For the assignment of helices, we have developed five different algorithms which are described in Section 3.3.4.

Each algorithm (β-sheets and helices) is an advance to its predecessor. Thus, the last algorithm of each section is the most recent and published version. Furthermore, all predecessors may contain (logical) pitfalls that are dealt with in each following or at least in the final algorithm. All algorithms are named according to their main feature or data structure.

Please note that all assignment algorithms process each chain of a protein separately. Therefore, the assignment of β-sheets spanning multiple chains is not supported.

SCOT is able to process single structures as well as folders containing all structures of interest with built-in support for parallel execution.

SCOT is written in C++ and parallelized using OpenMP.

3.3.1 Input

SCOT requires standard PDB files as input. All options can be set via command line arguments. A list of all supported arguments and a short documentation can be evoked using `--help`. In addition, trained ESOM files are required for the dihedral angle-based classification of turns.

3.3.1.1 PDB Files

The standard PDB file format was created in the 1970's and is supported by a large number of software for a multitude of applications. Each protein is represented by an individual file identified

by its PDB identifier. Each file contains meta information, such as the organism the protein is obtained from or experimental data, atom coordinates, sequence information, secondary structure information and more. The information is represented in a fixed column width format. Each line has a fixed length of 80 characters and contains a six-characters long prefix which denotes the type of the line and its (column) format for the remaining characters. However, the fixed column width format limits the support to protein structures consisting of at most 26 chains. This is due to the fact that there is only a single character reserved to specify the chain identifier (not counting digits). For such proteins, only files in the newer PDBML format are available. Nevertheless, we chose to support the standard PDB file format due to its acceptance by a wide range of software.

Our PDB file parsing procedure relies on the information given in the lines with the following prefixes: `REMARK 465` (for the support of missing residues), `SEQRES` (for the support of modified residues), `ATOM`, `HETATM` (modified residues), `TER`, and `ENDMDL` in NMR structures. We parse the residues according to the `ATOM` and `HETATM` lines in the order of their appearance. We distinguish between ligand and modified (protein) residues based on the `SEQRES` information. If the name of a residue in the `HETATM` lines is present in the `SEQRES` entry of the corresponding chain, it is (assumed to be a modified residue and) parsed as a protein residue. If there are alternate locations given for the atoms of a residue, we retain the conformation with the highest occupancy. If there are multiple conformations with equal and highest occupancies, we retain the first appearing conformation of these. Please note that the selected conformation applies to all atoms of a residue but not to all atoms of an entire chain or protein structure. Although the PDB file format specifies that the `ATOM` records should be listed from the amino to the carboxyl terminus, some PDB files provide different orderings, such as 2xmj@pdb. In such a case, we reorder the atoms accordingly.

Missing residues indicate residues that are known to be at a specific position in the sequence of a chain, but their coordinates are not available. We insert them in two ways. In general, missing residues have to be inserted at the sequence positions and insertion codes for which they are stated as missing. However, for some proteases, such as 1jou@pdb, missing residues, which have to be inserted in front of the first residue, share the same sequence position (e.g., 1) but a descending insertion code. Therefore, if the current residue r is the first non-missing residue to be inserted and if r has an insertion code, we add all missing residues that have the same sequence position as r and in the order of their appearance in the `REMARK 465` table. All other missing residues are added to the parsed sequence at their stated sequence positions. We also set some residues with modified backbone structures or covalently linked ligands as missing and do not parse their underlying atom information for the turn and secondary structure assignment. A list of these residues can be found in List 6.2 of the appendix.

Most X-ray structures do not include hydrogen atoms. However, the backbone nitrogen hydrogen atoms are essential for the determination of hydrogen bonds. Therefore, we assign the position of the backbone hydrogen atoms according to McDonald and Thornton [20]. We do not use input hydrogen atoms to obtain a consistent assignment. For each residue r_i, the hydrogen atom is placed in the plane with the carbonyl carbon (C) atom of the preceding residue r_{i-1}, and the N and the Cα atom in a distance of 1 Å to the nitrogen (N) atom but 4° closer to the Cα atom of r_i (see Figure 3.1). We only assign hydrogen atoms to residues with open valences at their backbone

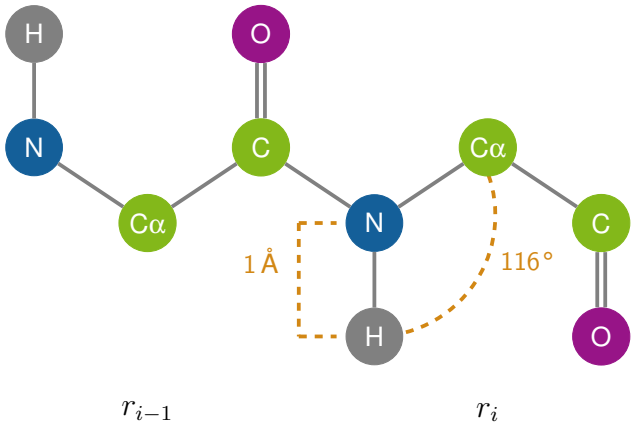


Figure 3.1: Visualization of the hydrogen atom placement. Visualization of the hydrogen atom (H) placement for residue r_i according to McDonald and Thornton [20]. This Figure is extracted from [7].

nitrogen atoms. A list of excluded residues is given in List 6.1 of the appendix. Predefined input hydrogens can also be used if hydrogens are provided in the input files, by the use of Reduce [87], for instance. The command line option `--input-hydrogens` disables the assignment of hydrogen atoms and forces SCOT to solely use the input hydrogen atoms. If this option is not given, we still use an input hydrogen atom of a residue if the reassignment of the hydrogen atom failed. This can happen due to missing backbone atoms C, O, or Cα.

We only parse the first model of NMR structures as the PDB file format itself does not support individual secondary structure annotations for each model.

The parsed residue sequence order can be validated with respect to the information provided in the `SEQRES` lines. If the command line option `--validate-sequence` is set, each parsed sequence mismatch and the parsed sequence itself is prompted to the terminal.

3.3.1.2 ESOM Files

In addition to a PDB protein file, SCOT requires files containing the trained ESOMs. There are two files for each turn category and length (e.g., *normal-5*) required to load its corresponding ESOM and assign turn classes. The weight file (`*.wts`) contains the weights of the neurons whereas the class mask file (`*.cmx`) contains the classification of the neurons. Their formats are specified at the webpage corresponding to [24]. All files have to be named according to their corresponding turns (e.g., `normal-5.wts`). Their location, if not present at the location of the executable, can be specified via the option `-e`. This also enables the use of user-defined novel turn classifications.

3.3.2 Turns

Turns are the fundamental elements for our classification of helices and sheets. We organize turns in three different families, namely, *normal*, *reverse*, and *open* [88], with different lengths

(categories). An overview of all classified turn categories is given in Figure 3.2.

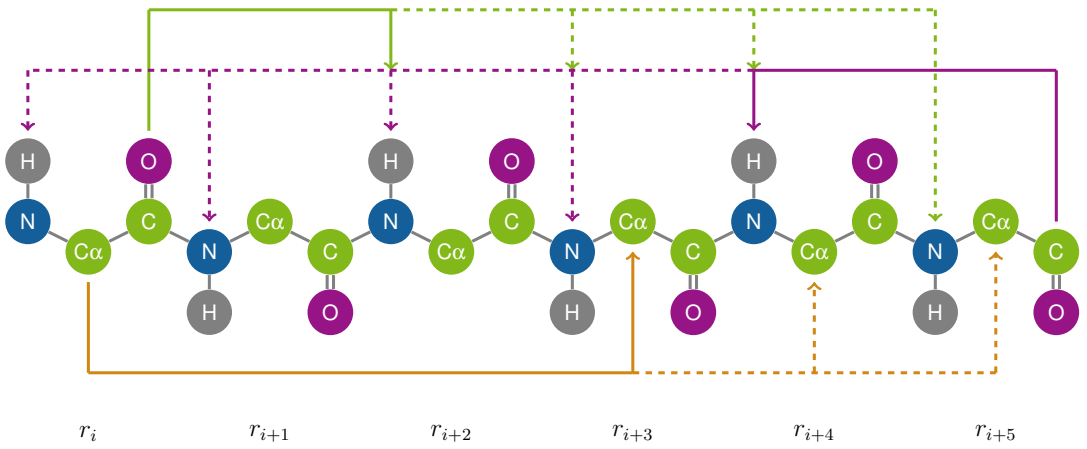


Figure 3.2: Visualization of the turns of all categories and lengths classified herein. *Normal* turns are highlighted in green, *reverse* turns in purple, and open turns based on a Cα–Cα atom distance between 4 Å and 8 Å in orange. This Figure is extracted from [7].

Our classification of turns, which improves the former turn classification by Koch and Klebe [88], is realized in two steps. First, we detect all hydrogen-bonded (*normal* and *reverse*) and non-hydrogen-bonded (*open*) turns. The latter are limited to turns with Cα–Cα distances between the first and the last residue of at most 8 Å. Second, we assign classes to each detected turn which are based on the respective dihedral angles.

The assignment of the turn categories starts with the determination of sequence regions that do not contain missing residues or residues with an atypical backbone structure (missing N, Cα, C, O, or OXT atoms). Within each such region, we detect hydrogen bonds for *normal* and *reverse* turns using the criterion defined by Dahiyat et al. [89] (see Equation 3.1). We use the following parameters with values from the original publication given in parentheses: $R_0 = 2.8$ Å, D_0 = 8 kcal/mol, 2.5 Å $\leq R \leq$ 3.5 Å (3.3 Å), $\theta \geq 130°$ (135°), and an energy threshold of $E_{HB} \leq -0.5$ kcal/mol (-2 kcal/mol). θ is the N-H-O angle, φ is the H-O-C angle, and φ is the angle between the normals of the two planes defined by the six atoms attached to the sp^2 centers. Open turns are detected by the Cα–Cα distance d between residues r_i and r_{i+k} with $k \in \{3, 4, 5\}$. A turn is assigned if the distance d is within the distance thresholds: 4 Å $\leq d \leq$ 8 Å. For *open-4 9* turns, an additional class assignment of turns with a distance 4 Å $\leq d \leq$ 10 Å is performed which enables the classification of PPII helices. For a given residue r_i, we detect *normal* contacts to residues r_{i+k} with $k \in \{2, 3, 4, 5\}$ from the O atom of r_i to the H atom (if present) of r_{i+k}. *Reverse* contacts are detected from the O atom of r_{i+k} to the H atom of r_i with $k \in \{1, 2, 3, 4, 5\}$. In addition, we also take the OXT atom (if present) of the C-terminal residue into account and only keep the strongest contact if contacts for both atoms (O, OXT) were detected. For all other residues, we do not drop multiple hydrogen bond contacts based on their energy. We keep all detected contacts and provide their energy according to Dahiyat et al. in the output. For open turns, the Cα–Cα distances are

reported.

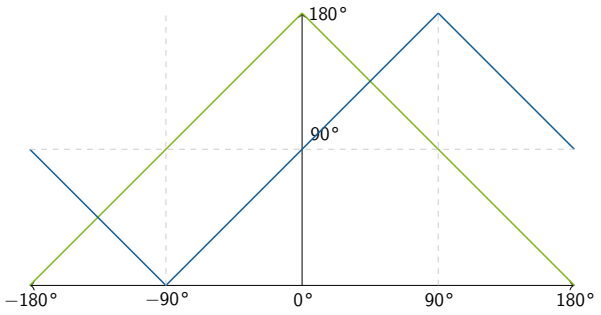
$$E_{HB} := D_0 \left\{ 5 \left(\frac{R_0}{R} \right)^{12} - 6 \left(\frac{R_0}{R} \right)^{10} \right\} \cos^2(\theta) \cos^2(\max(\varphi, \psi)) \tag{3.1}$$

The classification of the turns, detected in the previous step, utilizes ESOMs [24]. We clustered turn conformations based on the Jigsaw-transformed dihedral angles in the X-ray representatives dataset. ESOM training and clustering were performed utilizing the Databionic ESOM Tools [24]. For a *normal* or *open* turn t on residues r_i to $r_{i+|t|-1}$, we use the dihedral angles $\omega_i, \varphi_{i+1}, \psi_{i+1}, \ldots, \omega_{i+|t|-2}$. For a *reverse* turn t, we use $\varphi_i, \psi_i, \omega_i, \ldots, \psi_{i+|t|-1}$. These dihedral angles are transformed using our Jigsaw transformation which consists of two functions $f_1 : [-180°, 180°] \to [0°, 180°]$ and $f_2 : [-180°, 180°] \to [0°, 180°]$ given in Figure 3.3. Due to the bisected value range *collisions* appear: $f_1(-90°) = f_1(90°)$. To address this problem, f_2 is most discriminative whenever f_1 is not and vice versa, e.g., $f_2(-90°) = 0°, f_2(90°) = 180°$. This transformation is necessary for the clustering algorithm which uses a metric distance function d that otherwise fails to accurately reflect the distances in angular space. For instance, the numeric distance $d(-179°, 179°)$ is $358°$ but should be $2°$. Using the Jigsaw-transformed value, the distance functions report a small distance: $d(f_1(-179°), f_1(179°)) = 0°$, $d(f_2(-179°), f_2(179°)) = 2°$.

$$f_1(x) := \begin{cases} +x + 180° & -180° \leq x < 0° \\ -x + 180° & \text{otherwise} \end{cases}$$

$$f_2(x) := \begin{cases} -x - 90° & -180° \leq x < -90° \\ +x + 90° & -90° \leq x < 90° \\ -x + 270° & \text{otherwise} \end{cases}$$

(a) Formal definition of the Jigsaw transformation.



(b) Visualization of the Jigsaw transformation functions.

Figure 3.3: The Jigsaw transformation. This Figure is extracted from [7].

ESOMs were trained by visually analyzing and comparing turns and their transformed backbone dihedral angles in different clusters. Both Jigsaw transformation functions were applied to each backbone dihedral angle of a turn resulting in two transformed angles for each input dihedral angle. Clusters of turns with similar angles were assigned a numeric class based on the population size of the cluster. The fewer cluster members, the higher the number. That is, neurons are assigned a class. An overview of the ESOM parameters applied for the different turn categories can be found in Table 3.2. Please be referred to the doctoral thesis by Christiane Ehrt [90] for the details on the ESOM training and turn clustering.

For each turn we search for the neuron with the lowest distance in the corresponding ESOM for the turn category. We assign the turn the class of the respective neuron.

As the ratio between rows and columns should be significantly different from unity, we chose a

Turn type	|Angles|	|Turns|	Multiplier	|Neurons|	Dimensions (R×C)
*reverse-2**	5	977	100	97,700	382 × 256
reverse-3	8	134	100	13,400	142 × 95
reverse-4	11	4,113	100	411,300	785 × 524
reverse-5	14	2,892	100	289,200	658 × 440
reverse-6	17	1,869	100	186,900	530 × 353
normal-3	4	2,190	100	219,000	572 × 383
normal-4	7	83,212	10	832,120	1,117 × 745
normal-5	10	222,529	5	1,112,645	1,291 × 862
normal-6	13	10,964	100	1,096,400	1,283 × 855
open-4	7	99,883	1	99,883	386 × 259
open-5	10	117,894	1	117,894	420 × 281
open-6	13	83,711	1	83,711	354 × 237

Table 3.2: Properties of the ESOMs for each turn category. The feature vectors of the neurons used for the clustering contain two Jigsaw transformed values for each dihedral angle. * For the training set of *reverse-2* turns, we used all protein X-ray structures of the PDB (2017) to obtain an ESOM comprising at least 4,000 neurons which is the required minimum for an optimal training [24]. This Table is extracted from [7].

ratio of 1.5 rows per column. A Gaussian bell-shaped neighborhood kernel function (`-n gauss`) was used. We use an Euclidean grid distance function (`-d euc`) in combination with a (borderless) toroid topology (`-g toroid`) to avoid border effects. The initial radius was set to half the number of columns and the number of epochs to 200 (`-e 200`). The weight vectors were initialized by sampling the hyperplane spanned by largest principal components (`-i pca`).

3.3.3 Sheets

We have developed three different algorithms to assign β-sheets and strands, namely, Stranded (see Section 3.3.3.1), Linked (see Section 3.3.3.2), and Queued (see Section 3.3.3.3).

The Stranded algorithm utilizes seed fingerprints to find the sequence positions for the initial strands which are organized in trees to form β-sheets. The Linked algorithm uses an interim data structure and a graph for the final representation of β-sheets. The most recent algorithm is called Queued and solely relies on graphs as its core data structure. It also utilizes a queue during the extraction of the β-sheet and strand information from the graph. It is also the β-sheet assignment algorithm which is incorporated into the latest version of SCOT.

3.3.3.1 Stranded

The Stranded algorithm assigns β-sheets utilizing seed-fingerprints to detect regions of extended conformations within a protein's sequence and a tree as its core data structure. It performs a merging and shrinking procedure to all strands to obtain a consistent assignment.

The algorithm starts with the identification of intra-backbone hydrogen bonds using the hydrogen bond criterion by Dahiyat et al. [89]. The parameters for the hydrogen bond assignment are described in Section 3.3.2. Let $C_V \subseteq C$ be the set of (valid) residues of chain C that are not marked as missing and that have a backbone hydrogen atom (see Section 3.3.1). For each residue $r_i \in C_V$, we determine the set of (neighboring) residues $N_i \subseteq C_V$ whose oxygen (O) atoms are within a 4 Å proximity of the hydrogen atom (H) of r_i. For all neighboring residues $r_j \in N_i$ that form a hydrogen bond between the H atom of r_i and the O atom of r_j, we add the hydrogen bonds $hb^+_{i,j}$ and $hb^-_{j,i}$ to the set of hydrogen bonds HB. A hydrogen bond is defined if the hydrogen bond criterion returns an energy of at most $-0.5\,\mathrm{kcal/mol}$ for a given donor and acceptor pair. Finally, HB contains the hydrogen bonds for all residues $r_i \in C_V$. All hydrogen bonds in HB are sorted according to the sequence positions of their residues.

The next step deals with the determination of the initial strand positions in the sequence of the chain. For that, we create a binary seed-fingerprint F_E of the size of the chain $|F_E| = |C|$. For each hydrogen bond-based turn t (*normal*, *reverse*) of length $|t| \geq 3$ on residues $r_i, \ldots, r_{i+|t|-1}$, we mark all bits $b_i \in F_E$ corresponding to the inner residues $r_{i+1}, \ldots, r_{i+|t|-2}$ of t. We also set all bits b_i to 1 if the corresponding residue r_i is set as missing or has an invalid backbone (no N, Cα, C, or O or OXT atoms). After this procedure, all segments of consecutive unmarked bits ($b_i = 0$) are defined as seeds. We also create a second fingerprint F_C of the same length, which is used to mark the already analyzed or strand-classified regions of the sequence. This fingerprint is required to detect circularly defined β-sheets, as present in β-barrels, for instance, and to terminate their recursive detection (see Figure 3.4 for a β-barrel).

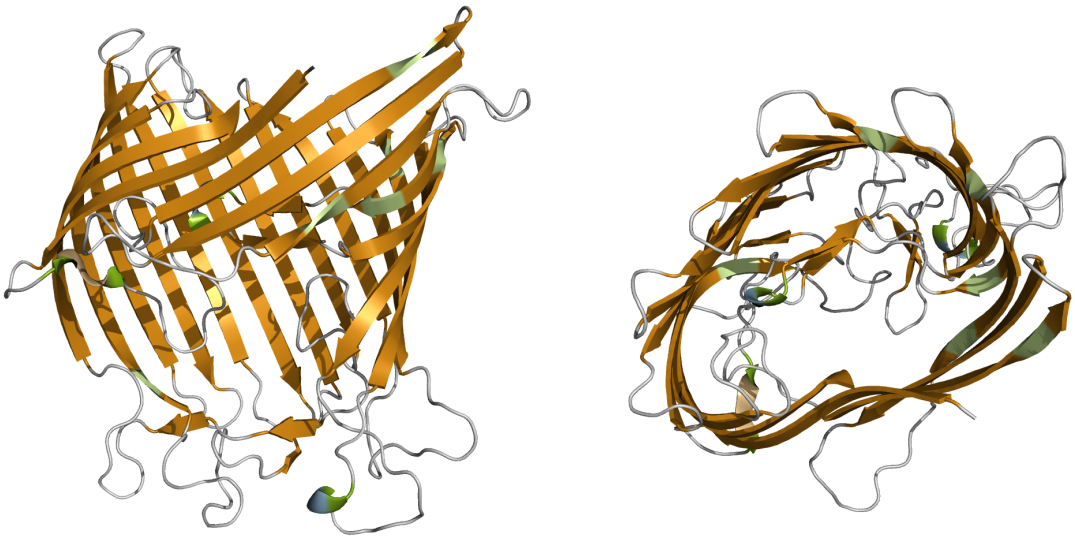


Figure 3.4: Visualization of a β-barrel. Visualization of a β-barrel in 1af6A@pdb assigned by the Queued algorithm (see Section 3.3.3.3) from two perspectives.

Utilizing the fingerprints F_C and F_E, we scan in ascending direction for sequence positions i whose corresponding bits are not marked (0) in both fingerprints. If for such a sequence position i a hydrogen bond $hb^{\pm}_{i,j} \in HB$ exists, we create a new undirected tree $T = (V, E, \tau)$. T represents a sheet and its vertices $v \in V$ represent the strands of that sheet. Each vertex contains its

child vertices, a set of hydrogen bonds to its parent and child vertices, and the sense (parallel or anti-parallel) with respect to its parent vertex. The initial tree contains $v_p \in V$ as the root with one child $v_c \in V$. Both vertices contain the initial hydrogen bond $hb^{\pm}_{i,j}$ but with respect to the (donor-acceptor) direction: $hb^{+}_{i,j}$ at v_p and $hb^{-}_{j,i}$ at v_c or vice versa.

Next, we geometrically determine the sense which can be parallel or anti-parallel. We define a vector for each residue r_i, r_j involved in $hb^{\pm}_{i,j}$. The vector $\overrightarrow{d_i}$ for residue r_i is defined from the N atom of its preceding residue r_{i-1} to the C atom of its succeeding residue r_{i+1}. If the residues r_{i-1} or r_{i+1} do not exist, have no such atoms, or are set as missing, we use the atoms of r_i instead. The vector $\overrightarrow{d_j}$ is defined analogously. If the angle between the vectors $\overrightarrow{d_i}$ and $\overrightarrow{d_j}$ is below 90°, the sense is parallel, and anti-parallel otherwise.

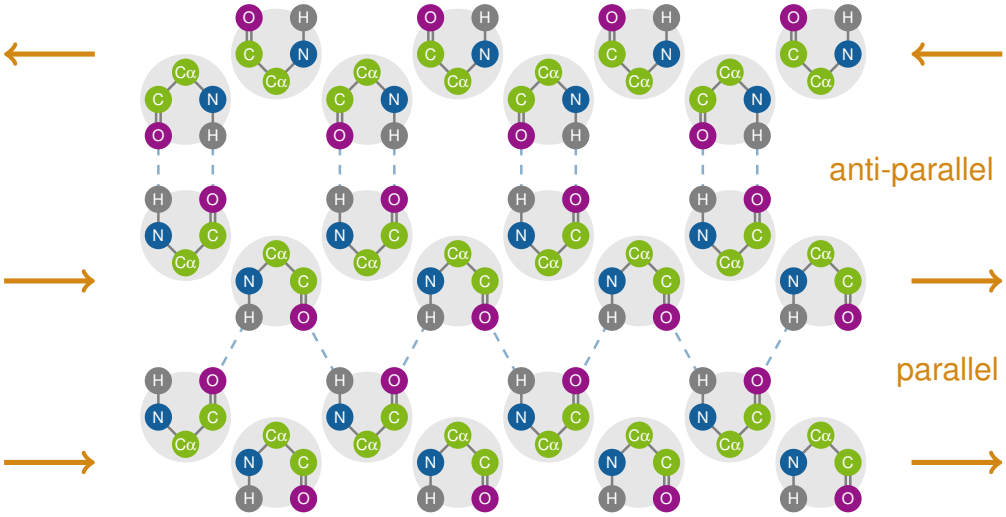


Figure 3.5: Visualization of the parallel and anti-parallel sheet hydrogen bonding patterns. This Figure is extracted from [7].

We search for hydrogen bonds in HB with respect to the sense starting at the initial hydrogen bond to the N- and the C-terminus (see Figure 3.5). Assuming the direction is parallel and the initial hydrogen bond is $hb^{+}_{i,j}$. The next hydrogen bond in N-terminal direction to be searched for is $hb^{+}_{i-2,j}$ and $hb^{-}_{i,j+2}$ in C-terminal direction. Each contact is removed from HB and added to v_p and v_c, again with respect to the direction. We proceed as long as we find hydrogen bonds in HB in compliance with the hydrogen bonding pattern. Once this procedure has stopped, we determine the sequence coverage segment s_p for v_p which spans from the minimum to the maximum sequence position of a hydrogen bond stored at v_p. For all sequence positions $i \in s_p$, we set the corresponding bits $b_i \in F_C$ to 1 to indicate the coverage. In addition, we start searching for hydrogen bonds in HB whose residue's sequence position is within s_p. For each identified hydrogen bond, we add a new child to v_p and proceed as described for v_p and v_c starting with the determination of the sense. As soon as there are no further hydrogen bonds left in HB within the sequence segment s_p, we recursively call this procedure on all children. Whenever the recursive call on a child has ended, we verify the number of hydrogen bonds at the child. If this number is below 2, we delete the child.

The next step of the algorithm is dedicated to the merging of strands. At first, we create a binary

fingerprint F_B containing the size of the chain $|F_B| = |C|$ bits. Each bit is assigned the initial value 0. We set all bits $b_i \in F_B$ to 1 whose associated residues are covered by a hydrogen-bonded turn with a maximum length of 4 of any class (*normal-3* and *-4*, *reverse-2*, *-3*, and *-4*). We extract all sequence coverage segments S from the vertices of all trees and sort them according to their sequence positions. For all two consecutive segments s_j, s_{j+1} and associated vertices v and w, we calculate the sequence distance $d_s(s_j, s_{j+1}) = s_{j+1}.front - s_j.back$. If the segments overlap ($d_s(s_j, s_{j+1}) \leq 0$), or if there is a gap of at most one residue ($d_s(s_j, s_{j+1}) \leq 2$) and the bits $b_i \in F_B$ with $s_j.back \leq i \leq s_{j+1}.front$ are not marked (0), we merge the corresponding vertices. Let v be the corresponding vertex to s_j and w the corresponding vertex to s_{j+1}. Assuming that v is a root vertex, all hydrogen bonds and children of v are moved to w, the sequence segment of w is adopted accordingly and v is removed from the set of trees. If v is not a root vertex, it is removed from its parent. We proceed with the next pair of succeeding vertices until the last pair has been checked.

To assign stable strand termini and to obtain a more consistent assignment, we shrink the sequence coverage segments corresponding to each vertex. Let s_v be the sequence coverage segment corresponding to a vertex v. For each terminus $s_v.front$, $s_v.back$, we calculate the number of hydrogen bonds. Note that there must be at least one hydrogen bond contact that defines each terminus. We increment $s_v.front$ by 1 if the number of hydrogen contacts is 1. We decrement $s_v.back$ analogously. A detailed example of this procedure is given for the Queued algorithm described in Section 3.3.3.3.

Finally, each tree represents a β-sheet and each vertex of a tree a strand of that sheet. The root vertex of a tree is the first strand of a reported β-sheet without a registration. All further vertices of a tree are added to that sheet and each registration, including the sense and the connecting hydrogen bond, is reported with respect to the parent of a vertex. Only strands of length at least 2 are retained. The algorithm stops branching a path in the tree whenever a strand does not fulfill this criterion.

3.3.3.2 Linked

The main difference of the Linked algorithm compared to the Stranded algorithm (see Section 3.3.3.1) is the use of a graph to represent β-sheets and especially support loops as present in β-barrels (see Figure 3.4 for an example of a β-barrel). In contrast to the Stranded algorithm we do not try to classify strands and sheets at once. Instead, we first detect all pairs of strands and combine them later on to a β-sheet graph representation during the merging procedure.

The algorithm also starts with the identification of backbone hydrogen bonds HB and the creation of a seed-fingerprint F_E similar to the Stranded algorithm. Next, we use an interim data structure consisting of strand hydrogen bonds $HB_S \subseteq HB$ and strand sequence segments S_S. HB_S contains all hydrogen bonds that are classified in strands. S_S contains all sequence segments that are assigned as hydrogen-bonded strands. In other words, S_S contains the residue segments and HB_S the associated hydrogen bonds (including the sense) connecting them to form sheets.

We scan in ascending direction for sequence positions i whose corresponding bits $b_i \in F_E$ are not marked ($b_i = 0$). If for such a sequence position i a hydrogen bond $hb^{\pm}_{i,j} \in HB$ exists, we pop $hb^{\pm}_{i,j}$ from HB and determine the sense using the geometric criteron as described for the Stranded algorithm. We create a new pair of sequence segments s_p and s_c. s_p initially encompasses solely the sequence position i of the initial hydrogen bond $hb^{\pm}_{i,j}$. s_c solely encompasses j analogously. These segments are extended whenever we find a hydrogen bond in HB with respect to the sense starting at the initial hydrogen bond to the N- and the C-terminal direction (see Figure 3.5). Each identified hydrogen bond is removed from HB and, including the determined sense, added to HB_S. Assuming the direction is parallel and the initial hydrogen bond is $hb^{+}_{i,j}$. The next hydrogen bond in N-terminal direction to be searched for is $hb^{+}_{i-2,j}$ and $hb^{-}_{i,j+2}$ in C-terminal direction. If $hb^{+}_{i-2,j}$ exists, $s_p.front$ is set to $i-2$. If $hb^{-}_{i,j+2}$ exists, $s_c.back$ is updated to $j+2$. We proceed as long as we find hydrogen bonds in HB in compliance with the hydrogen bonding pattern. If at the end of this procedure at least one hydrogen bond has been found, we add the initial hydrogen bond to HB_S and resume scanning for sequence positions with unmarked bits in F_E to find further pairs of strands. Otherwise, HB_S is empty and we drop the two segments s_p and s_c which are solely based on the initial (single) hydrogen bond $hb^{\pm}_{i,j}$.

Finally, after the scan is finished, S_S contains all sequence segments classified as strands and HB_S all hydrogen bonds these segments are based on. Note that although we classify strands in pairs, we do not save them as pairs but as single segments.

The strand merging procedure of the algorithm creates a blocking fingerprint F_B similar to the Stranded algorithm. We then sort all strand sequence segments S_S according to their sequence positions. For each consecutive pair of segments $s_j, s_{j+1} \in S_S$, we calculate the sequence distance $d_s(s_j, s_{j+1}) = s_{j+1}.front - s_j.back$ similar to the Stranded algorithm and merge the two segments if the same conditions hold true. If s_j and s_{j+1} are merged, $s_{j+1}.front$ is set to the minimum and $s_{j+1}.back$ to the maximum of the corresponding limits of both segments. s_j is removed from S_S and we proceed checking the distance between s_{j+1} and its successor s_{j+2}.

Before we create the graph to combine the strands (segments) to sheets, we also shrink the strands in the same manner as described for the Stranded algorithm. The main difference is that each segment is self-sufficient. Thus, if a segment does not fulfill the minimum length requirement and gets deleted, it does not require a reorganization of the data structure (formerly tree). Plus, the segments are given explicitly and not implicitly by the hydrogen bonds at each vertex.

We now create a graph $G = (V, E)$ based on the segments $s \in S_S$ to combine the strands (segments) to sheets. All segments are distinct and do not overlap because otherwise two overlapping segments would have been merged. In addition, the shrinking procedure does not create new overlaps as it only shrinks the segments. The creation of the graph is done in two steps. First, we create a vertex $v \in V$ for each segment $s \in S_S$ and label it with s. Second, we add edges with respect to the strand hydrogen bonds HB_S. For each hydrogen bond $hb^{\pm}_{i,j} \in HB_S$, we determine the vertices v and w for whose segments s_v contains i and s_w contains j analogously. If v and w are not adjacent, we add an edge $e = (v, w)$ to the set of edges E of G and label it with $hb^{\pm}_{i,j}$ including its direction (hb^{+} or hb^{-}) and sense (parallel or anti-parallel). After all hydrogen bonds in HB_S have been added to G, each connected component in G represents a β-sheet and

each vertex in such a component a strand of that sheet.

The final step deals with the extraction of the sheets and corresponding strands from G. As long as there are vertices V, we pick a vertex v from G, create a new β-sheet consisting of a strand covering the residues of the vertex' segment. We add each neighbor v_n of v as a new strand to the sheet utilizing the hydrogen bond and sense information stored at the incident edge $e = (v, v_n)$ for the strand's registration. This procedure is recursively called on the neighbors of v_n. Each vertex that has been processed is marked as such to provide a termination criterion in loop-defined sheets, i.e., β-barrels. As soon as this procedure stops, i.e., all vertices of a connected component have been processed and added to the sheet, we remove all of its vertices from G.

3.3.3.3 Queued

The Queued algorithm is our most recent one for the assignment of β-sheets. Similar to the Linked algorithm described in the previous Section 3.3.3.2, it utilizes graphs but avoids the necessity of the interim data structure of strand hydrogen bonds and strand sequence segments. Their information is directly modeled in the graph. It also introduces a kink detection procedure and a queue-based extraction of vertices from the graph to obtain sequence ordered β-sheets. Furthermore, it limits the detection and processing of hydrogen bonds to the donor-acceptor direction as each hydrogen bond $hb^{+}_{i,j}$ implies $hb^{-}_{j,i}$.

The algorithm starts with the detection of all hydrogen bonds HB similar to the algorithms discussed before but only takes the donor-acceptor direction ($hb^{+}_{i,j}$) into account. Furthermore, we take all hydrogen bonds in HB into account and do not limit their usage to seed regions.

Using this hydrogen bond information, we create an undirected labeled graph $G = (V, E)$ to group these hydrogen bonds to strands. Each vertex $u \in V$ represents a strand without an explicit label. Particularly, the vertices do not store any information apart from their incident neighbors $u \in V : (v, u) \in E$. Each edge $e \in E$ with $e = (v, w)$ is labeled with the hydrogen bonds between the strands represented by the incident vertices v and w. If there are no hydrogen bonds between the vertices v and w, the vertices are not adjacent: $e = (v, w) \notin E$.

As long as HB is not empty we pop the first hydrogen bond $hb^{+}_{i,j}$ with respect to the sequence position from HB and geometrically determine the sense (parallel or anti-parallel) identical to the previous algorithms. Next, we search for hydrogen bonds $HB_S \subseteq HB$ with respect to the sense starting at the initial hydrogen bond to the N- and the C-terminus (see Figure 3.5). Note that HB_S initially contains hb. The next hydrogen bond in N-terminal direction to be searched for is $hb^{+}_{i-2,j}$ and $hb^{+}_{j+2,i}$ in C-terminal direction. In contrast to the previous algorithms, we only take the donor-acceptor direction into account. Thus, whenever we require $hb^{-}_{i,j}$ we search for the corresponding $hb^{+}_{j,i}$ hydrogen bond. Each found hydrogen bond is removed from HB and added to HB_S. If searching in both directions has stopped and if $|HB_S| \geq 2$, a new pair of vertices v, w is added to the graph G and connected by an edge $e = (v, w)$. e is labeled with the contacts HB_S and the sense. The procedure stops as soon as HB is empty. At this stage, the graph contains an even number of vertices $|V| \bmod 2 = 0$ and $\frac{|V|}{2}$ pairs of distinct adjacent vertices (connected

components).

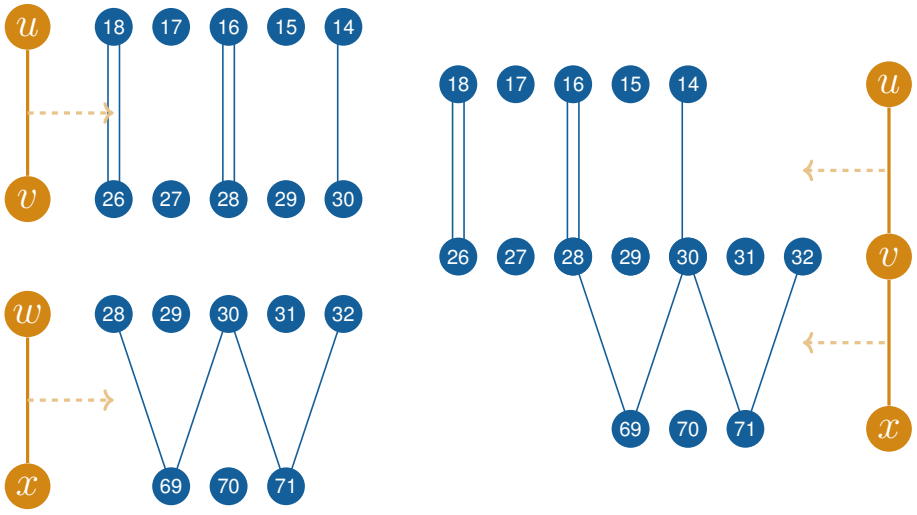


Figure 3.6: Visualization of the merging of two vertices representing strands. Merging two vertices v, w of two pairs of adjacent vertices u, v and w, x. The result consists of the merged vertex v which is now adjacent to u and x. The edge label information (hydrogen bond contacts) and the covered sequence residues are highlighted in blue. Merging takes place due to the overlap of residues 28–30 of the vertices v and w. This Figure is extracted from [7].

The next step of the algorithm is dedicated to the removal of vertices representing a strand in a non-extended region of the chain. First, we create a binary fingerprint F_B containing the size of the chain $|F_B| = |C|$ bits. Each bit is assigned the initial value 0. We set all bits $b_i \in F_B$ to 1 if the corresponding residue r_i is set as missing. Furthermore, we set all bits to 1 if they are covered by the residues of a hydrogen-bonded turn with a length of at most 4 of any class (*normal-3* and *-4*, *reverse-2*, *-3*, and *-4*). Using this fingerprint, we search for entirely blocked strands and corresponding vertices in the graph and remove these. More precisely, for each vertex $v \in V$, we calculate its sequence coverage segment s_v with $s_v.front$ as the minimum and $s.back$ as the maximum sequence position of the contacts of its incident edge. Please note that the vertex has only one incident edge as no merging of the adjacent vertex pairs has taken place yet. If all bits $b_i \in F_B$ with $s_v.front \leq i \leq s_v.back$ are marked in F_B, we remove v, its adjacent vertex w, plus the edge $e = (v, w)$ from G.

We now merge the vertices. For each vertex $v \in V$, we calculate its sequence coverage segment s_v as described before, and sort all segments in ascending order. For each pair of two consecutive segments s_j, s_{j+1}, we calculate the sequence distance $d_s(s_j, s_{j+1}) = s_{j+1}.front - s_j.back$. If the segments overlap ($d_s(s_j, s_{j+1}) \leq 0$), or if there is a gap of at most one residue ($d_s(s_j, s_{j+1}) = 1$) and the bits $b_i \in F_B$ with $s_j.back \leq i \leq s_{j+1}.front$ are not marked (0), we merge the corresponding vertices. Let $v, w \in V$ be vertices to be merged. We delete all edges $(w, x) \in E$ and add edges (v, x) to E. In other words, all neighbors of w are removed and added as neighbors to v without modifying the edge labels. We update the segment s_v of v, delete w from V, and proceed with s_v and its succeeding segment. We stop if no further merging takes place. An example is given in Figure 3.6. On the left, two pairs of adjacent vertices u, v and w, x are shown. The sequence

coverage segments are s_u = (14, 18), s_v = (26, 30), s_w = (28, 32), and s_x = (69, 71) based on the hydrogen bond contacts the edges are labeled with. The sequence distance $d_s(s_v, s_w)$ is -2. Therefore, the vertices v, w are merged. The result is depicted on the right hand side of Figure 3.6.

To obtain conformationally stable strand termini and a more consistent assignment, we shrink the sequence coverage segments of each vertex. Let s_v be the sequence coverage segment for a vertex $v \in V$. For each terminus $s_v.front$, $s_v.back$ of s_v, we calculate the number of hydrogen contacts stored at incident edges of v. In our example depicted in Figure 3.6 (right hand side), the number of contacts for $s_u.front$ is 1, and for $s_u.back$ is 2. Note that there must be at least one hydrogen bond contact that justifies each terminus. In general, we increment $s_v.front$ by 1 if the number of hydrogen contacts is less than 2. We decrement $s_v.back$ analogously. Thus, in our example, the initial $s_u = (14, 18)$ is shrunk to $s_u = (15, 18)$. After this procedure, we remove all vertices $v \in V$ whose segments s_v are not at least 2 residues long: $s_v.back - s_v.front = 0$. The remaining vertices with the shrunk ranges represent the final strands.

In each strand, we search for kinks, i.e., non-extended regions in strands. For each sequence segment of length 4 in a strand, we calculate the distance d between the Cα atoms of the first and the last residue r_i and r_{i+3}. If $d(r_i, r_{i+3}) \leq 8.5$ Å, we set a kink between residues r_{i+1} and r_{i+2}. If there are consecutive kinked sequence regions, such as $(r_i, r_{i+3}), (r_{i+1}, r_{i+4}), \ldots, (r_{i+k}, r_{i+k+3})$ with a total length $l = k + 3 + 1$, we set a kink between residues $r_{i+k/2+1}$ and $r_{i+k/2+2}$ if l is even or at residue $r_{i+\lfloor k/2 \rfloor+1}$ if l is odd. For a given strand spanning residues r_i to $r_{i+|s|}$, a kink is only reported for residues r_j with $j \in \{i + 2, \ldots, i + |s| - 1 - 2\}$. The threshold of 8.5 Å was chosen based on the four-residue Cα–Cα distance histogram for strands shown in Figure 3.7.

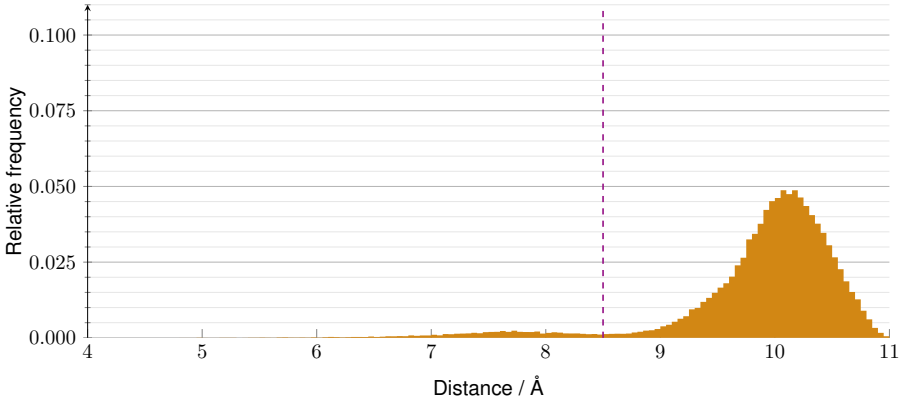


Figure 3.7: Histogram of the four-residue segment Cα–Cα distances for SCOT-assigned strands. The chosen kink distance is indicated by the dashed purple line. This Figure is extracted from [7].

Finally, each connected component of the graph represents a sheet and each vertex in such a component a strand of this sheet. All sheets and strands are reported in ascending order with respect to their sequence coverage segments by the use of a priority queue based on the sequence positions of these segments. Therefore, the first strand of a sheet is the one with the lowest value for $s.front$. This ensures an order-consistent reporting of the strands in the PDB file output.

By the use of the command line option `--split-kinked-strands`, the strands are split at the kink

positions. For a given strand s defined on residues r_i to $r_{i+|s|-1}$ and a kink in s between residues r_k and r_{k+1}, the strand is split into two strands s_1, s_2 from residues r_i to r_k and r_{k+1} to $r_{i+|s|}$. If a kink is solely defined on a residue r_k, this residue is dropped and s_1, s_2 are defined on r_i to r_{k-1} and r_{k+1} to $r_{i+|s|-1}$, respectively. Each split strand (e.g., s_1) is assigned a new registration that involves a residue within its segment.

3.3.4 Helices

We have developed five different algorithms to assign helices, namely, Combined (see Section 3.3.4.1), Kinked (see Section 3.3.4.2), Cut (see Section 3.3.4.3), Blocked (see Section 3.3.4.4), and Mixed (see Section 3.3.4.5).

The names are given with respect to their main functionality. The Combined algorithm combines or merges initial helices. The Kinked algorithm introduces the detection of kinks. The Cut algorithm splits or cuts helices with respect to four-residue segment Cα–Cα distances. The Blocked algorithm adds a blocking mechanism to the merging procedure. Finally, the Mixed algorithm assigns the class mixed to all helices for which a distinct class cannot be determined. It is also the helix assignment algorithm which is incorporated into the latest version of SCOT.

3.3.4.1 Combined

The Combined algorithm utilizes the classified turns (see Section 3.3.2) to assign right-handed α-, 3_{10}-, π-, γ-, and left-handed α-, and 3_{10}-helices. Initial core helices based on helix-class-specific turns are later combined to the final helices.

The algorithm processes each of the two helix categories (right- and left-handed) separately. Each helix class within a category is based on an individual turn type (e.g., *normal-5 1* for α-helices). All categories, classes, corresponding turn types, and parameters are shown in Table 3.3. We will explain the assignment of helices for the right-handed helices. The procedure is identical for the left-handed helices but based on the classes, turns, and parameters of that category.

The category of right-handed helices consists of α-, 3_{10}-, π-, and γ-helices. It also contains the extensions which are based on two *open* turn types, namely *open-5 1* (α), *open-4 2* (3_{10}), and *open-6 4* (π) turns. These turn types are used to extend right-handed helices with helical regions based on *open* turns that have a helical conformation with respect to the dihedral angles (see Table 6.1, appendix) similar to their hydrogen-bonded counterparts (*normal-5 1*, *normal-4 2*, and *normal-6 2* turns).

The procedure to assign right-handed helices is separated in two parts. First, the core helices for each class are determined based on the turn overlaps. Second, we combine these core helices to obtain the final helices. We will explain the determination of the core helices for the α-helices.

To determine the turn overlaps, we calculate the number of turns (*normal-5 1* for α-helices) each

Helix	Turn			Overlaps	
Right-handed	**Category**	**Length**	**Class**	**Overlaps**	**Length**
α (1)	normal	5	1	3	1
3_{10} (5)	normal	4	1	2	2
π (3)	normal	6	2	2	5
γ (4)	normal	3	1	2	2
Extension	open	5	1	4	1
	open	4	2		
	open	6	4		
Left-handed					
α (6)	normal	5	9	3	1
3_{10} (11)	normal	4	3	2	2

Table 3.3: Parameters and turn types used by the Combined algorithm. Parameters and turn types for the classification of helices and their extensions used by the Combined algorithm (see Section 3.3.4.1). See Table 6.1, appendix for the average dihedral angles, hydrogen bond energies, and four-residue segment Cα–Cα distances of the respective turn classes. The numbers in parentheses indicate the SSE class according to the PDB file format.

residue is spanned by using an integer fingerprint F_α of the size of the chain $|F_\alpha| = |C|$. Next, we search for sequence segments of consecutive sequence positions i for which every $b_i \in F_\alpha$ is at least 3 and of a sequence length of at least 1. Each such sequence segment s defines a core helix from $s.front$ to $s.back$. Each core helix also contains the sum of its corresponding turn overlaps $\sum_{i=s.front}^{s.back} b_i$.

The combination step combines the core helices of all classes of this category (α, 3_{10}, π, γ, extensions). We sort all core helices H with respect to their sequence positions. We search for subsets $H_S \subseteq H$ of transitively overlapping core helices with an overlap of at least 1 residue. In other words, for any two helices $h_s, h_t \in H_S$ there is a subset of consecutively overlapping helices in H_S for which h_s is the first and h_t is the last one or vice versa. In the words of graph theory, assume all core helices in H_S are vertices of a graph G and there is an edge between any two vertices if their corresponding helices overlap, then G is connected.

After this combination procedure, the core helices of each remaining subset H_S represent the (final) helix which is defined from the residue with the smallest to the one with the highest sequence position of the residues of all core helices $h \in H_S$. Its class is based on the sum of the lengths of the core helices of each of the four classes. The class with the highest coverage defines the final helix class. If there are multiple maxima, we calculate the number of turn overlaps among all core helices for each class. Similarly, the class with the maximum number of turn overlaps defines the final helix class. However, if there are also multiple maxima, we report the final helix for each class with maximal turn overlaps separately. In other words, if, based on the two criteria, a distinct class cannot be determined, the helix is reported for each of the maximal classes. This is due to the fact that the PDB File format does not support multiple helix class annotations within a single `HELIX` line, i.e., the sequence segment is reported several times for different classes.

Finally, all right- and left-handed helices are joined and sorted ascending with respect to their sequence positions.

3.3.4.2 Kinked

The Kinked algorithm assigns the same categories and helix classes as the Combined algorithm (see Section 3.3.4.1). The main difference is the introduction of three-layered core helices, each consisting of a core, a hull, and an extension. This separates the extension from the hydrogen bond-based core helix classes. Furthermore, it blocks the merging of overlapping helices if specific *open* turns are located at the point of overlap. The algorithm also supports the determination of kinks. All categories, classes and corresponding turn types, and parameters are given in Table 3.4.

Helix	**Turn**			**Overlaps**	
Right-handed	**Category**	**Length**	**Class**	**Overlaps**	**Length**
α (1)	normal	5	1	2	3
3_{10} (5)	normal	4	1	2	2
π (3)	normal	6	2	2	5
γ (4)	normal	3	1	2	2
Extension	open	5	1	3	1
	open	4	2		
	open	6	4		
Left-handed					
α (6)	normal	5	9	2	2
3_{10} (11)	normal	4	3	2	2
Extension	open	5	46	2	2
	open	4	15		

Table 3.4: Parameters and turn types used by the Kinked algorithm. Parameters and turn types for the classification of helices and their extensions used by the Kinked algorithm (see Section 3.3.4.2). See Table 6.1, appendix for the average dihedral angles, hydrogen bond energies, and four-residue segment Cα–Cα distances of the respective turn classes. The numbers in parentheses indicate the SSE class according to the PDB file format.

Similar to the Combined algorithm, each category is processed separately, but there are some differences in the processing of right- and left-handed helices. Therefore, we will explain the helix assignment by reference to the right-handed helices and point out the differences to the left-handed helices thereafter.

The classification of helices starts with the determination of the set of extensions E. These extensions are used to extend the hydrogen-bonded cores of the helices by geometrically similar (helical) conformations. For each residue, we determine the number of specific *open* spanning turns using an integer fingerprint F_E of the size of the chain $|F_E| = |C|$. We use *open* turns whose dihedral angles (see Table 6.1, appendix) correspond to those of the *normal* turns defining the corresponding helix. For the right-handed helices, *open-5 1* (α), *open-4 2* (3_{10}), and *open-6 4* (π)

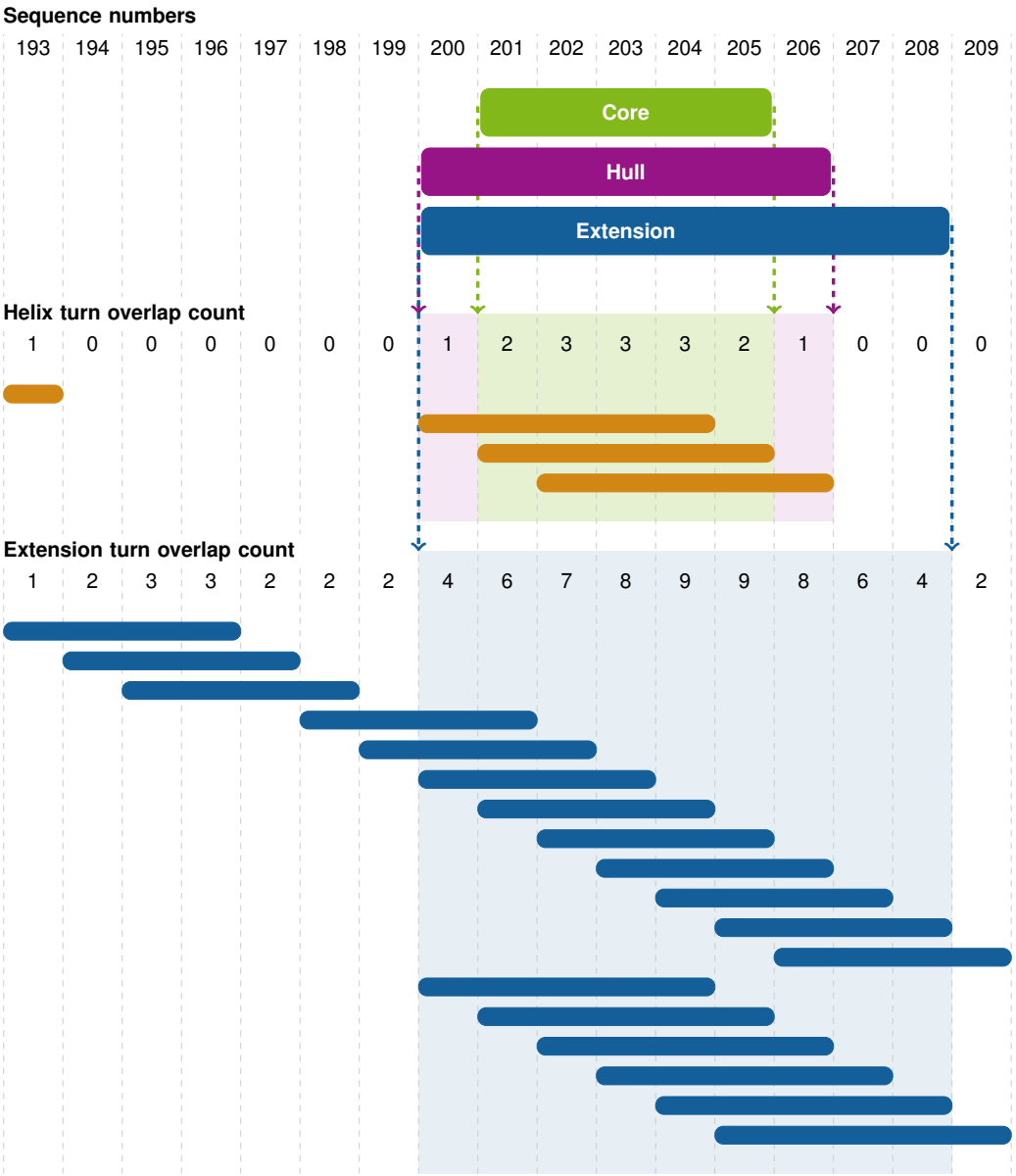


Figure 3.8: Visualization of the layers of a core helix. Visualization of the core helix layer definition based on an α-core helix at residues 193–209 of 1lg7A@pdb. The core (green) is based on three *normal-5 1* turns (highlighted in orange) which lead to a helix turn overlap of at least 2 from residue 201 to 205. The hull (purple) requires at least one *normal-5 1* turn. The extension shown in this example is based on *open-5 1* and *open-6 4* turns. The required turn overlap of at least 3 is given from residue 200 to 208. This Figure is reproduced by permission of Bioinformatics (2019) [7].

are used.

An extension $e \in E$ is a sequence segment with $1 \leq e.front \leq e.back \leq |F_E|$ of consecutive indices. The corresponding residues r_k of all integers $b_k \in |F_E|$ with $\forall k \in \{e.front, \ldots, e.back\}$ are spanned by at least 3 turns: $3 \leq b_k$. An example is given in the lower half of Figure 3.8. For residues 193–209 of 1lg7A@pdb, the aforementioned *open* turns and the corresponding excerpt of the

fingerprint are shown. The extension is defined on residues 200–208.

Next, we determine the cores and hulls of the helices and add the extensions to obtain three-layered core helices, as exemplarily shown in Figure 3.8. We determine the core helices for each class separately (e.g., α, 3_{10}). These are determined based on the turn overlaps similar to the extensions. This procedure is described for the example of an α-helix. The integer fingerprint F_α is solely based on the *normal-5 1* turns. The cores are defined as continuous segments with an overlap of at least 2 and of a sequence length of at least 3. The hull, which is based on the surrounding residues that are overlapped by at least 1 turn (see Figure 3.8), is assigned to each core. We also assign an extension $e \in E$ to the core helix if its core and e share at least one residue. Please note that an extension can be assigned to multiple core helices of different classes.

Class	Turn pos.
4	2
5	2
8	2
1	3
6	3
19	3

(a) *Open-4*

Class	Turn pos.
2	2
8	2
14	3
20	3
31	3
37	3
3	4
5	4
6	4
12	4
19	4

(b) *Open-5*

Class	Turn pos.
1	2
20	2
67	2
12	2,3
2	3
31	3
7	4
11	4
21	4
46	4,5
6	5
10	5
17	5
29	5

(c) *Open-6*

Table 3.5: *Open* turns and their relative turn sequence positions preventing the merging of helices. All *open* turns and their relative turn sequence positions whose corresponding bits $b_i \in F_B$ are marked and, thus, block the merging of two core helices in the merging procedure.

At this point, H contains all three-layered core helices. Each core helix consists of the three aforementioned layers, namely, a core, a hull, and an extension layer, plus a class (e.g., α for core helices based on *normal-5 1* turns). We now merge the core helices. To suppress the merging of overlapping core helices in certain regions of the sequence, we create a binary fingerprint F_B consisting of $|C|$ bits. For the *open* turns depicted in Table 3.5, we mark the corresponding bits in F_B. Each marked bit $b_i \in F_B$ indicates a *blocked* sequence position i. For instance, we mark b_{i+3} if there is an *open-5 20* turn starting at sequence position i. For this list of *open* turns and their positions, we analyzed all *open* turns that are within helices. All turns whose Cα–Cα distance was significantly higher than distances commonly observed in helices were selected. The turn positions were defined by visual inspection.

Next, we sort all core helices H of the current category (right-handed) with respect to their sequence positions (ascending). Let h_j be the first core helix in H. We create a sequence segment s based

on the extension $s = h_j.extension$. The following merging procedure checks pairs of consecutive helices for overlaps. The initial pair is h_j and h_{j+1}. There are four possible overlaps: cores, hulls (but no cores), extensions (but no cores and hulls), and no overlap at all. For instance, there is an overlap of cores if $h_j.core.back \geq h_{j+1}.core.front$. In such a case, we merge h_j and h_{j+1} by extending $s.back$ to $h_{j+1}.extension.back$ and proceed to check for overlaps between h_{j+1} and its successor h_{j+2}.

If only the hulls overlap, we search for sequence positions $i \in \{h_{j+1}.hull.front + 1, \ldots, h_j.hull.back - 1\}$ corresponding to marked bits $b_i \in F_B$. In other words, we search for a blocked sequence position in the overlapping sequence region excluding both hull terminus residues. If there is at least one marked sequence position, the merging is blocked. Let i be the minimum and k be the maximum sequence position with $b_i = 1$ and $b_k = 1$ in the defined overlapping region. We set $s.back$ to i and report s as the (final) helix. We create a new segment s based on h_{j+1} as initial core helix but with $s.front$ set to k instead of $h_{j+1}.extension.front$, and start a new merging procedure. Otherwise, if there are no bits marked or sequence positions blocked, we also merge the two core helices h_j and h_{j+1} by extending s as explained for overlapping cores and additionally search for kinks between the cores of h_j and h_{j+1} here. At first, we create an integer fingerprint F_K analogously to F_E, but based on the *normal* turns corresponding to the helix classes of h_j and h_{j+1}. For instance, if h_j is an α- and h_{j+1} a 3_{10}-core helix, we create F_K based on *normal-5 1* and *normal-4 2* turns. F_K covers the sequence positions from $h_j.core.back + 1$ to $h_{j+1}.core.front - 1$. To identify kinks we search for *valleys* in F_K. Each *valley* is a consecutive sequence segment s_v for which all corresponding integers $b_i \in F_K$ have equal values but the direct surrounding integers have a higher value. We define a kink for each segment s_v (*valley*) at the residue of its center, i.e., at sequence position $\lfloor \frac{s_v.front + s_v.back}{2} \rfloor$. The class of a kink is composed of two digits representing the classes of its surrounding core helices, e.g., 15 for a kink between an α- and a 3_{10}-helix core.

If only the extensions overlap, we remove or reduce the extensions between the two core helices h_j and h_{j+1} by setting $s.back$ to $h_j.hull.back$ and reporting s as a helix. We proceed with a new segment starting with the core helix h_{j+1} and setting its front to $h_{j+1}.hull.front$. If there is no overlap at all, we report s as a helix and proceed with h_{j+1} and a new segment.

We assign classes to the reported helices (segments) based on the sequence coverage and the overlaps of the helix-class defining turns, similar to the Combined algorithm described in Section 3.3.4.1.

There are two differences in the procedure for the assignment of left-handed helices. First, we do not block the merging of left-handed helices. Second, to maintain a two digit number for the class of a kink, the classes of its surrounding core helices are transformed from the left- to their right-handed counter part integers. The class of a kink between a left-handed α- and a 3_{10}-core helix is 15.

Finally, all right- and left-handed helices are joint and sorted ascending with respect to their sequence positions. All helices with a sequence length of at least 3 are reported.

3.3.4.3 Cut

Compared to the Kinked algorithm to assign helices described in the previous section (see Section 3.3.4.2), the Cut algorithm additionally supports a new helix category, namely, ribbon, consisting of PPII and (left-handed) 2.2_7-helices. It also introduces a splitting procedure for cores based on Cα–Cα distances and a Purity for the assigned helix class based on the number of turn overlaps. The merging procedure actually merges core helices instead of using a sequence segment and calculates the sequence coverage for each class by counting covered residues instead of summing up core helix lengths. All categories, classes and corresponding turn types, parameters, and the new splitting distances are shown in Table 3.6.

Helix	**Turn**			**Overlaps**		**Splitting**
Right-handed	**Category**	**Length**	**Class**	**Overlaps**	**Length**	**Distance**
α (1)	normal	5	1	2	3	> 6.25 Å
3_{10} (5)	normal	4	1	2	2	> 6.55 Å
π (3)	normal	6	2	2	5	> 7.00 Å
Extension	open	5	1	3	1	-
	open	4	2			
	open	6	4			
Left-handed						
α (6)	normal	5	9	2	2	-
3_{10} (11)	normal	4	3	2	2	-
Ribbon						
Polyproline II (10)	open	4	9	2	4	< 7.45 Å
2.2_7 (8)	normal	3	1	2	2	-

Table 3.6: Parameters and turn types used by the Cut algorithm. Parameters and turn types for the classification of helices and their extensions used by the Cut algorithm (see Section 3.3.4.3). See Table 6.1, appendix for the average dihedral angles, hydrogen bond energies, and four-residue segment Cα–Cα distances of the respective turn classes. The numbers in parentheses indicate the SSE class according to the PDB file format.

Similar to the description of the Kinked algorithm, we will also describe the procedure of the Cut algorithm according to the category of right-handed helices and point out the differences to the categories of left-handed and of ribbon helices thereafter. The category of right-handed helices consists of α-, 3_{10}-, and π-helices. The formerly included γ-helices were identified as 2.2_7-helices and moved to the category of ribbon helices. The π-helices are also processed differently. In contrast to other right-handed helices, we do not assign extensions to π-helices. Their extensions are identical to their hulls. We identify all right-handed core helices similar to the Kinked algorithm.

At this point, H contains all three-layered (right-handed) core helices. Each core helix consists of the three aforementioned layers, namely a core, a hull, and an extension layer, plus a helix class (e.g., α for core helices based on *normal-5 1* turns). We now split the core helices to obtain stable helix conformations. The splitting is based on four-residue segment Cα–Cα distances, which

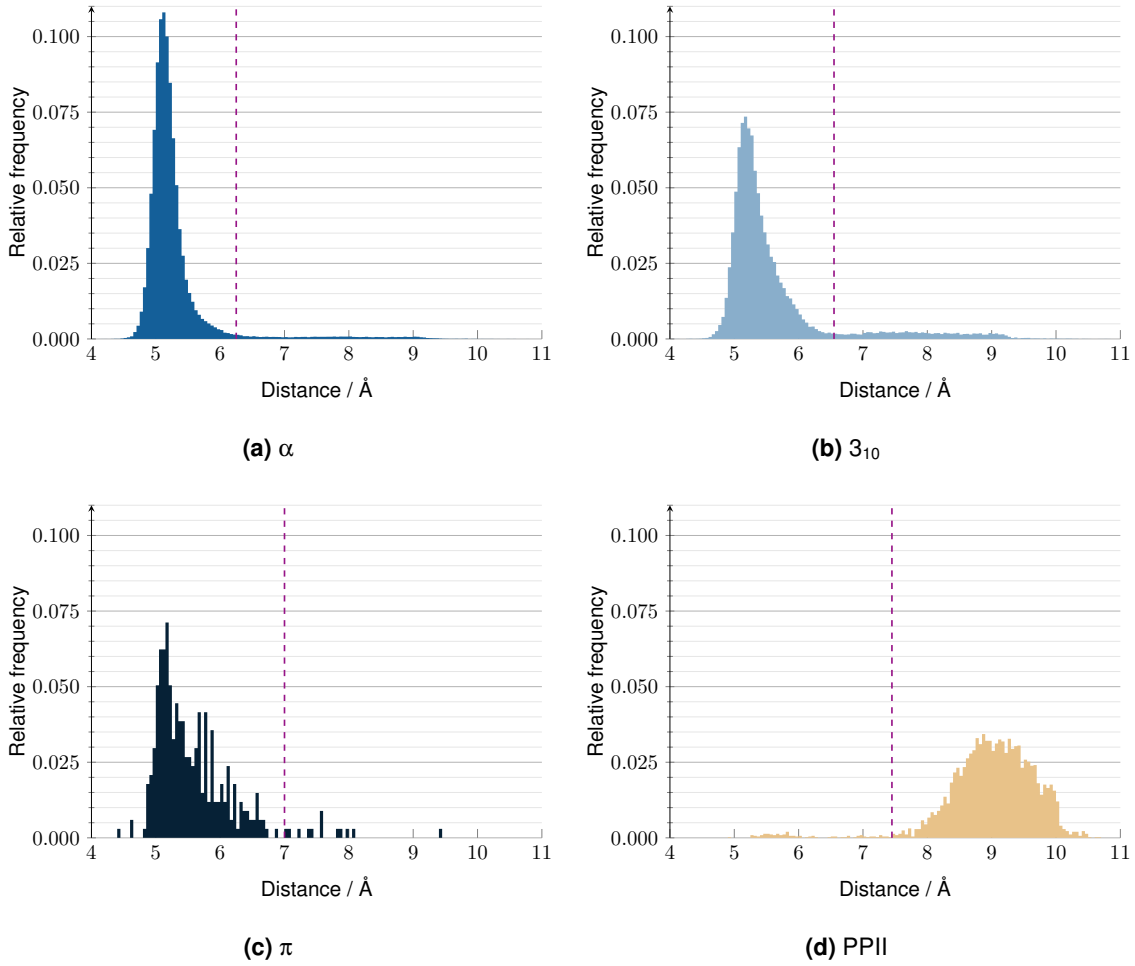


Figure 3.9: Histograms of the four-residue segment Cα–Cα distances for different helix classes. Histograms of the four-residue segment Cα–Cα distances for SCOT-assigned right-handed α- (a), 3_{10}- (b), and π-helices (c) and PPII helices (d) before the introduction of the core helix splitting procedure. The chosen splitting distance is indicated by the dashed purple line in each histogram. This Figure is extracted from [7].

were empirically derived by analyzing these distances with SCOT before implementing the splitting procedure. Figure 3.9 shows the corresponding histograms and the distances we derived for each helix class.

The spitting procedure starts with the calculation of the maximal sequence segment $s_{\max} = (\min(h.hull.front, h.extension.front), \max(h.hull.back, h.extension.back))$ for each core helix $h \in H$. For each four-residue Cα–Cα segment s within $\{s_{\max}.front - 1, \ldots, s_{\max}.back + 1\}$, we calculate the distance d between the Cα atoms of the first residue $r_{s.front}$ and the last residue $r_{s.back}$. If d is above the core helix class-specific threshold (e.g., 6.25 Å for α-helices, see Table 3.6), we split the core helix between residues $r_{s.front+1}$ and $r_{s.back-1}$. If the split is outside the core of h, we keep the split part that contains the core. We retain only those split core helices whose $s_{\max}$ has a length of at least 3. Figure 3.10 depicts an example of the splitting procedure. At the top the initial core helix h is shown. The vertical purple lines indicate sequence positions where the distance is above the threshold. For instance, the first purple line is the result of a distance above the threshold between

the Cα atoms of the residues at 194 and 197. The core helix h is split into core helices h_1, h_2, and h_3. We do not create a core helix for the segment of residues 193–195 because there is no core involved in this segment. We also drop the split core helix from residue 199 to 200 because its length is below the required minimum of 3.

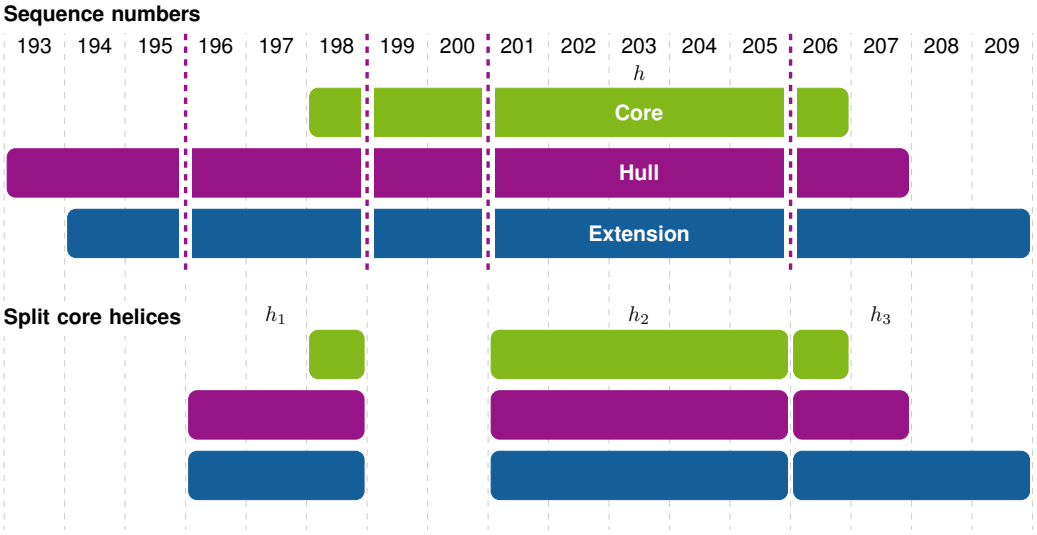


Figure 3.10: Generic example of the splitting of a core helix. Generic example of the splitting of a core helix h by the Cut algorithm (see Section 3.3.4.3). There are four splits (purple lines) based on Cα–Cα distances above the threshold between residues 194–197, 197–200, 199–202, and 204–207 leading to 5 segments. The first segment from residue 193 to 195 does not contain a part of the core and is, therefore, dropped. The segment from residue 199 to 200 is too short and is, therefore, also dropped. The segment from residue 196 to 198 contains a part of the core, is at least 3 residues long, and, thus, leads to the new core helix h_1. The same holds true for the segments 201–205 and 206–209 leading to core helices h_2 and h_3. This Figure is extracted from [7].

We determine kinks in each core helix $h \in H$. First, we create an integer fingerprint F_N analogously to the Kinked algorithm but based on the *normal* turns of helices of this category: *normal-5 1* (α), *normal-4 2* (3_{10}), and *normal-6 4* (π) turns. Next, we calculate the minimum number of overlaps for a residue that indicates a kink. For that, we determine the maximal value $o_{\max}$ (turn overlaps) of the integers $b_i \in F_N$. The maximal threshold t for a kink is set to $t = \max(0, \min(5, o_{\max}) - 2)$. Whenever the turn overlap $b_i \in F_N$ for a residue r_i is at least 2 smaller compared to the maximum overlap $o_{\max}$, r_i is involved in a kink. Each sequence segment s with $\forall i \in \{s.front, \ldots, s.back\}$ and $b_i \in F_N : b_i \geq t$ defines a kink at its center. If the length $|s|$ is even, the kink is defined between both central residues. Otherwise, the kink is defined at the central residue. Inside a core, the digits of the kink's class are equal (e.g., 11 for a kink inside an α-helix core). Otherwise, the digits may differ, e.g., 15 for a kink between an α- and a 3_{10}-helix core. Such kinks are determined during the following merging procedure and indicate separate conformationally distinct regions or classes of the final helix.

In contrast to the Kinked core merging procedure using a sequence segment to define a merged helix, we directly merge core helices here. Furthermore, we do not block core helices in the merging procedure as this functionality is covered by the splitting of core helices described above.

First, we sort the core helices in H with respect to the sequence positions of their cores (ascending). Note that H contains all core helices of this category. Then, for two consecutive core helices $h_i, h_{i+1} \in H$, we calculate the overlap of their layers. There are four possible overlaps: cores, hulls (but no cores), extensions (but no cores and hulls), and no overlap at all. For instance, if there is an overlap of cores if $h_i.core.back \geq h_{i+1}.core.front$, we merge h_i and h_{i+1} by merging their layers. The core of the merged helix h_i is set to $h_i.core = (\min(h_i.core.front, h_{i+1}.core.front), \max(h_i.core.back, h_{i+1}.core.back)$. The hull and extension are set analogously. We also merge the sets of kinks, i.e., $h_i.kinks = h_i.kinks \cup h_{i+1}.kinks$. We remove h_{i+1} from H and proceed with h_i and its new successor. If only the hulls overlap, we also merge the helices, but search for kinks between the cores of the helices $h_i.core.back$ and $h_{i+1}.core.front$ in the same manner as we searched for kinks in cores. If only the extensions overlap, we remove or reduce the extensions between the two helices in such a way, that each extension is limited to the respective hull. The extension of h_i is set to $h_i.extension.back = \min(h_i.hull.back, h_i.extension.back)$ and the extension of h_{i+1} is set to $h_{i+1}.extension.front = \max(h_{i+1}.hull.front, h_{i+1}.extension.front)$ analogously. This prevents regions within helices that are not stabilized by at least one hydrogen bond-based turn. Finally, if there is no overlap at all, we proceed with h_{i+1} and its successor. In each merging operation, we store the sequence coverage (covered residues) and turn overlaps for each class of the core helices the merged helix originates from. For instance, the reported helix of the core helix in Figure 3.8 is defined on residues 200–208.

After the merging procedure, we determine the class of each of the core helices $h \in H$ which works similar to the Combined algorithm (see Section 3.3.4.1). In contrast, however, the calculation of the sequence coverage does not sum up the lengths of the encompassed helices during the merging procedure but uses a set of covered residues. This avoids the double counting of residues in the overlaps of core helices of the same class. For instance, let $h_1, h_2 \in H$ be α and $h_3 \in H$ be 3_{10} core helices with cores defined on residues 3–12, 9–16, and 15–30 respectively. The sequence coverage based on the lengths is α: 18 and 3_{10}: 16 whereas the coverage based on a set of residues is α: 14 and 3_{10}: 16 resulting in a different final helix class assignment (3_{10} instead of α). The turn overlaps are calculated for the merged helix, to avoid the same issue. Furthermore, we add the calculation of the Purity. If the class for h can be defined based on the sequence coverage, the Purity is set to 1. Otherwise, we calculate the Purity based on the turn overlaps. We calculate the total sum o of all turn overlaps and also the sums for each class separately (e.g., o_α for α-core helices), both with respect to the maximal sequence segment $s_{\max}$ of h. The Purity is calculated by the class-specific turn overlaps divided by the total number of turn overlaps (e.g., $\frac{o_\alpha}{o}$).

Finally, all core helices $h \in H$ are reported as helices from the minimum of the hull and the extension to the N-terminus (front) and the maximum to the C-terminus (back) analogously.

The assignment of left-handed helices is identical to the one of the right-handed helices with some exceptions. First, we do not determine kinks, neither in cores nor in hulls. Second, we do not assign extensions or hulls. Both are set to be identical to the cores. Third, the Purity is based on the turn overlaps of the turns used within this category (*normal-5 9*, *normal-4 3*).

There are two helices classes in the category of ribbon helices: PPII and (left-handed) 2.2_7-helices, referred to as ribbon due to the extended character of the backbone conformation. Similar to the

category of left-handed helices, the Purity is calculated based on the turn overlaps of the turns used within this category (*open-4 9*, *normal-3 1*). We also do not search for kinks and do not assign extensions here. But we assign hulls for ribbon helices. Furthermore, we do not split the core helices of 2.2_7-helices. However, we split the PPII core helices in a similar manner compared to the fingerprint-based blocking of strands described in Section 3.3.3.3.

For each PPII core helix h, we create a binary fingerprint F_P of the length of the maximal sequence segment $s_{\max}$ as described for right-handed helices. For every segment s of length 4 within $s_{\max}$, we calculate the distance d between the Cα atoms of the first $r_{s.front}$ and the last residue $r_{s.back}$. If d is below 7.45 Å (see Figure 3.9d for the Cα–Cα distance histogram), we set the corresponding bits of the central two residues of s ($r_{s.front+1}$ and $r_{s.front+2}$) in F_P. We also set each bit $b_i \in F_P$ if the corresponding residue r_i is also involved in a strand to favor strands (hydrogen bond-stabilized) over PPII helices. Each set bit $b_i = 1$ defines a split and we keep only those parts of the helix for which all bits are not set ($b_i = 0$). Similarly to the splitting of right-handed helices, we drop all parts that do not contain a residue of the core or are not at least 3 residues long.

Finally, all right- and left-handed as well as ribbon helices are joined and sorted ascending with respect to their sequence positions. All helices with a sequence length of at least 3 are reported.

3.3.4.4 Blocked

The Blocked algorithm to assign helices adds a blocking functionality to the merging procedure, which was introduced for the Kinked (see Section 3.3.4.2) but dropped in the Cut algorithm (see Section 3.3.4.3).

We assign helices using the categories, classes and corresponding turn types, parameters, and splitting distances used by the Cut algorithm (see Table 3.6). We identically determine the core helices, but whenever we split a helix, we mark residues that become core helix termini of the split helices as blocked. In the example depicted in Figure 3.10 residues 196 and 198 of helix h_1, residues 201 and 205 of h_2, and residue 206 of h_3 are marked as blocked. These marked termini (block positions) are used during the merging procedure.

Before the core helices are merged, we delete all impure core helices. A helix is assumed to be impure if there are more turn overlaps of turns corresponding to other helix classes within this core helix than of the turns of this core helix class. In more detail, for a given core helix h, we determine the maximal sequence segment $s_{\max}$ (see Section 3.3.4.3). We calculate the turn overlaps for all classes of the category under investigation (e.g., right-handed) in the sequence segment $s_{\max}$. If the turn overlaps o for the class of h do not define a maximum, h is assumed to be impure and deleted.

The merging procedure is similar to the one of the Cut algorithm. The major difference is that when two consecutive core helices h_i and h_{i+1} overlap, we may block the merging of these helices based on the aforementioned block positions. Whenever the hulls of h_i and h_{i+1} overlap, we still merge both helices if one of the following conditions holds true. First, the extensions of h_i and

h_{i+1} are identical, i.e., defined on the same residues. Second, h_{i+1} is fully encapsulated and at least one terminus (front or back residue) is not marked as blocked. Fully encapsulated means that $h_{i+1}.extension.back \leq h_i.extension.back$. Note that $h_i.extension.front \leq h_{i+1}.extension.front$ is implicitly given due to the sorting/order and at least the front or the back of both must be different because the first condition was evaluated as false. Third, h_{i+1} is not fully encapsulated and neither $h_i.extension.back$ nor $h_{i+1}.extension.front$ are marked as blocked. The merging itself is identical to the previously introduced algorithm. If we do not merge h_i and h_{i+1} due to one of these three conditions, we remove or reduce the extensions of both helices similar to the case of overlapping extensions described for the previous algorithm. In a nutshell, the back extension of h_i is limited to the maximum of the back of the core and the hull. The front of the extension of h_{i+1} is handled analogously. However, there is still one final case which is handled differently. If the merging of h_i and h_{i+1} is blocked, we report h_{i+1} as an individual helix if it is fully encapsulated and fully blocked, i.e., both termini of h_{i+1} are block positions.

The succeeding steps, including the assignment of the classes, the filtering with respect to the required length, the processing of left-handed and ribbon helices, and the final sorting, are identical to the Cut algorithm (see Section 3.3.4.3).

3.3.4.5 Mixed

The Mixed algorithm for the assignment of helices regroups the helix classes and redefines the importance of π-helices as a (semi) separate group. In addition, the 2.2_7-helices of the category of ribbon helices were identified as left-handed 2.2_7-helices and their right-handed counterparts are introduced. However, the difference to the other algorithms is the processing of multiple maxima during the determination of a helix class. In contrast to these algorithms, we do not report the identical helix for each maximal class but assign the class *mixed* (0) and report the helix only once. Another difference is that during the merging procedure, we drop the blocking functionality which was previously introduced by the Blocked algorithm (see Section 3.3.4.4). All categories, classes and corresponding turn types, parameters, and the splitting distances are shown in Table 3.7.

The determination of the extensions E, the core helices H, and the kinks in core helices is identical to the Cut algorithm (see Section 3.3.4.3) with two exceptions for the right-handed helices. First, the extensions are based on the *open* turn counterparts with respect to the dihedral angles (see Table 6.1, appendix) of the *normal* turns that are used for the right-handed helices plus the ones for the π-helices. Even though π-helices are not part of this category, they are frequently located within right-handed helices or at their termini and, therefore, influence and may extend these helices. Second, the determination of kinks utilizes an integer fingerprint F_K based on the *normal* turns of helices of this category: *normal-5 1* (α), *normal-4 1* (3_{10}), and *normal-6 2* (π) turns. Again we include the turns for the π-helices.

The merging procedure is identical to the one of the Cut algorithm with the following two exceptions. The determination of kinks in hulls also includes the *normal-6 2* of π-helices for the creation of the fingerprint F_K. The other exception concerns the assignment of the final class to a core helix $h \in H$. We solely take the turn overlaps into consideration and do not use the sequence coverage.

Helix	Turn			Overlaps		Splitting
Right-handed	**Category**	**Length**	**Class**	**Overlaps**	**Length**	**Distance**
α (1)	*normal*	5	1	2	3	> 6.25 Å
3_{10} (5)	*normal*	4	1	2	2	> 6.55 Å
Extension	*open*	5	1	3	1	-
	open	4	2			
	open	6	4			
π						
π (3)	*normal*	6	2	2	5	> 7.00 Å
Extension	*open*	6	4	2	1	-
Left-handed						
α (6)	*normal*	5	9	2	2	-
3_{10} (11)	*normal*	4	3	2	2	-
Ribbon						
Polyproline II (10)	*open*	4	9	2	4	< 7.45 Å
Right-handed 2.2_7 (4)	*normal*	3	2	2	2	-
Left-handed 2.2_7 (8)	*normal*	3	1	2	2	-

Table 3.7: Parameters and turn types used by the Mixed algorithm. Parameters and turn types for the classification of helices and their extensions used by the Mixed algorithm (see Section 3.3.4.5). See Table 6.1, appendix for the average dihedral angles, hydrogen bond energies, and four-residue segment Cα–Cα distances of the respective turn classes. The numbers in parentheses indicate the SSE class according to the PDB file format. This Table is extracted from [7].

The Purity of a helix h is reported for all classes that contribute to the final class and not solely for the maximal classes. Furthermore, if there is a distinct maximum for the turn overlaps, we report the corresponding class (α or 3_{10}). Otherwise, we report the helix class mixed. See Section 3.3.4.3 for more details on the calculation of the Purity.

Before we report the final helices, we perform two filtering steps. First, we remove all core helices that are shorter than 3 residues. Second, we remove all kinks that are in a three-residue proximity of a helix terminus (N and C) as helix termini are more flexible which leads to high deviations from the ideal distances in helices.

The helices can also be split at the kink positions similar to the β-strands using the command line option `--split-kinked-helices`.

Finally, all core helices $h \in H$ are reported as helices from the minimum sequence position of the hull and the extension and the maximum sequence position analogously.

π-helices are assigned independent of all other helix classes. The turns and parameters for the π-helix classification are given in Table 3.7. We split π-helices as specified for α- and 3_{10}-helices using a Cα–Cα distance of 7 Å as defined based on the histogram in Figure 3.9c. In contrast to

the assignment of the right-handed helices, the extensions are solely based on *open-6 4* turns with an overlap of at least 2. The Purity and kinks are also determined based on the turn overlaps of *normal-5 1*, *normal-4 1*, and *normal-6 2* turns. No merging with helices of a different class is performed leading to helices which might overlap with helices of other classes. The merging is restricted to π-helices only.

In the category of left-handed helices, which contains the left-handed α- and 3_{10}-helices, we do not split the core helices. We also do not determine kinks here because these helices are usually too short to contain any kinks with respect to our definition. The Purity is based on the turn overlaps of the turns used within this category (*normal-5 9*, *normal-4 3*).

There are three helix classes in our category of ribbon helices: PPII helices, and right- and left-handed 2.2_7-helices. Similar to the category of left-handed helices, the Purity is calculated based on the turn overlaps of the turns used within this category (*open-4 9*, *normal-3 2*, *normal-3 1*). We also do not search for kinks, nor do we assign extensions here. Furthermore, we do not split the core helices of 2.2_7-helices. However, we split the core helices of the PPII helices in the same way as already described in detail for the Cut algorithm in Section 3.3.4.3.

Finally, all right-handed, left-handed, and ribbon helices are joint and sorted ascending with respect to their sequence positions. All helices with a sequence length of at least 3 are reported.

3.3.5 Output: PDB File Writing

For each PDB input file, we write a PDB output file containing the SCOT SSE assignment and an optional PyMOL [32] script on request using the command line option `--write-pymol`.

3.3.5.1 PDB File

The PDB output file contains all lines from the PDB input file except for the `HELIX`, `SHEET`, `TURN`, `REMARK 650`, `REMARK 700`, and `REMARK 750` lines. The assigned primary SSEs, such as helices and sheets, are provided in the PDB file format. In the `HELIX` and `TURN` lines, the serial number (columns 8–10) is reset to 1 for each chain. In the `HELIX` lines, the helix identifier (columns 12–14) is equal to the serial number. In the `SHEET` lines, we use an integer value for the representation of the sheet identifier (columns 12–14). In the `TURN` lines, the turn identifier (columns 12–14) is reset for each turn family (*normal*, *reverse*, and *open*). In addition, columns 40–66 contain the human readable turn family, length, and class. This information is also given in numeric format at the end of the `TURN` line (columns 68, 70, and 72–73). The last column contains the energy of the hydrogen bond for *normal* and *reverse* turns or the distance from the first residue's to the last residue's Cα atom for *open* turns.

We use the `REMARK` lines to provide additional information on the assigned SSEs. We use the sections `REMARK 650` for the helix, the `REMARK 700` for the sheet, and the `REMARK 750` for the turn information. Each of these remark sections starts with the SSE the section is dedicated to and

the name of the determination method (SCOT). The `REMARK 650` lines contain two separate tables. The first table provides the assigned kinks for the helices. The first columns contain the helix serial number (columns 12–15) and identifier (columns 17–20) followed by the residues defining the kink (columns 22–32 and 34–44), and the kink class (columns 46–47). The table headings are also given. The second table contains the helix class Purities. The first two columns are equal to the previous table followed by the chain identifier (column 22), the class (columns 25–26), and the Purity (columns 28–32). The `REMARK 700` lines provide the kink information for sheets which is in the same format as described for the helix kink information. As we do not assign classes to kinks in sheets, the class column is left blank. The `REMARK 750` lines contain a short description of the tailing columns in the `TURN` lines.

Our PDB file output is suitable for most visualization tools, such as PyMOL [32] or UCSF Chimera [17], because we particularly do not make use of the comment columns in `HELIX` lines.

There are two more options to modify the PDB file output. If the option `--write-hydrogens` is used, the reassigned hydrogen atoms are added to the output file. Each hydrogen atom is written in the line preceding the `ATOM` lines of the corresponding residue. We use the atom serial of the following atom for the hydrogen atom. The option `--write-sse-only` limits the content of the output file to the `HELIX`, `SHEET`, and `TURN` lines.

3.3.5.2 PyMOL Visualization Script

The optional PyMOL script visualizes the assigned helices and sheets with separate colors for different helix classes and strands (see Table 2.1 for the color coding). Furthermore, the script also highlights kinks and termini. We use the `AngleBetweenHelices` module by Holder [91] to calculate the angle between kink-separated parts of helices and strands and highlight these parts by additional vector representations of the split SSEs. We updated the script to color these vectors gray and decreased their radius to avoid any distraction from the colored SSEs.

3.4 SHAFT Reimplementation

The Relibase [92] server is a web-based system for searching and analyzing protein-ligand structures in the PDB. The standalone version as well as the publicly provided web service of the software was retired by the CCDC in 2018. The SHAFT algorithm by Koch and Cole [18] was originally implemented in the Relibase server. However, before we have developed SCOT, we chose to reimplement the SHAFT algorithm as a standalone application due to several limitations of the Relibase core structure. For instance, there is no support for OXT hydrogen bonds. In addition, hydrogen atoms at N atoms with alternate locations are not supported either. The requirement to use the Relibase itself to apply the SHAFT classification limited its usability.

Our reimplementation is based on the same implementation framework we also use for SCOT. This means that it shares the implementation scheme, its parallelization support, and especially its input

and output procedures described in Sections 3.3.1 and 3.3.5. SHAFT requires its individual ESOM files and additional files for each turn category containing the means and standard deviations of the weights (`*.msd`). These files also have to follow the naming scheme presented for SCOT (e.g., `normal-5.msd`).

Furthermore, we applied bug fixes and added some functionalities to the algorithm. Due to the utilization of the SCOT parsing procedure, we were able to add the support for OXT atoms at the C-terminus for the *reverse* turn detection similar to SCOT. We also fixed the detection of the Schellman motif during the C-cap assignment for α-helices (see Figure 3.11). For a current helix terminus at sequence position i, the original code required an *open-4 1* turn at $i - 1$ which has to be $i + 1$ to identify the Schellman motif and to extend the helix terminus. The extension was reduced from $i + 4$ to $i + 3$. These changes were done according to Aurora et al. [93].

(a) SHAFT **(b)** Aurora

Figure 3.11: Visualization of the Schellman Ccap motif. Visualization of the Schellman Ccap motif of the original SHAFT implementation by Koch and Cole [18] (a) and the reimplementation according to Aurora et al. [93] (b). The current residue r_i and the *normal-6 1* and *normal-4 1* turns constituting the Schellman motif are colored in blue, and the extension and the extended residue are colored in purple.

Our reimplementation also provides output in the PDB file format. In contrast to the Relibase output, we drop the input helix annotation (`HELIX` lines) and solely provide the SHAFT helix annotation without the use of comments. The format of the `TURN` lines was adapted to the one of the `HELIX` lines. This especially affects the columns for the residue and category/class information. We also added the energy for *normal* and *reverse*, the Cα–Cα distance for *open* turns, and the category and class in numeric format at the end of the lines (see Section 3.3.5 for the exact column indices). We write `REMARK 650` and `REMARK 750` lines solely containing the name of the determination method. The optional PyMOL output is not available for SHAFT.

3.5 Results

This section covers the comparison and evaluation of the different algorithms developed for SCOT described in Section 3.3. For a detailed comparison of SCOT to other SSAMs be referred to the results section of SNOT (see Section 4.6).

3.5.1 Accuracy of the PDB File Parsing Procedure

The parsing of PDB files is a considerable and oftentimes underrated challenge. For instance, the fact that the sequence numbering is not necessarily monotonically increasing means that for two given residues r_i and r_j one cannot say whether r_i comes first in sequence or not based on the corresponding sequence numbers. This also holds true if $i < j$. In 5k2a@pdb sequence position 208 is followed by 1001, 1046 by 1056, and 1106 by 219. In other cases, the sequence numbering contains gaps for which no missing residues are defined, such as between sequence positions 205 and 209 in 3sd9@pdb. Some of these issues are motivated by assigning identical sequence numbers to residues with similar functions among proteins of different species.

We validated our PDB file parsing procedure with the help of the 2018 copy of the PDB dataset. We compared the `SEQRES` sequence information and, especially, the residue order of the input PDB file to the sequence order of the parsed protein structure. Out of $385{,}143$ separate protein chains, we found mismatches in $1{,}319$ chains leading to an error rate of $0.34\,\%$. This number of mismatches also includes all (format and logical) errors present in the PDB input files themselves, such as wrong `SEQRES` entries (e.g., 3tdn@pdb), which lead to a differently parsed sequence order. Thus, on datasets consisting of files without such issues the parsing rate of error is even significantly lower.

3.5.2 Turns

3.5.2.1 Choosing a Hydrogen Bond Criterion

Hydrogen bond-based turns play a major role as their overlaps define (core) helices, helix kinks, a helix' classification and Purity, the seeds for initial strands, and the blocking of merging two strands. Thus, hydrogen-bonded turns are the fundamental elements of our SSE assignment. However, their identification relies on a hydrogen-bond criterion as the input data does not cover this information. Similar to the assignment of SSEs an ideal criterion to detect the presence of a hydrogen bond does not exist.

We analyzed four different hydrogen bond criteria, namely, Kabsch and Sander [22], Mayo et al. [21], Dahiyat et al. [89], and STRIDE [51], to find the one most suitable to our needs. We chose the criterion by Dahiyat et al. and performed an exhaustive test case evaluation. Please be referred to the doctoral thesis by Christiane Ehrt [90] for the details. The final parameter optimization was based on our classification of strands. The parameter values were chosen to maximize the consistency of assigned strands.

3.5.2.2 The Special Role of Reverse Turns

The *reverse* turns play a minor or indirect role in our classification of primary SSEs. They are no key element of helices of any class nor of β-sheets or strands, respectively. In addition, they are not

used individually but as part of the group of hydrogen-bonded turns. More precisely, they are solely used in the classification of β-sheets for the identification of seeds by the Stranded algorithm and in the merging procedure of all β-sheet assignment algorithms to inhibit the merging of individual strands.

To analyze their role we assigned helices with the Mixed (see Section 3.3.4.5) and β-sheets with the Queued algorithm (see Section 3.3.3.3) to the proteins of the X-ray representatives dataset. We searched for all turns and their interactions (none, N-terminal overlap, included, C-terminal overlap) with helices and strands. For N- and C-terminal overlaps, we analyzed the number of turns with respect to the size of the overlap, i.e., the number of overlapping residues (see Figure 3.12 for a generic example of these overlaps).

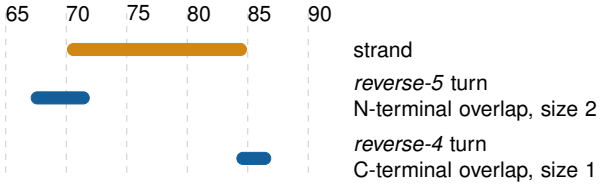


Figure 3.12: Generic example of an N-terminal and a C-terminal overlap between two *reverse* turns and a strand. The overlap sizes, i.e., number of overlapping residues, are also given.

A total of 9,014 *reverse* turns were classified for this dataset. 5,863 of these turns had overlaps given in Table 3.8. *Reverse* turns play a minor role for helices of any class or category. Nonetheless, they are essential for the termination of strands. 7,969 (95.68 %) of a total of 8,329 *reverse* turn overlaps reported for this dataset are located at the N- or C-terminus of a strand with an overlap of 1 residue.

SSE overlap	Overlapping residues	Helices	Strands
N-terminus	1	73	4,138
	2	7	4
	3	4	0
	5	130	0
Included	5	0	5
C-terminus	1	18	3,831
	2	2	14
	3	0	3
Total		234	7,995

Table 3.8: The number of *reverse* turn overlaps. The number of *reverse* turn overlaps with helices assigned by the Mixed (see Section 3.3.4.5) and strands assigned by the Queued algorithm (see Section 3.3.3.3) for the X-ray representatives dataset.

Having said that, we investigated whether the *reverse* turns overlapping strands are predominantly located between strands to form the sharp turns connecting two strands of a β-sheet and leading to the chain reversal. We were especially encouraged by the fact that the number of turn overlaps (7,995) is higher compared to the number of overlapping turns (5,863). Thus, there is a high number

of turns overlapping multiple SSEs, e.g., two neighboring strands. We searched for non-strand regions between strands of a maximum size. For small disordered sequence regions of sizes from 2 to at most 8 residues, we determined the number of *reverse* turns sharing at least one of the non-strand residues. Table 3.9 shows the results.

Region size	*Reverse-2*	*Reverse-3*	*Reverse-4*	*Reverse-5*	*Reverse-6*	Total	Acc.
2	0	3	2,447	6	0	2,456	2,456
3	0	75	226	249	8	555	3,011
4	0	1	462	442	550	1,455	4,466
5	0	1	84	1,286	399	1,770	6,236
6	0	3	41	15	388	447	6,683
7	2	1	34	11	16	64	6,747
8	0	7	40	29	6	82	6,829

Table 3.9: The number of *reverse* turns located in non-strand sequence regions of different sizes separating two strands. Column Acc. contains the accumulated totals.

6,683 (74.14 %) out of 9,014 *reverse* turns reported for this dataset are located in non-strand sequence regions of at most 6 residues. As soon as the size of these regions falls below the length of a *reverse* turn under investigation, its number of occurrences decreases drastically. Given the fact that a high number of turns overlap more than one SSE, we determined the number of turns that connect strands. I.e., we searched for *reverse* turns whose hydrogen bond connects the terminal residues of two sequence neighboring strands. We discovered 2,447 *reverse-4*, 249 *reverse-5*, and 547 *reverse-6* turns that connect the terminal residues of strands. In other words, striking 38.07 % of all reported *reverse* turns for this dataset connect two strands.

These findings clearly indicate the importance and relevance of *reverse* turns to form β-sheets which are in compliance with findings by Street et al. [94].

3.5.3 Sheets

3.5.3.1 Take your Seeds

The initial idea to classify β-sheets was to search sequence regions of extended conformations based on four-residue segment Cα–Cα distances. The fact that some SSAMs (e.g., MKDSSP, a reimplementation of the DSSP algorithm, see Section 4.4.3) classify strands of less than 4 residues of length – in some cases even single residue strands are assigned – we followed a different approach to support such lengths.

We classified the X-ray representatives dataset with MKDSSP (strands), SCOT (turns), and SCOT using the DSSP hydrogen bond criterion for comparison. We created seed fingerprints for SCOT and SCOT with the DSSP hydrogen bond criterion based on their hydrogen bond-based turns in two variants. The first variant uses all residues of a turn whereas the other variant corresponds to the one of the Stranded algorithm, i.e., using all but the terminal residues of a turn.

Table 3.10 shows the usage of seeds for both SCOT and both turn usage variants. Using all residues of a turn to mark their sequence positions as non-extended or non-seed regions, misses 195 strands classified by MKDSSP. However, if all but the terminal residues of turns are taken into account, only 2 strands are missed. The comparison of the two SCOT variants underlines the benefits of the Dahiyat over the DSSP hydrogen bond criterion here. Although a significant higher number of seeds are discovered using the DSSP criterion, a fewer number of strands could be identified using them.

	SCOT	SCOT (DSSP)
Seeds	46,746	54,701
Used seeds	29,458	31,143
Unused seeds	17,388	23,857
Missed strands	195	5,305
Miss ratio	0.56 %	15.19 %

(a) All residues

	SCOT	SCOT (DSSP)
Seeds	65,668	96,201
Used seeds	29,763	44,917
Unused seeds	35,905	51,284
Missed strands	2	450
Miss ratio	0.01 %	1.29 %

(b) No terminal residues

Table 3.10: Number of seeds and their usage identified by the Stranded algorithm using the Dahiyat or the DSSP hydrogen bond criterion. The strands were classified by MKDSSP. Table (a) uses all residues whereas (b) uses all but the terminal residues for the creation of seeds. The latter one corresponds to the version used by the Stranded algorithm.

	SCOT	SCOT (DSSP)
Seeds	31	55
Used seeds	24	34
Unused seeds	7	21
Strands	31	29
Missed strands	0	2

Table 3.11: Detailed analysis of used and unused seeds for protein 1nszA@pdb for SCOT and SCOT (DSSP) with respect to 31 classified strands by MKDSSP.

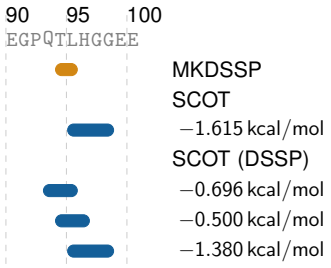


Figure 3.13: Hydrogen bond-based turns and energies for a strand assigned by MKDSSP in 1nszA@pdb which is missed by SCOT (DSSP) in contrast to SCOT.

The strands missed by SCOT using the DSSP hydrogen bond criterion are of average length 2.6 which is comparatively short. The maximum length for a missed strand is 7 in 2dplA@pdb, residues 48–54. Table 3.11 provides the detailed seed usage for 1nszA@pdb for both, SCOT and SCOT (DSSP). Although almost twice as much seeds are generated by SCOT (DSSP) it misses two of the strands classified by MKDSSP. Figure 3.13 shows the hydrogen bond-based turns that impedes the creation of a seed by SCOT (DSSP). The first two turns are based on hydrogen bonds of very high energies close to the threshold of $-0.5\,\mathrm{kcal/mol}$. The strongest and third turn is also detected by SCOT using the Dahiyat hydrogen bond criterion.

The comparison of the two SCOT variants underlines the benefits of the Dahiyat over the DSSP hydrogen bond criterion here. Although a significant higher number of seeds are discovered using the DSSP criterion, a fewer number of strands could be identified using them. Figure 3.14

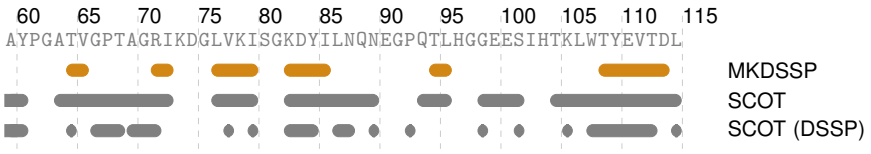


Figure 3.14: Excerpt of the strands classified by MKDSSP and the seeds generated by SCOT and SCOT (DSSP) for 1nszA@pdb.

exemplarily visualizes that the seeds created using the DSSP hydrogen bond criterion are shorter and widely scattered.

Summing up, in $1.29\,\%$ of the investigated cases the DSSP hydrogen bond criterion led to the classification of turns and strands at the same spots in sequence which is not the case for the Dahiyat criterion.

3.5.3.2 The Progress in the Assignment of β-Sheets

The main step forward from the Stranded (see Section 3.3.3.1) to the Linked (see Section 3.3.3.2) algorithm is the transition from trees to graphs for the internal representation of β-sheets. However, the notation of β-sheets according to the PDB file format immediately suggests the use of trees as the data structure to represent the combination of strands to β-sheets. Although this tree-like notation is also used for loop structures, such as β-barrels (see Figure 3.4 for the visualization of a β-barrel), the information on loops or the ability to model loop structures during the assignment process, can be substantial. During the shrinking procedure, the sequence length of a strand may be decreased below the required minimum. Assume a β-barrel with at least one strand that is not a root nor a leaf and which becomes deleted during the shrinking procedure. In such a case, the tree is split into two separate trees representing separate β-sheets because each vertex is reachable only via one distinct path. In graphs, vertices connected in a circle are reachable by at least two distinct paths. This maintains the coherence of the represented β-sheet in the deletion of one vertex of this circle. Figure 3.15 illustrates an example of the consequences of deleting a vertex in a tree representing a β-barrel. The dashed line can only be modeled using graphs and maintains the coherence after the deletion of the vertex v_4.

The next progress in the assignment of β-sheets from the Linked to the Queued algorithm (see Section 3.3.3.3) covers multiple aspects. Unlike in the previous algorithms, we inspect hydrogen bonds solely in donor-acceptor direction as for each $h^{+}_{i,j}$ the opposite direction $h^{-}_{j,i}$ is implicitly given. This has no effect on the classification outcome (see Figure 3.16 and Table 3.12) but reduces the complexity of the implementation to a considerable extent.

Algorithm	Stranded	Linked	Queued
β	19,278	32,921	32,818

Table 3.12: The number of strands assigned by the Stranded, Linked, and Queued algorithm.

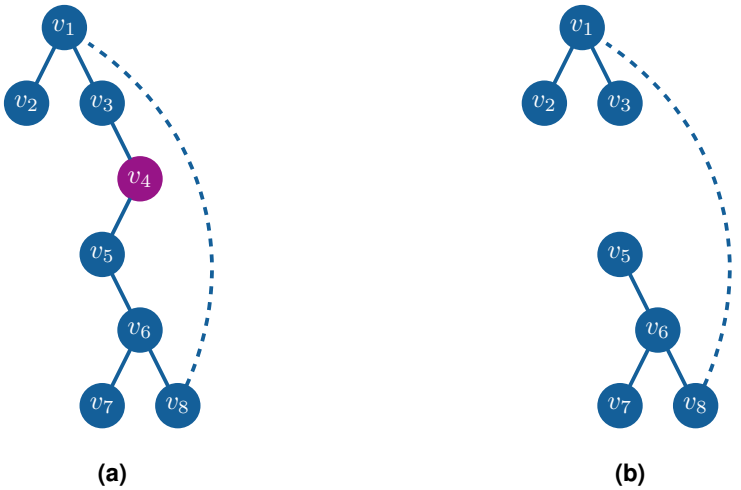


Figure 3.15: Deletion of a vertex in a tree representing a β-barrel. The deletion of vertex v_4 in a tree representing a β-barrel (a) leads to two separate trees (b). The dashed line indicates the closing of the loop which cannot be modeled by trees but by graphs.

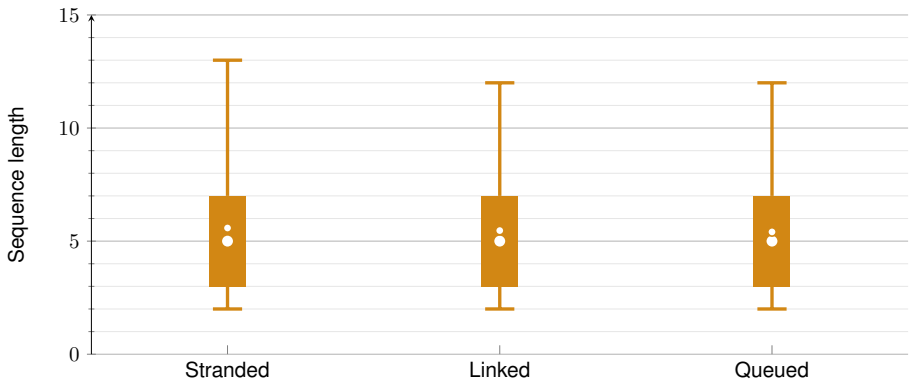


Figure 3.16: Boxplots showing the sequence lengths of strands assigned by our algorithms. The median is indicated by a big and the mean by a small white dot. Outliers were omitted in favor of a concise visualization.

In contrast to the Stranded and Linked algorithm, the Queued algorithm takes all hydrogen bonds into account and does not limit them to seed sequence regions. The motivation to use seeds is to search for strands in extended regions. However, strands in non-extended regions are due to this non-extended conformation usually very short. Thus, the benefits of using seeds are covered by the filtering of strands with respect to required sequence length. Another difference to the Linked algorithm is that there is no separation between the classified sequence segments and the corresponding hydrogen bonds. This also reduces the complexity of the implementation and avoids the storage of redundant information. The introduction of the queue tackles two important issues. First, it ensures an ordering of the vertices that is runtime independent. In dependence on the chosen C++ data structure for the storage of vertices, the order may vary depending on their memory addresses in each run. Second, it ensures that a vertex is only processed once and loops can be terminated. Both aspects are important when reporting the final β-sheets.

Finally, the importance of kinks assigned by the Queued algorithm and the benefits of the algorithm in general in comparison to other SSAMs is discussed in the chapter of SNOT (see Chapter 4).

3.5.4 Helices

3.5.4.1 The Progress in the Assignment of Helices

Similar to the progress of the algorithms to assign β-sheets, the first major step forward covers the data structure utilized for the storing of the elements in question, here helices. Instead of using sequence segments that indicate a classified helix, the Kinked (see Section 3.3.4.2) algorithm represents each helix via core helices consisting of three layers, namely, a core, a hull, and an extension. Each layer fulfills a unique task which especially comes into play during the merging procedure of the Kinked and the following algorithms (Cut, Blocked, and Mixed, see Sections 3.3.4.3 to 3.3.4.5). These layers also allow the search for kinks in still hydrogen bond-covered but – compared to cores – more flexible regions. However, in the merging procedure the final (merged) helix is still stored as a sequence segment instead of using a core helix.

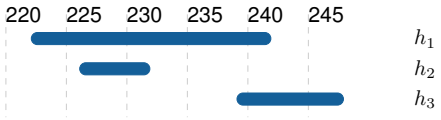


Figure 3.17: A simplified and generic example indicating the problem of the merging procedure of the Kinked algorithm. Once the core helices h_1 and h_2 are merged the merging procedure stops because there is no overlap of h_2 and h_3. The overlap of h_1 and h_3 is not taken into consideration.

Assume h_1, h_2, and h_3 are core helices as shown in Figure 3.17. After h_1 and h_2 were merged, h_2 and h_3 are examined for an overlap. Thus, the merging stops although there is an overlap of h_1 and h_3. This issue is solved by explicitly merging core helices in the Cut algorithm. Another difference to the Combined (see Section 3.3.4.1) algorithm is the use of the blocking fingerprint during the merging procedure. The major differences in the following algorithms cover the blocking, splitting, and merging of core helices, which will be discussed with respect to right-handed helices in the following section.

3.5.4.2 The Role of Right-Handed Helices in General and π-Helices in Particular

We classified the X-ray representatives dataset with the five algorithms to show the effect of their blocking, splitting, and merging procedures. Table 3.13 in combination with Figure 3.18a clearly shows the effect of the blocking procedure for α-helices. The Kinked algorithm assigns more but shorter α-helices. This trend is adhered by the following algorithms. The Cut algorithm introduces the splitting of core helices in advance to its merging procedure. It also increases the number of classified helices (except π-helices) and decreases the average length of these helices. The Blocked algorithm preserves the information of splits at blocked core helix termini which also play

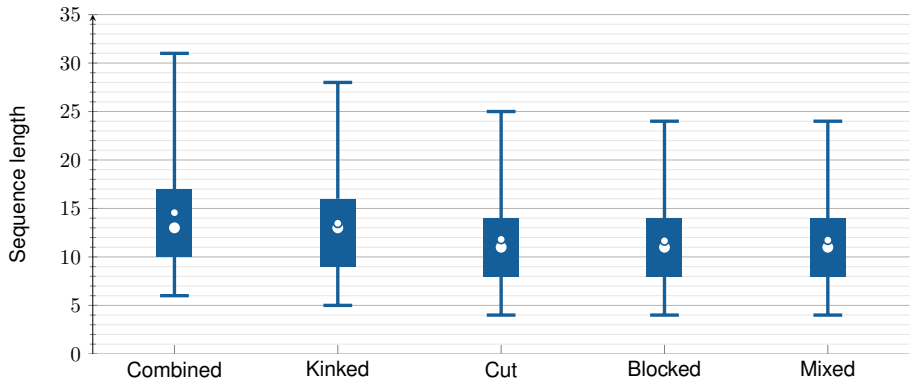
a role during the merging of core helices. Finally, the Mixed algorithm drops this information again. Furthermore, π-helices become a distinct category and are not merged with other helix classes. This results in the significant increase of the number of assigned π-helices from 135 to 572.

Helix	Combined	Kinked	Cut	Blocked	Mixed
Right-handed					
α (1)	20,125	23,713	25,447	25,734	25,588
3_{10} (5)	8,144	7,789	9,439	9,706	9,628
π (3)	82	83	61	135	572
Mixed (0)	(617)	(20)	n.s.	n.s.	147
Left-handed					
α (6)	0	4	4	4	4
3_{10} (11)	0	108	108	108	108
Ribbon					
PPII (10)	n.s.	n.s.	2,754	2,754	2,754

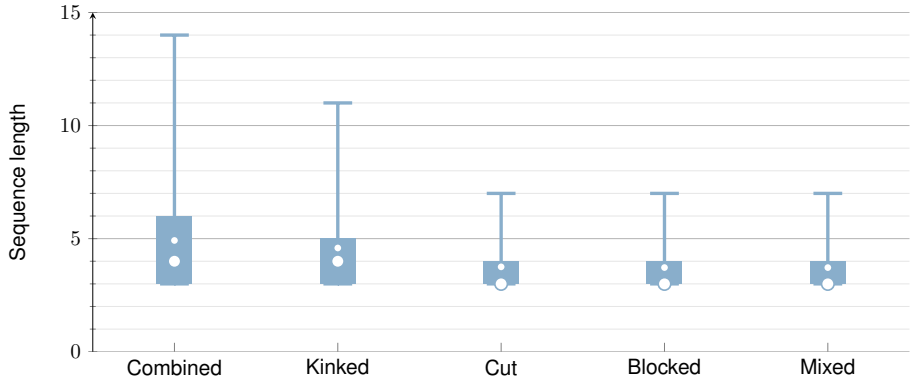
Table 3.13: The number of helices assigned by the Combined, Kinked, Cut, Blocked, and Mixed algorithm. n.s. indicates that the respective class is not supported by the algorithm. The number of right-handed mixed helices for the Combined and Kinked algorithm are due to an incomplete implementation of the final class assignment.

The overall trend in the number of assigned helices in combination with their sequence lengths reveal the challenge of differentiating between the right-handed helix classes. From the Kinked algorithm to the Mixed, the different blocking and splitting features resulted in the observation that π-helices influence right-handed α- and 3_{10}-helices but are distinct enough to form an individual category. This is also underlined by the fact that the Mixed algorithm assigns right-handed helices with the most stable sequence lengths (see Figure 3.18 for sequence length boxplots). For α- and 3_{10}-helices, there are no significant changes in the number of assigned helices and their sequence lengths for the Cut, Blocked, and Mixed algorithm. Though, we find major differences for all algorithms for the π-helices which highlight their individual role. This special role of the π-helices is also discussed in Section 4.6.

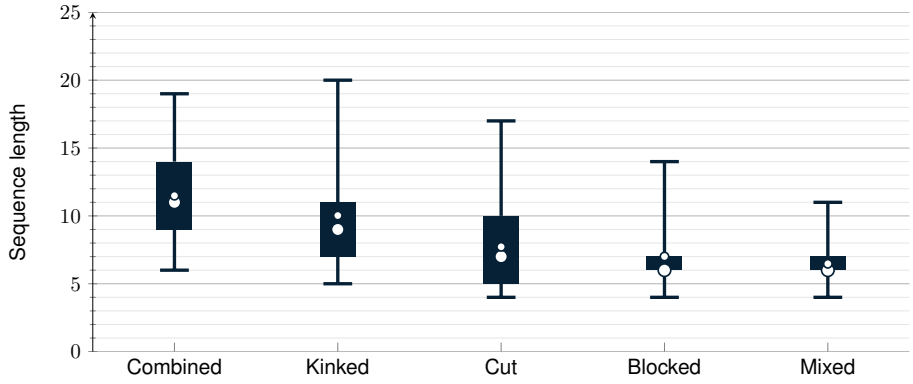
The convergence of α- and 3_{10}-helices with respect to their numbers and sequence lengths seems to suggest that their assignment by the Cut algorithm is already at a final stage of development. However, the Mixed algorithm solely uses the turn overlaps as the only criterion to define the final class for a helix. The number of assigned mixed helices indicates that this parameter is not distinct in all situations. Other SSAMs, such as SHAFT, use a class hierarchy (α, 3_{10}, π) to solve the class assignment challenge in ambiguous cases. However, the small number of 147 mixed helices among a total of 38,801 ($\approx 0.4\,\%$) proves that the turn overlaps are nevertheless suitable to determine the class of a helix in general. Especially with respect to the application of the SSE assignment by SCOT in protein structure comparison (see Chapter 5), the mixed helix class offers its usage as a template class that can be matched to α- and 3_{10}-helices. An example of a mixed helix including the corresponding turn overlaps is given in Section 4.6.



(a) α



(b) 3_{10}



(c) π

Figure 3.18: Boxplots showing the distribution of the sequence lengths of right-handed helices assigned by our algorithms. The median is indicated by a big and the mean by a small white dot. Outliers were omitted in favor of a concise visualization.

3.5.4.3 Helix Kinks

The detection of kinks in helices is almost as ambitious as the detection of helices themselves. We introduced two separate procedures to define kinks in helices. The procedure implemented in the

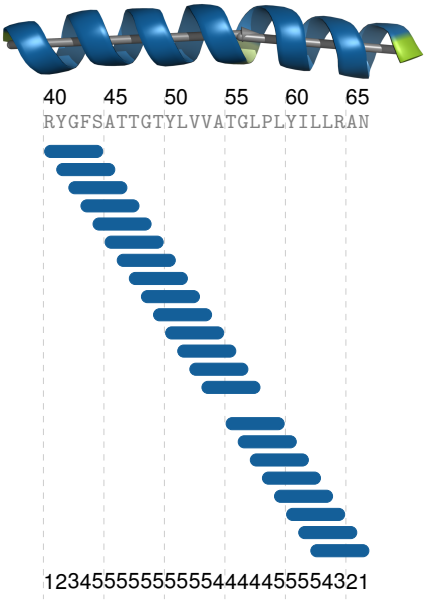


Figure 3.19: Turn overlaps (bottom) for a SCOT-assigned α-helix in 3b9w@pdb. The kink detection based on an turn overlap minimum (Kinked algorithm) assigns a kink at sequence position 56 (highlighted in green) whereas the kink detection based on the difference to the maximum turn overlaps (Cut, Blocked, and Mixed algorithms) does not assign this kink. The BDA between the vectors is 18.72°.

Kinked algorithm searches for a minimum in the turn overlaps whereas the procedure of the Cut and all following algorithms require a certain difference in the turn overlaps to the maximum turn overlaps. It is obvious that the first kink detection procedure detects all kinks of the other procedure and more as it is less restrictive. The visual inspection revealed that this fact leads to kinks that are located in bent but not kinked regions. According to Kumar and Bansal [85] the maximum BDA is above 30° in a kinked, between 20° and 30° in a bent, and below 20° in a linear region.

Figure 3.19 depicts a kink in 3b9w@pdb that is identified only by the first (minimum) kink detection procedure. Only one turn is missing in the otherwise perfect *normal-5 1* turn stacking for the presented α-helix. The turn overlaps at the bottom of the figure show minimum overlaps from residues 54 to 58. Thus, the kink procedure based on the minimum detects a kink at residue 56 which is highlighted in green in the helix at the top. However, the missing turn and the resulting minimum do not lead to a significant change in the helical geometry. This is also reflected by a low BDA of 18.72° for the vectors and a maximum BDA of 21.73° for the helix, which is typical for a bent helix according to the presented scheme by Kumar and Bansal.

This difference in the sensitivity of the two kink detection algorithms is also reflected by the boxplots shown in Figure 3.20. These represent the BDAs at kink residues for both detection procedures and for all helices classified by the Mixed algorithm in the X-ray representatives dataset. The BDAs were calculated as described in Section 4.3.2.1. While the procedure based on the minimum still covers most of the BDAs present at residues of helices in general, a clear distinction exists for the procedure based on the maximum difference. It is worth mentioning that the BDAs of kink residues were not excluded for the boxplot of BDAs in helices. The BDA is already considerably low due to

the Cα–Cα splitting.

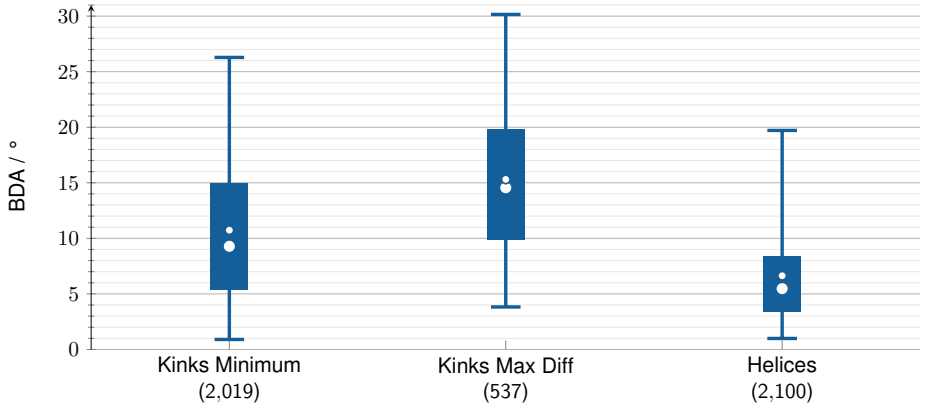


Figure 3.20: Boxplots of the BDAs at kink residues. Boxplots of the BDAs at kink residues detected by the two different procedures and at residues of right-handed helices classified by the Mixed algorithm in the X-ray representatives dataset. The calculation of the BDA is described in Section 4.3.2.1. The numbers in parentheses show the number of detected kinks or classified helices. Outliers were omitted in favor of a concise visualization.

3.5.5 Runtime and Memory Consumption

The major drawback of SCOT compared to other SSAMs is its runtime and memory consumption. The comparatively large ESOM files require approximately 700 MB of disc space whereas the ones of SHAFT require 12 MB. The information stored therein is kept in memory during the classification which leads to a memory consumption of approximately 1 GB for serial execution and up to 2.5 GB for parallel execution with 30 threads. We compared the runtime of the SCOT and the SHAFT implementation for the X-ray representatives dataset using 30 threads. The initialization including the parsing of the ESOM files took 2 s for SHAFT and 86 s for SCOT. Note that the initialization is always done in a serial manner. The total runtime for the classification of this dataset was 5 min 27 s for SHAFT and approximately 4 h 28 min 17 s for SCOT.

3.6 Discussion

We introduce SCOT as a new SSAM which assigns numerous SSEs classes which are commonly observed in protein structures. Furthermore, it enables investigations on rarely occurring SSEs. It provides necessary information about helices and strands, reports irregularities therein (kinks and Purity values), and offers the assignment of distinct sets of turn conformations which are classified according to the underlying dihedral angles. This information is provided in the established and widely supported PDB file format.

SCOT, which was inspired by SHAFT, shares a distinct set of similarities to the SSAM of Koch and Cole. Both classify the same turn categories and lengths (e.g., *normal-5*), use trained ESOMs

based on clustered dihedral angle ranges, and assign helices on the basis of turn overlaps. In spite of that, the focus of each method differs as do the assignment of and the view on SSEs. First and foremost the detection of turns is based on a different and more sophisticated hydrogen bond criterion that led to more meaningful hydrogen bonds as verified in an exhaustive test case evaluation (see doctoral thesis by Christiane Ehrt [90]). It also led to fewer hydrogen bonds as the criterion is more restrictive. The analysis of the seeds usage in Section 3.5.3.1 underlines this finding. In addition, the utilization of the DSSP hydrogen bond criterion requires SHAFT to remove weak double hydrogen bonds during the classification of turns, which is not necessary using the criterion by Dahiyat et al. with our settings.

The next major difference covers the preparation of the turn dihedral angles for the clustering. We address the challenge to use an Euclidean distance measure in angular space by our Jigsaw transformation. SHAFT uses an ω-transformation for all ω angles. Values for this angle in turns usually vary around $0°$ and $\pm 180°$ and the transformation, therefore, shifts their values by adding $90°$ to address the mentioned challenge. Next, SHAFT uses a z-transformation on all dihedral angles to normalize their values. This transformation requires the mean and the standard deviation of each angle. Their calculation is also inhibited by the same challenge as their determination by SHAFT is based on an Euclidean distance measure. Although there are approaches published with respect to the angular space [95, 96], we renounced to do a normalization due to the following reason. The typical values for each dihedral angle (π, ψ, ω) differ but φ angles are solely compared to φ angles. The same applies to ψ and ω angle values. Thus, a normalization to cope with different value ranges is not required.

Neglecting the fact that modern hardware can process larger ESOMs, the clustering itself is more elaborated compared to SHAFT. The number of neurons without a class assignment is much higher for SHAFT which especially pertains the *open* turns. For instance, the class mask for *open-5* turns of SHAFT contains $5{,}619$ of $20{,}184$ ($27.8\,\%$) neurons without a class whereas there are only $15{,}054$ out of $118{,}020$ ($12.8\,\%$) for SCOT. This is of special relevance as such turns have no distinct conformation but are used for the extension of SHAFT α-helices. In contrast, SCOT utilizes the *open* turn counterparts of the corresponding *normal* turns with respect to their dihedral angles for the extension of helices. SHAFT extends helices by *normal* turn overlaps (α, 3_{10}, π, and γ) and N- and C-cap motifs (α and 3_{10} only). For the latter, numerous turns of different categories, lengths, and classes are used to detect different capping motifs.

A further important difference is the splitting of helices based on Cα–Cα distances resulting in low BDAs inside helices. This was shown with respect to kinks and will also be further evaluated in Section 4.6.3.1.

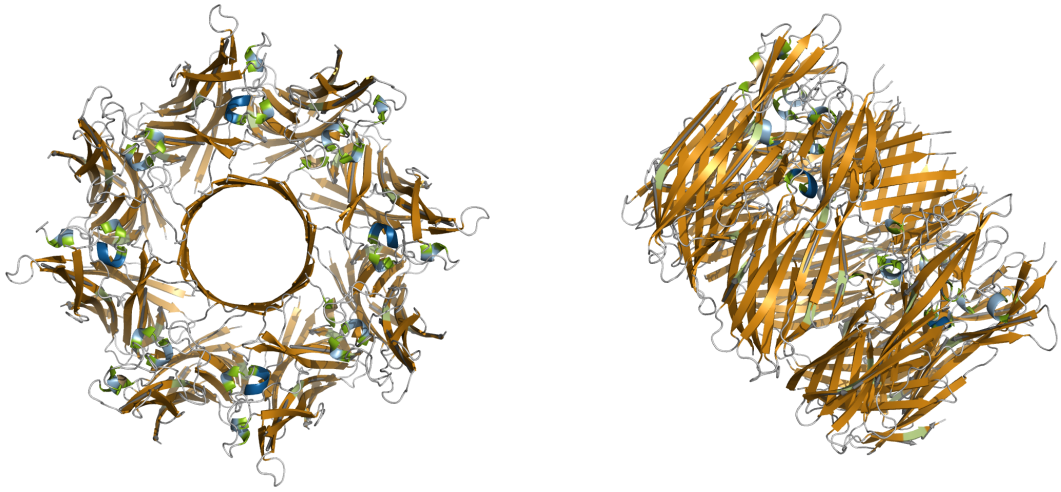
The last important detail that separates SCOT from SHAFT is the merging procedure. SCOT detects all helix classes separately and merges only overlapping (right-handed) α- and 3_{10}-helices. More important, the final class is based on the underlying turn overlaps of all turns corresponding to right-handed helices (and not solely on the turns defining the helices to be merged). SHAFT uses a hierarchical classification of helices (α, 3_{10}, π, γ), removes included helices and merges the remaining (right-handed) α-, 3_{10}-, and π-helices. The class of a helix after each pairwise merging step complies to the class of the longer of the two helices with respect to sequence length.

In addition to these fundamental differences in the two underlying methodologies in assigning turns and helices, there are multiple functionalities that set SCOT further apart. SCOT additionally supports the classification of right-handed mixed, left-handed α- and 3_{10}-, left- and right-handed 2.2_7-, and PPII helices. It provides more in-depth information about the classified helices and their irregularities by the Purity values and the kinks. In addition, it supports the classification of β-sheets also including a kink detection. Both SSEs, helices and strands, can be split based on kinks to increase its number of applications. Finally, SCOT enables a more in-depth analysis of the assigned SSEs by the optional PyMOL scripts.

The remaining questions regarding the differences in the assigned SSEs by SCOT and SHAFT will be answered in Section 4.6 after the introduction of our SSE/SSAM evaluation tool SNOT.

There is potential for optimization that affects the implementation. The ESOMs in the classification of turns contain millions of neurons. The class of a turn is calculated by determining the closest neuron with respect to a distance function.

The current implementation to determine the class of a turn requires to calculate the distance of the turn's feature vector to the feature vectors of all neurons. Approximative or heuristic algorithms are no alternative as the assignment of classes to turns has to be deterministic, i.e., always assigns the same class to the same turn. Thus, an optimization with respect to the data structure instead of an optimized distance algorithm seems most promising. Nevertheless, the practical runtimes are still acceptable on modern workstations. Therefore, we concentrated on the methodology in assigning SSEs instead.



(a) **(b)**

Figure 3.21: Visualization of 4p1x@pdb with SCOT assigned SSEs from two different perspectives. The PDB SSE annotation defines an eight-chain-spanning β-sheet, which is also identified by SCOT but based on separate β-sheets of different chain.

A blind spot of the methodology of SCOT is the lack of support for multi-chain β-sheets. In the 2018 copy of the PDB $51{,}448$ of $901{,}439$ ($5.71\,\%$) annotated β-sheets span at least 2 (e.g., 3aae@pdb)

and up to 30 (e.g., 5wz3@pdb, PDB annotation) different chains. Looking at these numbers one has to keep in mind, that they are highly biased as the PDB is by far not a representative set of protein structures and its provided annotation is equally not consistent. The β-sheets spanning fewer chains (e.g., 47,764 spanning 2 chains, PDB annotation) are also classified by SCOT in most cases as they often consist of combined β-sheets of separate chains. This holds also true for β-sheets spanning a higher number of different chains. Such β-sheets are β-barrels of channel-proteins in many cases and, thus, represent an important structural motif. Therefore, this is one of the most important issues on the agenda for the next version of SCOT. Figure 3.21 gives an example of a channel-protein containing a multi-chain-spanning β-sheet according to the PDB provided SSE annotation, which is also identified by SCOT.

In summary, SCOT represents a novel, widely applicable, and comprehensive method for SSE assignments which can be easily visualized by the accompanying PyMOL scripts. Its methodology, functionality, and versatility clearly detaches SCOT as an independent tool from its origins. The rigorous PDB file preprocessing steps ensure the reliable processing of most PDB files including modified residues, D-amino acids, insertion codes, and alternate locations. The output of a PDB file which includes the novel assignments ensures the immediate use of the SSEs for further analyses.

"I did not tell half of what I saw, for I knew I would not be believed."

Marco Polo

4

SNOT | Benchmarking SSE Classifications

4.1 Introduction

Whenever a task, a tool, or the pure human curiosity is eager for secondary structure information, one is faced with the question which classification or SSAM to choose. The formulation of this question, however, is not a trivial challenge in itself: which one is the most suitable, recent, consistent, reasonable, intuitive, ...? Due to the fact that the PDB [9] provides secondary structure information per se for its protein structures, this question is not asked as often as it should be.

To evaluate the researcher preferences for SSAMs, we searched through the entire 2018 copy of the PDB protein files for `DETERMINATION METHOD`, which is referred to the determination method of the secondary structure information. In more than 135,000 protein files we found approximately 1,600 times `DSSP`, 1,200 times `AUTHOR PROVIDED`, 6 times `MOE`, and even several times `TAKEN FROM ... PDB ENTRY ...`. A more detailed list can be found in Table 4.1. This clearly demonstrates that the secondary structure information in the PDB is neither consistent nor does it show a significant majority. Or, in more provocative words, the fact that in less than 5 % of these files the information about the determination is even given, reflects how little attention is paid to the assignment of SSEs.

Determination method	Occurrences
DSSP	1,700
AUTHOR PROVIDED.	1,021
AUTHOR DETERMINED	776
AUTHOR	158
AUTHOR PROVIDED	18
PROVIDED BY DEPOSITOR	15
	14
KABSCH AND SANDER	12
PROCHECK, WITH IDENTIFICATION	7
HELIX DETERMINATION METHOD	7
MOE	6
TAKEN FROM RELEASED PDB ENTRY 1VSF	6
TAKEN FROM PDB ENTRY 1AQ2.	4
BASED ON SUBMISSION 4APE	4
TAKEN FROM RELEASED PDB ENTRY 1AY6	4
...	
RAMACHANDRAN	2
AUTHOR DETERMINED BY USING PYMOL'S DSS-	2
DSSP AND O.	1
AUTHOR-MODIFIED KABSCH & SANDER	1
SEQUENTIAL AND MEDIUM-RANGE NOE	1

Table 4.1: Statistic on the declaration of `DETERMINATION METHOD` in the 2018 copy of the PDB.

There are more than 30 different SSAMs available in the literature (see Table 4.2). This number motivates a bunch of more questions and criteria enabling their differentiation, such as the supported SSE types (helix, sheet) and classes, the in- and output formats, whether it is a standalone tool or a web service, the supported operating systems, the programming language, the runtime, the availability, and many more. It also shows that there is no absolute secondary structure formalism or definition and the assignment is to some extent subjective [16]. This fact raises two questions. First, how can SSAMs be compared or evaluated? And second, is the lack of an objective tool to compare SSEs a major reason for the lack of a debate about their assignment?

This chapter is organized as follows: Section 4.2 depicts the state of the art of tools and parameters to evaluate SSE assignments. Section 4.3 motivates the necessity of SNOT and describes its methodology, i.e., its six Observers (or functionalities). In Section 4.6, we exhaustively evaluate and compare SCOT (see Chapter 3) to six different SSAMs, namely, DSSP [22], STRIDE [51], SHAFT [18], ASSP [65], DISICL [54], and SEGNO [53], using different aspects of SNOT. Finally, Section 4.7 discusses the findings of the previous section and motivates open challenges.

Method	DH	HB	GO	HX	α	3_{10}	π	ω	2.2_7	LH	LC	PPII	HK	SH	SK	TU	Cα	|Y|	Year	AV
SCOT [7]	●	●	●	●	●	●	●		●	●	●	●	●	●	●	●		12	2019	●
PSSC [48]	●	●		●	●	●	●			●		●		●		◐		7	2014	●
SHAFT [18]	●	●		●	●	●	●	◐								●		6	2011	
DSSP – PPII [49, 50]	●	●		●	●	●	●					●		●		●		5	2011	●
STRIDE [51]	●	●		●	●	●	●							●		●		6	1995	●
KAKSI [52]	●		●	●									●	●				3	2005	●
SEGNO [53]	●		●	●	●	●	●					●		●				6	2005	◐
DISICL [54]	●			●	●	●	●			●		●		●		●		8	2014	●
PROSS [55]	●			●								●		●		●		4	1999	●
Chen et al. [56]		●	●	●										●				2	2009	
Levitt & Greer [47]		●	●	●										●		●		3	1977	
β-Spider [57]		◉	●	●										●				3	2005	
DSSPcont [58]		●		●	●	●	●							●		◐		5	2003	◐
SECSTR [59]		●		●	●	●	●							●		◐		5	2002	◐
PROMOTIF [60]		●		●	●	●				●				●		●		6	1996	●
SSTRUC [61, 62]		●		●	●	●								●				5	1990	●
DSSP [22]		●		●	●	●	●			●	●			●		◐		5	1983	●
RaFoSA [63]			●	●	●	●	●							●		◐	●	6	2016	●
SACF [64]			●	●	●	●	●			●				●			●	7	2016	
ASSP [65]			●	●	●	●	●			●	●	●		●	◐		●	9	2015	●
Kneller & Hinsen [66]			●	●	●	●	●							●			●	6	2015	●
PCASSO [67]			●	●										●			●	3	2014	●
SST [40]			●	●						●	●			●		◐	●	5	2012	●
SABA [68]			●	●	●	●								●			●	5	2011	●
PMML [69]			●	●										●			●	3	2011	●
PROSIGN [70]			●	●	●	●	●							●			●	6	2008	◐
PALSSE [71]			●	●										●			●	3	2005	●
Taylor et al. [72]			●	●	●	●								●			●	5	2005	●
Zhang & Skolnick [11]			●	●								●					●	3	2005	◐
VoTAP [73]			●	●										●			●	3	2004	●
STICK [74]			●	●										●			●	3	2001	
XTLSSTR [75]			●	●	●	●						●		●		◐		5	1999	●
P-SEA [76]			●	●										●			●	3	1997	●
YASSPA (GETSSE) [77]			●	●										●			●	3	1997	●
P-Curve [78] (v3.1)			●	●	●	●	●			●				●				3	1989	◐
DEFINE_STRUCTURE [79]			●	●	●	●								●		●	●	5	1988	◐
SKSP [80]	◉	◉	◉	●										●				2	2007	●
CONSENSUS/TCM [81]	◉	◉	◉	●										●				1	1993	

Table 4.2: SSAMs grouped by their underlying methodology. SSAMs grouped by their underlying methodology based on dihedral angles (D), hydrogen bonds (B), or geometry (G) and their features: helix (HX), left-handed (LH), left-handed classes (LC), helix kinks (HK), sheet (SH), sheet kinks (SK), turns (TU), number of yes (|Y|), year of publication, and availability (A). A feature can be fully (●) or partly (◐) supported. A (◉) indicates that it is not applicable. β-Spider uses only the contact energy (◉). A (◐) in the last column indicates that an SSAM is available on request. This Table is extracted from [7].

4.2 State of the Art

During the development of our own classification tool SCOT (see Chapter 3), we were searching for a tool that can provide numerous geometric parameters and also supports the analysis of all types of SSEs and classes alike. There are some geometric parameters published [97, 25, 26] by which SSE assignments can be evaluated and compared. However, there is no tool that enables the automated calculation for datasets of protein structures. In addition, some of these parameters are not directly associated to a certain application, which gives them an abstract character. One exception is HELANAL [98]. It uses solely the Cα atoms of a protein to geometrically characterize helices. It calculates geometrical parameters, such as the Twist, Vtor, or BDAs between local successive helix axes. Unfortunately, HELANAL focuses solely on α-helices.

This lack of tools in general or a single tool in the best case to evaluate all aspects of an SSE assignment may be one of the reasons why DSSP [22] and STRIDE [51] are still the most commonly used tools neglecting their limitations and the existence of more sophisticated SSAMs.

4.3 SNOT

We have developed SNOT (Secondary structure Numeric Observation Tool) to combine a multitude of functionalities and parameters in a single tool (similar to the motivation of SCOT). In addition, SNOT provides new features which enable an exhaustive, objective, and easy to use evaluation method for SSE assignments. Furthermore, SNOT is not limited to an SSE type or class. Instead, SNOT processes SSE types and classes separately but also combines them to right-handed helices, left-handed helices, and extended conformations. The latter consists of PPII helices and β-sheets.

SNOT provides six different Observers (functionalities) (see Section 4.3.2) for the analysis of secondary structure annotations:

- **Geometry** - geometric properties
- **Residues** - residue statistics
- **Consensus** - consensus of two classifiers
- **Consistency** - consistency of the classification with respect to conformational changes
- **Overlaps** - analysis of overlapping SSEs
- **Coverage** - analysis of the sequence coverage

The Geometry Observer calculates a multitude of numerical, statistical, and geometrical properties, such as type-specific BDAs, or the Twist. These properties can provide a measure of conformational consistency for a secondary structure classification in itself. High deviations in these parameters

correspond to a less strict conformational consistency within a type and/or a class of SSE. The Residues Observer provides information on residue frequencies in SSEs. The Consensus Observer calculates the consensus of two classifications by the use of fingerprints. The bits corresponding to the sequence covered residues of SSEs are marked and the consensus for two fingerprints is reflected by the Tanimoto coefficient. The Consistency Observer operates on protein ensembles and calculates the consistency of a classification with respect to conformational variations. It also uses fingerprints in the same manner as the Consensus Observer but introduces an additional weighted Tanimoto coefficient measure. This weighted Tanimoto coefficient reflects the expected or more realistic consistency especially for multiple fingerprints. The Overlaps Observer uncovers the overlaps of SSEs of different types or different classes. Finally, the Coverage Observer determines the relative and absolute number of covered residues for each SSE type and class.

SNOT is written in C++.

4.3.1 Input

SNOT serially processes PDB files using the same parsing procedure as SCOT described in Section 3.3.1 with additional parsing of SSE annotations. Each provided input (command line argument) can either be a single PDB file or a directory containing PDB files.

4.3.1.1 PDB Files

The parsing procedure of SNOT requires standard PDB files as input and consists of two major steps. First, we parse a protein's sequence information. This step is identical to the parsing procedure used of SCOT (see Section 3.3.1.1). Second, we parse the SSE information from `HELIX`, `SHEET`, and `TURN` lines plus the additional information provided by SCOT in the corresponding `REMARK 650`, `REMARK 700`, and `REMARK 750` lines. However, this additional information is parsed optionally to maintain the full compatibility for PDB files not providing such information (e.g., with SSE annotations by other SSAMs).

Each `HELIX` line represents a helix. We extract the sequence information (front and back residue), the classification, and the comment for each helix. The kinks and purities specified in the `REMARK 650` section are assigned and added to a helix based on the helix identifier in columns 7–10.

Each `SHEET` line represents a strand of the corresponding β-sheet. The affiliation of a strand to a β-sheet is realized by the sheet identifier in columns 12–14. However, we use the sense in columns 39–40 to group the strands and start a new β-sheet whenever the the sense is 0 or the registration is not given. We add all strands to the present sheet until a new sheet is identified. This requires that all strands belonging to one β-sheet must be given consecutively. We extract the sequence information, the sense, and the registration (if present) from each `SHEET` line. Note that the first strand of a β-sheet does not have a registration with respect to the PDB file format. Also note that some SSAMs do not provide registration information. In such a case, all strands are added as

separate β-sheets, each solely consisting of a single strand. The kinks specified in the `REMARK 700` section are assigned and added to a strand based on the strand and β-sheet identifiers in columns 7–10 and 12–14.

Each `TURN` line represents a turn. As turns are not primary SSEs, they are rarely detected by other SSAMs and are not specified by the PDB file format. Thus, we solely support the `TURN` line format provided by SCOT (see Chapter 3). The format for the turns of SHAFT [18] provided by the Relibase [92] seems similar but still has some differences which are not supported in this parsing procedure. We extract the sequence information for each `TURN` line and parse the type, classification, and energy (distance for *open* turns) information from columns 67–68, 72–73, and 75–80 respectively. Please note that the support of turns is provided although they are not used by any of the observers yet.

4.3.2 Observers

There are six different Observers provided by SNOT to analyze the SSE information presented in the PDB files: Geometry (see Section 4.3.2.1), Residues (see Section 4.3.2.2), Consensus (see Section 4.3.2.3), Consistency (see Section 4.3.2.4), Overlaps (see Section 4.3.2.5), and Coverage (see Section 4.3.2.6). Each can be addressed using the functionality name as a command line flag, e.g., `--geometry`. In addition, each Observer provides a short documentation which can be evoked by the use of `--help`. These documentations also include a description of the output files column headings. In addition, all Observers require at least an output directory and support optional arguments, such as an output file extension (`-e`, default: `.txt`), and an output file column delimiter (`-d`, default: ,). All other arguments, flags, and options mainly focus on the input and are discussed individually for each Observer.

The Observers process each SSE type (helix or strand), class (e.g., α-helices), SSE groups (e.g., right-handed helices or extended conformations), and the entire protein separately. We combine all right-handed helices, left-handed helices, as well as PPII helices and strands (extended conformations) in separate groups. These groups allow the analysis of entire (conformational similar) groups and the comparison of SSAMs supporting different grades of details in the assignment of SSEs. In the following, we explain each observer for a given SSE type t and class c. For SSE groups, all types and classes of the group are combined and considered as one type and class. Please note that these groups are supported by the Consensus, Consistency, Coverage, and Residues Observers only.

4.3.2.1 Geometry

The Geometry Observer calculates geometric properties of SSEs. For each SSE type and class, the dihedral angles, numerical and geometrical properties, and BDAs (individually for helices and strands) are calculated. Each property is provided in separate files with suffixes `dih`, `res`, `sse`, `num`, and `bda`. We will explain the methodology leading to the parameters in each output file for a given

SSE type t and class c.

Dihedral Angles (`dih`)

The `dih` file contains the dihedral angles φ, ψ, and ω, plus the auto-scaled B-factor for each residue. The auto-scaling is a normalization according to Carugo and Argos [99]. For this procedure, the mean μ_b and the standard deviation σ_b of the B-factors of all backbone atoms of an input protein are calculated in advance. The auto-scaled B-factor $\mathrm{auto}(b_r)$ for a residue r is based on the mean B-factor of its backbone atoms and is defined in Equation 4.1. The result is a scaled B-factor distribution around the mean of 0 with unit variance.

$$\mathrm{auto}(b_r) := \frac{b_r - \mu_b}{\sigma_b} \tag{4.1}$$

Secondary Structure Elements (`sse`)

The `sse` file contains general information about the SSEs, such as the front and the back residue they are defined on as well as the length based on our internal residue identifiers. We also add the Purity from the `REMARK` section as provided by SCOT (see Chapter 3). If this information is not present, the purity is set to 1.

Geometric Properties (`geo`)

Based on the work by Sugeta and Miyazawa [97, 65], we calculate several geometric properties. All properties are calculated for sequence segments of length 4 within the sequence segment of each SSE. Thus, there are no such properties for SSEs of length < 4. For a given segment s, we calculate the Cα–Cα distance d between the residues at $s.front$ and $s.back$.

Let c_1, c_2, c_3, c_4 be the cartesian coordinates of the Cα atoms of the segment's residues. The vectors $\overrightarrow{b_1}, \overrightarrow{b_2}, \overrightarrow{b_3}$ represent the pseudo bonds between the atoms, $\overrightarrow{v_1}$ and $\overrightarrow{v_2}$ the planes that lie perpendicular to the axis of the helix described by c_1, c_2, c_3, c_4, and U the helix axis (see Equation 4.2).

$$\begin{aligned}
\overrightarrow{b_1} &:= c_2 - c_1 \\
\overrightarrow{b_2} &:= c_3 - c_2 \\
\overrightarrow{b_3} &:= c_4 - c_3 \\
\overrightarrow{v_1} &:= \overrightarrow{b_1} - \overrightarrow{b_2} \\
\overrightarrow{v_2} &:= \overrightarrow{b_2} - \overrightarrow{b_3} \\
U &:= \frac{\overrightarrow{v_1} \times \overrightarrow{v_2}}{|\overrightarrow{v_1} \times \overrightarrow{v_2}|}
\end{aligned} \tag{4.2}$$

We calculate the Rise (see Equation 4.3), the Twist (see Equation 4.4), Vtor, and the Radius (see

Equation 4.5). The parameter Vtor is calculated similarly to the dihedral angles φ, for instance, but is based on the Cα atoms of the segment's residues.

$$\mathrm{Rise}(s) := \frac{\overrightarrow{b_1} \cdot U}{|U|} \tag{4.3}$$

$$\mathrm{Twist}(s) := \arccos\left(\frac{\overrightarrow{v_1} \cdot \overrightarrow{v_2}}{|\overrightarrow{v_1}| \cdot |\overrightarrow{v_2}|}\right) \tag{4.4}$$

$$\mathrm{Radius}(s) := \frac{\sqrt{|\overrightarrow{v_1}| \cdot |\overrightarrow{v_2}|}}{(2 \cdot (1 - \cos(twist(s)))} \tag{4.5}$$

(a) Rise **(b)** Twist **(c)** Radius

Figure 4.1: Visualization of the Rise, Twist, and the Radius. Visualization of the Rise (a), the Twist (b), and the Radius (c).

The Rise represents the height of one helical coil and corresponds to the length of the coil's segment helix axis. The Twist reflects the twist of the helix axis and the Radius is determined perpendicular to the helix axis. These properties are visualized in Figure 4.1. Please note that the values may be negative under certain conditions, especially in the case of left-handed helices.

Finally, the class Purity is added so it can easily be correlated to the calculated properties.

Bending Angle (`bda`)

We use separate equations for the determination of the BDAs in helices τ_H and strands τ_S. In helices, we calculate the BDA τ_H in segments s of length 7 between the sub-segments $s_1 = (s.front, s.front + 3)$ and $s_2 = (s.front + 3, s.front + 6)$. For both segments, we calculate the U-matrices U_1, U_2 according to Equation 4.2. The BDA τ_H is calculated as described in Equation 4.6

$$\tau_H(s_1, s_2) := \arccos(U_1 \cdot U_2) \tag{4.6}$$

In strands, we calculate the BDA τ_S in segments of length 5. Let $r_1, \ldots, r_5$ be the underlying residues of a segment s. Let c_1, n_3, c_3, n_5 be the Cartesian coordinates of the C atom of r_1, the N

atom of r_3, the C atom of r_3, and the N atom of r_5 respectively. τ_S is based on the pseudo bond vectors $\overrightarrow{b_1}, \overrightarrow{b_2}$ between c_1, n_3 and c_3, n_5. We normalize $\overrightarrow{b_1}$ and $\overrightarrow{b_2}$ to a unit length of 1. The final BDA τ_S is the angle between these vectors (see Equation 4.7).

$$\begin{aligned} \overrightarrow{b_1} &:= n_3 - c_1 \\ \overrightarrow{b_2} &:= n_5 - c_3 \\ \tau_S(s) &:= \arccos\left(\frac{\overrightarrow{b_1} \cdot \overrightarrow{b_2}}{|\overrightarrow{b_1}| \cdot |\overrightarrow{b_2}|}\right) \end{aligned} \tag{4.7}$$

We also add the auto-scaled B-factor (see Section 4.3.2.1) to the file to be able to correlate the flexibility to the BDAs.

4.3.2.2 Residues

The Residues Observer calculates statistical properties of the underlying residues of SSEs. We will explain the methodology for a given SSE type t and class c.

For each SSE type and class $S_{t,c}$, we count the number of residues in total $R(S_{t,c})$ as well as the occurrences of each residue r individually in $R_r(S_{t,c})$. We count all standard residues separately. All other, such as modified residues, can be combined (`--group-non-standard-residues`) using the residue name XXX (alterable with `-g`). We also determine these values ($R(P)$ and $R_r(P)$) for the entire protein structures. We calculate the relative frequency $f(r)$ and the conformational parameter $p(r)$ according to Chou and Fasman [25] given in Equations 4.8 and 4.9.

$$f(r) := \frac{R_r(S_{t,c})}{R_r(P)} \tag{4.8}$$

$$p(r) := \frac{R_r(S_{t,c}) \cdot R(P)}{R(S_{t,c}) \cdot R_r(P)} \tag{4.9}$$

We also perform the d-test for significance according to Wilmot and Thornton [26] (see Equation 4.10).

$$d(r) := \frac{R_r(S_{t,c}) - \left(\frac{R_r(P) \cdot R(P)}{R(P)}\right)}{\sqrt{\left(\frac{R_r(P)}{R(S_{t,c})}\right) \cdot \left(1 - \frac{R_r(P)}{R(P)}\right)}} \tag{4.10}$$

This test indicates how significant the over- or under-representation of residue r in SSE $S_{t,c}$ is. For an easier perception, we also provide a bin based indicator D for the significance shown in

Equation 4.11. The different thresholds indicate different levels of significance (1.97: 5 %, 2.57: 1 %, 3.3: 0.1 %). In case of *?*, no statement regarding the significance can be made.

$$D(d(r)) := \begin{cases} +++ & +3.30 \leq d(r) \\ ++ & +2.57 \leq d(r) < +3.30 \\ + & +1.97 \leq d(r) < +2.57 \\ ? & -1.97 \leq d(r) < +1.97 \\ - & -2.57 \leq d(r) < -1.97 \\ -- & -3.30 \leq d(r) < -2.57 \\ --- & d(r) \leq -3.30 \end{cases} \tag{4.11}$$

4.3.2.3 Consensus

The Consensus Observer processes pairs of identical proteins (P_1, P_2) and compares their SSE annotation to determine the consensus with respect to an SSE. For each protein and each SSE type t and class c, a binary fingerprint $F_{t,c}$ with $|F_{t,c}| = |P_1| = |P_2|$ is created. The bits $b_i \in F_{t,c}$ corresponding to the residues covered by all SSEs of type t and class c are marked ($b_i = 1$). Given two fingerprints $F_{1,t,c}, F_{2,t,c}$ of two proteins P_1, P_2, the consensus is defined by the Tanimoto coefficient (see Equation 4.12) with $b_{1,i} \in F_{1,t,c}$ being the i-th bit of fingerprint $F_{1,t,c}$ and $b_{2,i} \in F_{2,t,c}$ defined analogously. $n = |F_{1,t,c}| = |F_{2,t,c}|$ is the length of the two fingerprints. If the denominator is 0, i.e., no bit is set in both fingerprints, the Tanimoto coefficient is defined as 1. The terms Tanimoto and Jaccard coefficient are often used synonymously.

$$\text{Tanimoto}(F_1, F_2) := \frac{|\{i \in \{1, \ldots, n\} | b_{1,i} = 1 \wedge b_{2,i} = 1, b_{1,i} \in F_1, b_{2,i} \in F_2\}|}{|\{i \in \{1, \ldots, n\} | b_{1,i} = 1 \vee b_{2,i} = 1, b_{1,i} \in F_1, b_{2,i} \in F_2\}|} \tag{4.12}$$

In a secondary structure type based fingerprint F_t, all bits are marked that correspond to residues that are covered by any SSE of type t. This is done analogously for SSE group fingerprints.

Finally, we calculate the consensus for the entire proteins by creating integer fingerprints based on helices and strands. We use the class value 100 for strands in set fingerprints to avoid collisions with helix classes. The PDB file format specifies that the helix class is a two digit number. Plus, the class 0 is used by SCOT (see Chapter 3), for instance, to indicate mixed helices of no specific class. Thus, it cannot be used to represent strands.

We provide the minimum, maximum, mean, and standard deviation at the end of each file. The fingerprints can be exported by the use of `--write-fingerprints`.

4.3.2.4 Consistency

The Consistency Observer reflects the consistency of an SSE classification with respect to conformational flexibility, such as present in NMR ensembles. It processes the ensembles with SSE assignments by multiple SSAMs to obtain a comparable consistency for each SSAM among the SSAMs. It requires that the residues within a specific ensemble among all classifiers are numbered equally. In other words, a residue r must have the same sequence number i in all models of an ensemble and among all classifiers. However, the sequences must not necessarily be of the same length and contain the same residues. We process all models of an ensemble in a preparation step and combine all model sequences to an ensemble sequence.

We provide two consistency measures, the Tanimoto coefficient (see Equation 4.14) and a weighted Tanimoto coefficient (see Equation 4.16). Let $F_1, \ldots, F_K$ be the fingerprints for the structures or models $\{P_1, \ldots, P_K\}$ of an ensemble E with SSE annotations. For each model $P_k \in E$ and for each SSE type t and class c, we create a binary fingerprint $F_{k,t,c}$ with $|F_{k,t,c}| = |P_k|$. We mark all bits in $F_{k,t,c}$ similar to the Consensus Observer described in Section 4.3.2.3. In a nutshell, all bits $b_i \in F_{k,t,c}$ are marked whose corresponding residues are part of an SSE in P_k of type t and class c. Let n be the sequence length of all models $\{P_1, \ldots, P_K\}$ of ensemble E.

For both Tanimoto coefficients, we start with the calculation of the number indices D based on the fingerprints of all SSAMs for this specific ensemble E. D contains the number of indices for which at least one bit $b_{k,i}$ at index i is set (see Equation 4.13) in any of the fingerprints of all SSAMs with $F_1, \ldots, F_A$ being all fingerprints of all SSAMs for ensemble E. This ensures that the consistency for the same ensemble for each SSAM is calculated relative to the same divisor. This results in a unified penalty score independent of the SSE lengths differences of multiple SSAMs. The limited domain ($[0.5, 1]$) of the weighted Tanimoto coefficient is due to the fact that, for the dividend, the majority of the number of bits set to 0 or 1 is taken into account which is at least $K/2$ here.

$$D(F_1, \ldots, F_A) := |\{i \in \{1, \ldots, n\} | \exists a \in \{1, \ldots, A\} : b_{a,i} = 1, b_{a,i} \in F_a\}| \tag{4.13}$$

Due to D, which is based on the fingerprints of *all* classifiers, the calculated consistencies are not comparable to consistencies obtained by a different set of classifiers.

$$\text{Tanimoto}(F_1, \ldots, F_K) := \frac{|\{i \in \{1, \ldots, n\} | \bigvee_{k=1}^{K} b_{k,i} = 1, b_{k,i} \in F_k\}|}{D} \tag{4.14}$$

One of the drawbacks of the standard Tanimoto coefficient is that if at an index i a bit in one of the fingerprints $F_{k,t,c}$ is not marked whereas it is in all others, this is counted as inconsistent although the majority of bits is consistent. Therefore, we introduce the weighted Tanimoto coefficient to cope with this challenge (see Equation 4.16). Here, we set the number of the most frequent bit value (b or $\overline{b}$) for a given index i in relation to the number of fingerprints n. Thus, the consistency for a given set of fingerprints is at least 0.5. This also means that the consistency of fingerprints that are

entirely marked or unmarked is 1.

$$C_i(F_1, \ldots, F_K) := |\{k \in \{1, \ldots, K\} | b_{k,i} = 1, b_{k,i} \in F_k\}| \tag{4.15}$$

$$\text{Tanimoto}_\text{W}(F_1, \ldots, F_K) := \frac{\sum_{i=1}^{n} \max(C_i(F_1, \ldots, F_K), 1 - C_i(F_1, \ldots, F_K))/K}{D} \tag{4.16}$$

An example of the benefit of the weighted Tanimoto in contrast to the regular Tanimoto coefficient is given in Figure 4.2. The helix at indices 96 to 106 is only missing in the last fingerprint. Taking only these indices into account the Tanimoto coefficient for this specific sequence region is 0 whereas the weighted Tanimoto coefficient is $0.8\overline{3}$, which better reflects the consistency.

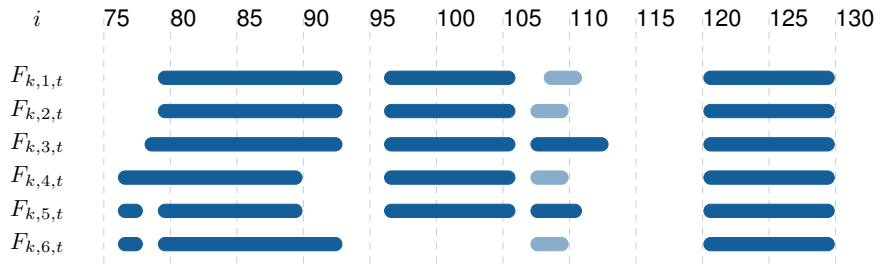


Figure 4.2: Generic example of the comparison of the Tanimoto and the weighted Tanimoto coefficient. Generic example of assigned helices for which the weighted Tanimoto coefficient (≈ 0.86) reflects the perceived or expected consistency more closely than the Tanimoto coefficient (≈ 0.49). Both are calculated based on all assigned helices.

Figure 4.2 also shows an example in which the overall $Tanimoto(F_{k,t}) = \frac{11+10}{43} \approx 0.49$ and $Tanimoto_W(F_{k,t}) = \frac{(2\cdot3)+4+(11\cdot6)+(3\cdot4)+(10\cdot5)+3+(2\cdot4)+2+(2\cdot5)+(10\cdot6)}{6\cdot43} \approx 0.86$. In the nominator, $(2 \cdot 3)$ are for the helices at residues 76 to 77, 4 (most consistent value is 0) for 78, and $(11 \cdot 6)$ for 79 to 89. The two Tanimoto coefficients are written to different files with suffixes `tan` and `wtan`. We provide the minimum, the maximum, the mean, and the standard deviation at the end of each output file. The binary fingerprints can be exported by the use of `--write-fingerprints`.

The creation of fingerprints for the SSE types and the SSE groups is done as described for the Consensus Observer (see Section 4.3.2.3).

4.3.2.5 Overlaps

The Overlaps Observer determines the overlap of SSEs. An overlap $\text{overlaps}(S_1, S_2)$ (see Equation 4.17) is the number of residues two SSEs S_1, S_2 have in common. For a protein P, we check for overlaps between helices, strands, and helices and strands.

$$\text{overlaps}(S_1, S_2) := |S_1 \cap S_2| \tag{4.17}$$

We also calculate the relative overlap with respect to the respective SSE length $|S_1|$ or $|S_2|$ which is exemplarily given for S_1 in Equation 4.18.

$$\mathrm{overlaps}_{\mathrm{S}_1}(S_1, S_2) := \frac{\mathrm{overlaps}(S_1, S_2)}{|S_1|} \tag{4.18}$$

We provide the minimum, the maximum, the mean, and the standard deviation at the end of each output file.

4.3.2.6 Coverage

The Coverage Observer calculates the SSE sequence coverage. For each protein P, a binary fingerprint $F_{t,c}$ of length $|F_{t,c}| = |P|$ is created. Note that every bit corresponds to a residue in P. The corresponding bits $b_i \in F_{t,c}$ of the residues of all SSEs of type t and class c are marked ($b_i = 1$). This is done similarly compared to the Consensus Observer described in Section 4.3.2.3. The $\mathrm{coverage}$ is defined by Equation 4.19.

$$\mathrm{coverage}(F) := \frac{|\{i \in \{1, \ldots, |F|\} | b_i = 1, b_i \in F\}|}{|F|} \tag{4.19}$$

We also create fingerprints F_t in which all SSEs of type t regardless of their class and a fingerprint F regardless of their type and class are taken into account and calculate the coverage accordingly. We report the minimum, the maximum, the mean, and the standard deviation at the end of each output file.

4.3.3 Output

Each Observer creates individual output files which are named according to the following scheme. The output file names consist of the secondary structure type (`H` for helix, `S` for strand, `P` for protein), the class (e.g., 1 for right-handed α-helices), the file type (e.g., `dih`), and the file extension. For instance, the file `H1-cvg.txt` contains the sequence coverages of right-handed α-helix (see Section 4.3.2.6). Files based on the entire protein are named `P-cvg.txt`, for instance. The file containing the coverage for the group of right-handed helices is named `H1+H3+H5-cvg.txt`. All output files are organized in columns using a comma as delimiter by default. The documentation for the columns is individual for each Observer and can be consulted using `--help`. Floating point values are given with a `4` digit precision which can be set globally in the source code. Residues are written according to the PDB file format in their fixed column representation consisting of the name, the chain identifier, the sequence number, and the insertion code.

4.4 Application of SSAMs

Most of the SSAMs provide output files in individual formats which were transformed into the PDB file format for evaluation purposes. To this end, we replaced the PDB secondary structure annotation (`HELIX` and `SHEET` lines) of the input files with the SSE assignments obtained from the individual tools. In all cases, the transformed `SHEET` lines did not contain a registration because, on the one hand, they were not provided in the output files of the geometry-based SSAMs, and, on the other hand, the registration was not required at any step of the evaluation. In the following, we describe the transformations we applied for each SSAM.

4.4.1 ASSP

For ASSP (version 1.0) [65], we used the `*_assp.out` files which contain a list of the assigned SSEs. Each element is annotated with its class as a character string (e.g., `AlphaHelix`) and integer number (e.g., 1), and the N- and C-terminal residues. We transformed each line into its PDB-conform equivalent. We changed the helix class number for left-handed 3_{10}-helices from 12 to 11 and left-handed π-helices from 11 to 13 which corresponds to the numbering used for the SCOT-assigned helices. A class number of 0 is used to indicate a strand or sheet. Insertion codes are not present in the output files which led to some errors in the final evaluation.

4.4.2 DISICL

Although DISICL (version 1.0) [54] provides files in different formats including the PDB file format, the contents of these files differ. For instance, in the output files in PDB file format (DISICL simple format), all π-helices are annotated as α-helices. Hence, we used a customized library which utilizes a combination of the simplified and detailed format. The detailed library contains the residues grouped into 18 different secondary structure classes. In compliance with the authors, we transformed these according to the following scheme: `Alpha-helix` and `Helix-cap` to α-, `3/10-helix`, `Turn type I`, and `Turn-cap` to 3_{10}-, `Pi-helix` to π-, `Left-handed turn` to left-handed, and `Polyproline-like` and `Beta-bulge` to PPII helices, as well as `Beta-strand` and `Beta-cap` to strands. Each residue list contains the residues annotated with an internal index numbering instead of the original sequence number. Each residue in the list implies that itself and its successor are classified as part of the SSE type defined in the list. Thus, we reported all segments of residues with consecutive indices plus the successor of the segment's last residue as SSEs according to the adjusted scheme. Consequently, the minimum length of helices and strands was 2. We noticed that DISICL fails to parse `ATOM` lines in which the y or z coordinate is ≤ -100.

4.4.3 MKDSSP

MKDSSP (version 2.2.1) is an implementation of the DSSP algorithm [22] and is available via the yum package manager in CentOS. It provides the output in the DSSP format. All residues of an input protein are annotated with several properties including the assignment to an SSE. We transformed this annotation according to the following scheme: `H` to α-, `G` to 3_{10}-, and `I` to π-helices, as well as `E` to β-strands. We scanned the residue sequence for consecutive segments with identical assignments. For helices, we also considered the chirality to differentiate between the left- and right-handed helix classes. If the chirality + was given, we assigned right-handed classes (α: 1, 3_{10}: 5, π: 3), and left-handed classes (α: 6, 3_{10}: 11, π: 13) otherwise. In addition, we also separated segments on differing chiralities. Each such segment was reported as an SSE and written to the PDB file.

4.4.4 SEGNO

Although SEGNO (version 3.1) [53] provides output files in the PDB file format, we used the information written to the standard output. The PDB file format output files contained errors, such as unaligned helix and strand identifiers, missing insertion codes, and missing helices in some cases. The standard output contains the residue-based SSE assignments similar to the DSSP format. We extracted this information similar to MKDSSP utilizing the following scheme: `H` as α-, `G` as 3_{10}-, `M` as mixed, `P` as PPII helices, and `F`, `I`, `Q`, `N` as β-strands.

4.4.5 SHAFT

The SHAFT [18] classification is realized by our in-house Relibase [92] server (version 3.3.0). Output files in the PDB file format are available from the server. It appends the helix annotation to the one from the PDB using the comment `SHAFT` for the assigned helices. The strand assignment from the original PDB file is retained. However, the visualization program UCSF Chimera [17] does not support comments in `HELIX` lines and fails to read the SSE information upon opening. During the processing of the input files, we noticed the following issues. `TER` lines were ignored in some files of the NMR ensembles. The element symbol X, some space groups, and `LINK` lines are not supported and had to be removed prior to processing. Journal article titles (`JRNL`) in the `REMARK` section have to be limited to 255 characters. For modified residues, their `HETATM` lines are renamed to `ATOM` lines, the name of the residue itself is changed to the original name, and the atoms corresponding to the modification are added to the `HETATM` entries connected via `LINK` records. Finally, protein residues are identified by their N, Cα, C, and O atoms. However, if a ligand residue contains atoms with these names, it also listed in the `SEQRES` entry of the output file. In consequence, although the Relibase provides files in the PDB file format, we used the input PDB files and replaced the HELIX lines with the corresponding lines from the Relibase to avoid differently parsed protein structures in the analysis of SSAMs. In addition, we disregarded the comments in the HELIX lines.

Here, SHAFT corresponds to the implementation provided by the Relibase and not to our reimplementation described in Section 3.4. In contrast to our implementation, the Relibase version is published and available to the scientific community, which enables the reader to reproduce the following results. In addition, our bug fixes and changes to the implementation may not comply with the authors.

4.4.6 STRIDE

STRIDE (release 01/29/96) [51] also provides an individual format which is similar to the ASSP output used herein. In contrast, the respective lines (`LOC`) lack the integer number to identify the class of an SSE and solely provide the character string. Thus, we used the following scheme to transform the secondary structure annotations: `AlphaHelix` as α-, `310Helix` as 3_{10}-, `PiHelix` as π-helices, and `Strand` as β-strands.

4.5 Analysis of SSAMs

For the analysis of SCOT and six other SSAMs, we used the Observers of SNOT. In addition, we analyzed the following use cases.

Structural alignments of the protein structures of the same CATH topology and superfamily (see Section 2.3.7.1) were performed using two methods. To this end, the CATH domains of the corresponding PDB structures were extracted and the SSEs were assigned using the SSAMs discussed in this study.

The UCSF Chimera MatchMaker [100] was applied for the CATH superfamily pairs. The impact of SSE information on the initial sequence alignment was varied from 0 to 1 with a 0.1 step size. An SSE similarity contribution of 0.8 was determined as the optimum to successfully align all superfamily pairs of the dataset. Apart from that, default settings were applied. Alignments of the domain pairs were performed for the proteins with SSEs assigned by the different methods and the RMSD values per protein pair (no iterative alignment optimization) were evaluated.

LOCK2 [101]-based structure alignments for both, the superfamily and the topology dataset, were obtained using default settings. The resulting scores were normalized by the number of overall matched SSEs per alignment.

For SCOT, different SSE assignments of the domains were used. Apart from the default assignments, we omitted π-helices for the alignment, split strands based on the strand kink data, and split both, helices and strands, based on the corresponding kinks. For all these settings, we evaluated the alignment performance for both datasets to obtain optimum settings for SCOT-based assignments (i.e., settings which result in low RMSD values and high per-residue-SSE scores using LOCK2 [101]). These settings can be applied to successfully superpose topologically similar protein structures.

4.6 Results

This section demonstrates the applicability, the versatility, and the benefits of SNOT by comparing SCOT to six other state-of-the-art SSAMs. Please be reminded that this section uses the SHAFT classification provided by the Relibase and explicitly not our reimplementation described in Section 3.4.

4.6.1 Yet Another SSAM?

Faced with a multitude of available algorithms for the automated assignment of SSEs in proteins, the question arises whether yet another method is necessary. Despite the overwhelming number of available SSAMs, DSSP and STRIDE are the most commonly used methods. Comparing the SSAMs' citation counts in Web of Science v.5.30 [102], a clear superiority in citation count per year of both methods over all other methods can be observed (with exceptions for tools which were published in addition to further results or methods). In this publication, we do not only present SCOT as a novel alternative for the SSE assignment. We also try to highlight the advantages of hydrogen bonding-based SSAMs over others which also explain the methods' reception.

Table 4.2 provides an overview of available SSAMs together with their year of publication and supported features. On a first glance, it is obvious that the major differences between the methods are the underlying approach and the resulting differences in the assigned SSE classes. Furthermore, the availability is not guaranteed for all methods and some are only available as web servers restricting their use to a small number of protein structures. Most tools support the assignment of helical and extended conformations, but only a small subset of methods assigns rare SSEs, such as left-handed helices, or turns. Moreover, there are only two other methods that indicate kinks (ASSP in strands and KAKSI in helices) within regular SSEs. These criteria indicate a major drawback which prompted us to design the novel method SCOT.

For the analysis of SSEs, the use of more than one tool is often necessary, e.g., assigning helices with STRIDE and subsequently finding kinks with tools, such as Kink Finder [103], MC-HELAN [104] or HELANAL [98]. In contrast, our method SCOT is the very first method that enables the most extensive analysis of SSEs and turns in proteins. It assigns most of the known helix classes, as well as kinks for both, helices and strands, in a single step. Furthermore, it provides an easily interpretable output for the visual inspection of all SSE characteristics.

To investigate the general applicability of SCOT and the reliability of the assigned SSEs, we picked six other SSAMs (DSSP [22], STRIDE [51], SHAFT [18], ASSP [65], DISICL [54], and SEGNO [53]) to put the results of our method into context based on different quality criteria. These were selected as they cover most of the SSE classes that are assigned by SCOT.

4.6.2 The SCOT Secondary Structure Assignment

Turns in the proteins' backbone are the basis for all assignments discussed in this section. We distinguish *normal* (hydrogen bond from the backbone carbonyl O of residue r_i to the nitrogen H of residue r_{i+k}), *reverse* (hydrogen bond N–H of residue r_i to C–O of residue r_{i+k}) and *open* turns which are characterized by distinct Cα–Cα distances between 4 and 8 Å and do not contain a backbone hydrogen bond (see Section 3.3.2). These turns are further subdivided based on their dihedral angle ranges with the help of trained ESOMs (see Table 6.1, appendix for the average dihedral angles, hydrogen bond energies, and four-residue segment Cα–Cα distances of the respective turn classes). SCOT assigns multiple helix classes and extended conformations based on a distinct set of underlying turn classes, hydrogen bonding patterns, and geometric criteria. These are summarized below and compared to those of the other SSAMs.

4.6.3 Helices

4.6.3.1 Right-Handed α- and 3_{10}-Helices

Right-handed α- and 3_{10}-helices are assigned based on overlapping *normal-5 1* and *normal-4 1* turns. Per-residue overlaps of at least 2 with lengths of 3 (α) and 2 (3_{10}) are required for an initial assignment of a helix core (see Section 3.3.4.5). These initial cores are extended by overlapping turns of classes *open-5 1*, *open-4 2*, and *open-6 4* with a Cα–Cα distance below or equal to 8 Å (see Figure 6.1, appendix for Ramachandran plots for *open* and *normal* turns of these classes). Overlapping α- and 3_{10}-helices are subsequently merged to yield a unique residue-based helix assignment. The final class of the helix (α, 3_{10}) is determined based on the number of per-residue overlaps of the two helix-constituting *normal* turn classes. In case of equal numbers of turn overlaps for both turn classes, a helix class cannot be assigned and we define them as mixed helices (class 0). Given the permissive nature of these assignments based on both, hydrogen-bonded and non hydrogen-bonded turns, a further post-processing step was required to get rid of kinked and irregular helical structures (see Figure 4.3).

We assign four-residue segments for the complete protein structure and calculate the Cα–Cα distances (four-residue segment Cα–Cα distances). Regular helical regions are characterized by distinct optima for these distances (see Figures 3.9a and 3.9b for the four-residue segment Cα–Cα distance histograms). As presented in the histogram in Figure 4.4, the introduction of distance cut-offs to split α- and 3_{10}-helices leads to less bent helices for SCOT. The validity of these overlaps, which also ensure proper φ and ψ angles and helical parameters, were evaluated using the X-ray representatives dataset.

In general, SCOT-assigned helices are most similar to those of STRIDE (see Table 4.3). A complete comparison of different geometric parameters and dihedral angles of α- and 3_{10}-helices assigned by the discussed algorithms can be found in Table 6.2, appendix. The high fluctuations of the φ and ψ angles for DISICL, the PDB classification, and SHAFT are in line with highly deviating

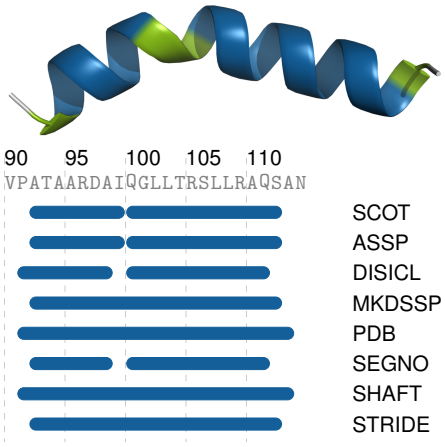


Figure 4.3: Different α-helix assignments. Different right-handed α-helix assignments for residues 90–114 of chain A of the structure 3rxy@pdb. The Purities of the SCOT-assigned helices are α: 0.893, 3_{10}: 0.107 and α: 0.898, 3_{10}: 0.102 Residues 98–101 show BDAs of 28.3°, 44.7°, 48.1°, and 29.2°. This Figure is reproduced by permission of Bioinformatics (2019) [7].

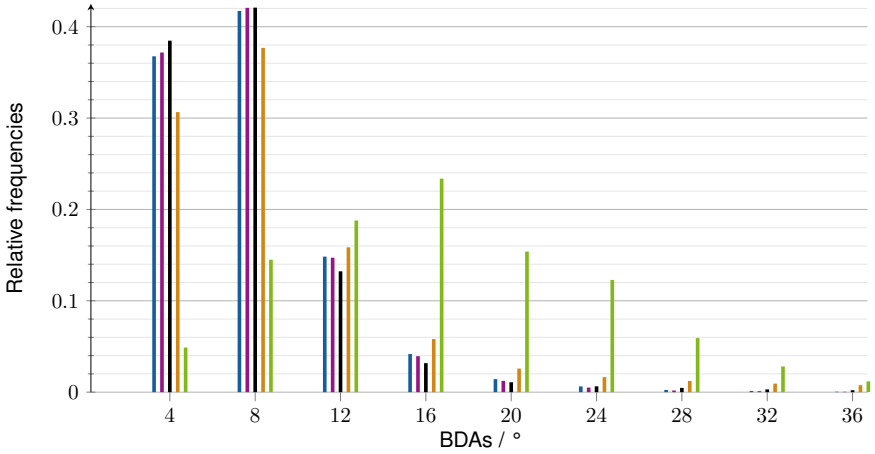


Figure 4.4: BDA histograms of the distances for α-helices assigned by different SSAMs. BDA histograms of the distances for α-helices assigned by SCOT (blue), $SCOT_{kinked}$ (purple), ASSP (black), and SHAFT (orange). BDA histogram for the SCOT assigned kink residues (green). This Figure is reproduced by permission of Bioinformatics (2019) [7].

helical Twists and Radii as well as comparatively high bending angles (BDAs). Additionally, the scaled B-factors for the helical residues are significantly higher. The tools characterize helices in less stable protein regions. Together with ASSP, DISICL, MKDSSP, and SEGNO, SCOT-assigned α-helices show the most stable values for the Radius, the Rise, the Twist, and the Vtor. Significant length differences can be shown for SHAFT and the PDB with approximately 2 residues longer helices. In contrast, DISICL helices are on average shorter. The huge difference in the assignment of helix termini, which is a topic of major interest, shows huge discrepancies in their definition as given in Figure 4.3. This underpins the findings of Tyagi and co-workers [16] for nine different SSAMs. Concerning over- and underrepresented residue types in α-helices, SCOT is most similar

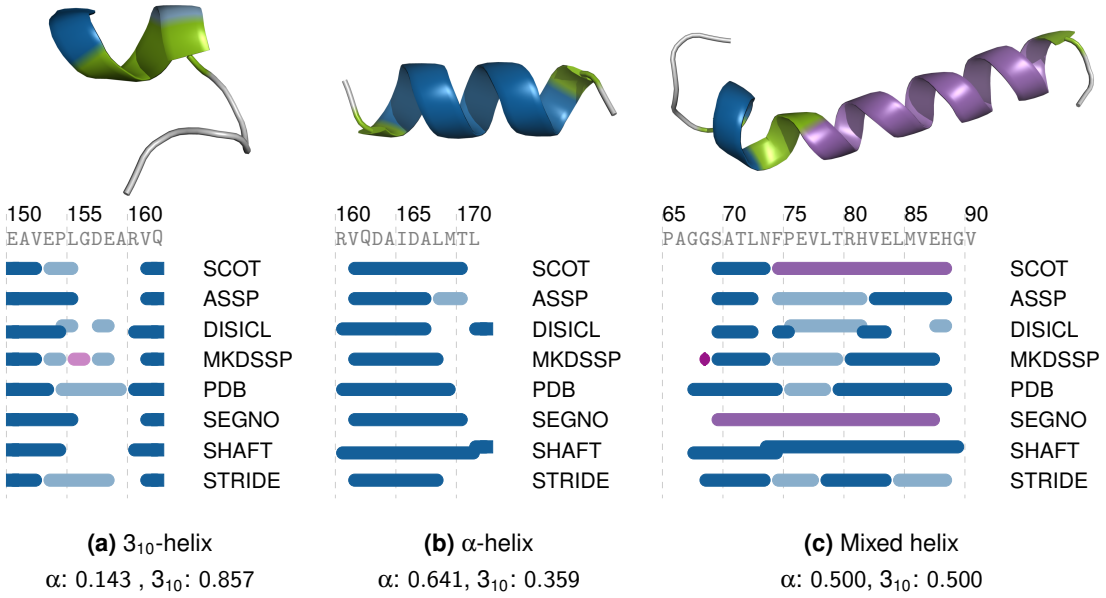
to MKDSSP (see Table 6.4, appendix for the conformational parameters of residues in α-helices). Altogether, the consensus for α- and 3_{10}-helices is highest for SCOT and the MKDSSP classification (see Tables 6.12a and 6.12b, appendix for the consensus of all methods).

H1+H3+H5	SCOT	ASSP	DISICL	MKDSSP	PDB	SHAFT	SEGNO	STRIDE
SCOT	1	0.8777	0.6536	0.8792	0.8095	0.6502	0.7908	0.9185
ASSP		1	0.6185	0.8253	0.7626	0.6137	0.7478	0.8500
DISICL			1	0.6214	0.6764	0.4496	0.6762	0.6575
MKDSSP				1	0.8147	0.6479	0.7665	0.9047
PDB					1	0.5553	0.8975	0.8182
SEGNO						1	0.5397	0.6487
SHAFT							1	0.7764
STRIDE								1

Table 4.3: Consensus of different SSAMs in the assignment of right-handed helices for the X-ray representatives dataset.

SCOT assigns the geometrically most consistent 3_{10}-helices when compared to all other methods (see Table 6.2, appendix for an overview of all geometric characteristics). The average 3_{10}-helix length is 4 residues. DISICL-, MKDSSP-, SEGNO-, and STRIDE-assigned 3_{10}-helices are on average shorter, while these of the PDB and the SHAFT assignment are longer which is in accord with higher φ and ψ angle deviations. The scaled mean B-factor for 3_{10}-helices is higher than that for α-helices for all assignments in this analysis underpinning the studies of Enkhabayar and co-workers [105]. They characterized 3_{10}-helices as para-helices given their instability and high variances in the geometric parameters. For *normal-4* turns of class 2, which are used for the 3_{10}-helix assignment in our study, we observe lower hydrogen bond energies as compared to *normal-5 1* turns. Moreover, Pro residues are overrepresented in this helix class according to SCOT, MKDSSP, PDB, SEGNO, SHAFT, and STRIDE (see Table 6.4, appendix for the conformational parameters of residues in 3_{10}-helices).

Based on the structure 3rxy@pdb (chain A), the classification of α-, 3_{10}-, and mixed helices by SCOT and their assignment by other methods can be discussed. Mixed helices are rarely assigned by SCOT to meet the problem of $r_i \rightarrow r_{i+3}$ hydrogen bonds occurring within α-helices. The fraction of residues assigned as mixed helix residues is $10.35\,\%$ for SEGNO whereas the fraction of SCOT-assigned mixed helix residues is $0.09\,\%$ in the X-ray representatives dataset (see Table 4.4). The overall consensus between the SCOT- and SEGNO-assigned mixed helices is only 0.0028 (see Table 6.12d, appendix). The average helical parameters together with the dihedral angles are given in Table 6.2 of the appendix. For mixed helices assigned by SCOT, all of these characteristics lie between those of α- and 3_{10}-helices underlining the difficulty of a unique assignment. Due to bifurcated hydrogen bonds occurring in α- and 3_{10}-helices, we find helices for which a final classification is not possible. Figure 4.5 gives an example of predominantly α-helical, predominantly 3_{10}-helical, and mixed class residue backbone segments in 3rxy@pdb together with their corresponding three-dimensional helix structure. It provides an overview of assignments and highlights major differences between all methods. (For SCOT classifications, we assign relative helix Purities which are reported in `REMARK 650` of the output PDB file.) For our representative



(a) 3_{10}-helix
α: 0.143 , 3_{10}: 0.857

(b) α-helix
α: 0.641, 3_{10}: 0.359

(c) Mixed helix
α: 0.500, 3_{10}: 0.500

Figure 4.5: Examples of the assignment of different helix classes by SCOT and six other SSAMs. Examples of the assignment of 3_{10}- (a), α- (b), and mixed (c) helices by SCOT and six other SSAMs for the protein structure 3rxy@pdb (chain A). This Figure is reproduced by permission of Bioinformatics (2019) [7].

dataset, α-helices show an average Purity of 0.87 while 3_{10}-helices are characterized by an even higher average Purity of 0.92 (see Table 6.2, appendix). The slightly lower Purity of α-helical structures can be attributed to the occurrence of $r_i \rightarrow r_{i+3}$ and $r_i \rightarrow r_{i+5}$ hydrogen bonds. The helices which are constituted by the latter pattern are discussed in Section 4.6.3.2. In contrast to geometry-based methods SCOT, does not rely on uniform φ and ψ angles which were shown to be inappropriate for 3_{10}-helix definitions [105]. We focus on hydrogen bonding patterns to assign the final helix classes.

Due to our helix assignment using Cα–Cα distances for helix splitting purposes, it is no longer necessary to assign kinks for SCOT-derived helices. The number of kinks (BDA > 30°) in α- and 3_{10}-helices for SCOT is 250. SCOT rarely assigns continuous helical stretches in bent regions. Nevertheless, kinks in the assigned helices can still be observed. We tried to identify further kinks by an additional hydrogen bonding pattern. An analysis of known helix kinks [106] revealed that they are characterized by a non-consecutive sequence of turn overlaps. Looking for regions with missing hydrogen-bonded turns might assist in the identification of bent regions in protein helices as those regions should consequently be more flexible. This was already discussed in a previous study focusing on the identification of kinked regions and residues involved at these kink positions [103]. We analyzed the conformational parameter of residues at and in the immediate sequential proximity of the assigned kinks and found a preference for Pro at position r_{k+1} which is in line with previous studies [104]. The preferred residue at SCOT-defined kinks is Leu (see Table 4.5 for the conformational parameters of residues at kink positions). While also prevalent in helices (helix-stabilizing), we find a much higher conformational parameter for Leu at the kink position (2 vs. 1.3 for α-helices). Intriguingly, only 24 % of our hydrogen bond-based kinks show BDAs above 20° and only 3 % have BDAs above 30° (previously characterized as kink regions [85]).

	SCOT	ASSP	DISICL	MKDSSP	PDB	SEGNO	SHAFT	STRIDE
H0 (mixed)	0.0009	n/a	n/a	n/a	n/a	0.1035	n/a	n/a
H1 (RH α)	0.3276	0.3139	0.3920	0.3073	0.3718	0.2207	0.3926	0.3260
H3 (RH π)	0.0040	0.0047	0.0066	0.0054	n/a	n/a	0.0004	0.0002
H4 (RH 2.2_7)	0.0000	n/a	n/a	n/a	n/a	n/a	n/a	n/a
H5 (RH 3_{10})	0.0391	0.0331	0.1597	0.0361	0.0504	0.0222	0.0453	0.0409
H6 (LH α)	0.0000	0.0000	0.0145*	0.0068	n/a	n/a	n/a	n/a
H8 (LH 2.2_7)	n/a	n/a	n/a	n/a	0.0000	n/a	0.0188	n/a
H10 (PPII)	0.0126	0.0142	0.0890	n/a	n/a	0.0291	n/a	n/a
H11 (LH 3_{10})	0.0004	0.0001	n/a*	0.0026	n/a	n/a	n/a	n/a
H13 (LH π)	n/a	0.0000	n/a*	0.0002	n/a	n/a	n/a	n/a
S0 (β)	0.1937	0.2006	0.2819	0.2060	0.2063	0.2558	n/a**	0.2084
SUM	0.5783	0.5666	0.9437	0.5644	0.6285	0.6313	0.6634	0.5755

Table 4.4: Sequence coverage of different SSAMs for the X-ray representatives dataset. The coverage is defined as the number of residues assigned to an SSE type divided by the number of residues in a protein. The summed coverages for ASSP, DISICL, PDB, SHAFT, and SCOT are biased as these methods assign overlapping SSEs. The handedness of helices is given by RH (right-handed) and LH (left-handed). SSE classes not supported by SSAMs are annotated with n/a. * As DISICL assigns left-handed helices without a specific class, the coverage for all assigned left-handed helices is summarized in row H6. ** The strand information in SHAFT corresponds to the sheet entries in the original PDB file. This Table is extracted from [7].

Consequently, the residues assigned by SCOT are kink-prone (i.e., hot spots for kink formation) rather than indeed the source of a helical deformation. Asking what kinks we miss, we analyzed the helices with residue positions with high BDAs. We find 48 helical residues with BDAs above 30° in helices with at maximum 9 residues. Splitting them by kinks would lead to very short helical stretches and the elimination of many short helices. This is in line with a generally rising BDA with decreasing helix length [107]. The remaining approximately 200 cases of high BDAs are cases that were neither found based on high Cα–Cα distance nor by a minimum of hydrogen bond interactions. Consequently, the use of HELANAL [98] or KinkFinder [103] might help to identify further geometric kinks in the structures of interest.

4.6.3.2 Right-Handed π-Helices

Similar to the co-occurrence of α- and 3_{10}-helices, π-helices are frequently observed as part of other helix classes [108] and are also referred to as α-bulges [82]. The merging of α-, 3_{10}-, and π-helices for SCOT led to a low number of assigned π-helices. In SCOT, π-helices are regarded as a special class and do not undergo the merging steps as explained for the previously discussed helix classes. π-helices are assigned on the basis of overlapping *normal-6 2* turns with an overlap length of at least 5 and an overlap count of at least 2. I.e., at least two consecutive *normal-6 2* turns are required for a π-helix assignment. For the X-ray representatives dataset, we assigned 572 π-helices of which 552 (96.5 %) overlap with α-helices by at least one residue (113 (19.8 %) are completely included) and 68 (11.9 %) with 3_{10}-helices by at least one residue (1 is completely included). This inconsistency is also reflected by the low average π-helix Purity of 0.56 for this

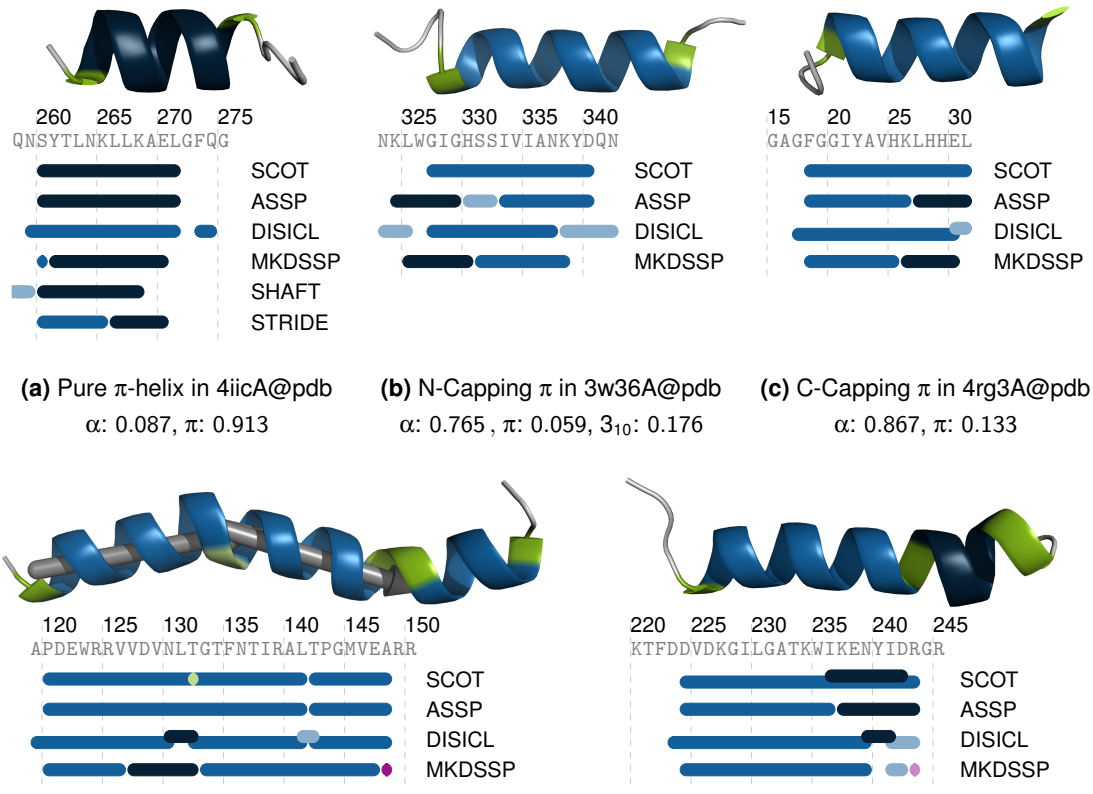
H1+H3+H5	r_{k-3}	r_{k-2}	r_{k-1}	r_k	r_{k+1}	r_{k+2}	r_{k+3}
Ala	1.0651	1.6737	1.1086	1.1795	1.1520	1.4563	0.6956
Cys	0.3074	0.3074	0.7684	1.0425	0.6148	0.7684	0.9221
Asp	0.7254	0.7254	1.0093	0.6062	0.6623	0.4731	0.4100
Glu	0.8409	1.0371	1.4855	0.7130	0.9810	0.8128	0.7848
Phe	1.3678	1.5093	0.8961	1.4397	1.1320	1.5565	1.3678
Gly	0.4641	0.5373	0.7327	0.5385	0.8304	0.4885	1.2212
His	0.5241	0.5241	0.9171	0.6295	0.3275	0.7206	0.5896
Ile	1.6747	0.8038	1.1053	1.2874	0.9043	0.8373	1.3397
Lys	0.8471	1.0165	1.1182	0.9960	1.1859	1.7958	0.8132
Leu	1.5424	1.5836	0.9049	1.9995	1.1517	1.1106	1.6658
Met	1.1141	1.6206	1.2154	0.7443	1.3167	0.9116	1.4180
Asn	1.0242	1.0242	0.8016	0.7048	0.5344	0.5344	1.0687
Pro	0.7592	0.6394	0.2797	0.3840	2.6374	0.6394	0.2797
Gln	1.3698	1.2176	1.3698	1.0324	1.2684	1.3698	1.2176
Arg	0.9183	1.2121	1.6162	1.0174	0.9183	1.0285	1.1019
Ser	0.7653	0.7653	0.6428	0.5018	0.8571	0.9489	0.4592
Thr	0.6931	0.5198	0.8317	0.8032	0.5892	1.3516	0.9011
Val	1.2015	0.6141	1.1214	1.1471	0.7743	1.0413	1.2282
Trp	1.6388	1.5022	1.2291	1.4667	1.3657	0.6828	0.9560
Tyr	0.8750	0.7109	0.8750	1.4838	0.7656	0.7656	1.4766
XXX	1.4142	2.1213	1.0607	0.9993	1.4142	1.7678	1.7678

Table 4.5: Conformational parameter for residues in SCOT-defined helix kink regions. Conformational parameter P as defined by the method of Chou and Fasman [25] for residues in SCOT-defined helix kink regions. This Table is extracted from [7].

dataset with a maximum Purity of 0.913 (see Table 6.2, appendix). Nevertheless, we identified three π-helices with Purities above 0.8 which were also detected by other methods (see Figure 4.6a for an example). This contradicts the hypothesis that π-helices do not occur as regular independent SSEs [109]. The average π-helix length for ASSP and SCOT is 6 residues. The higher average BDA for π- and 3_{10}-helices suggests that the occurrence of *normal-6* and *normal-4* turns leads to deviations from the ideal helix geometry. Intriguingly, π-helical stretches are characterized by a lower B-factor than α- and 3_{10}-helices.

The stabilizing effects of predominantly hydrophobic and aromatic residues in π-helices [110] (see Table 6.4, appendix for overrepresented residues in π-helices) might be a reason in addition to the higher number of backbone hydrogen bonds due to the co-occurrence of *normal-5* and *normal-6* turns constituting π-helices (see Purity in Table 6.2, appendix). The residue preferences differ between the methods designed for the assignment of π-helices (see Table 6.4, appendix for the conformational parameter of π-helices based on different SSAMs). For most assignment methods, Ala, Gly, and Pro residues are underrepresented while Val, Leu, Ile, Phe, and Tyr residues are preferentially found. The assignments of SCOT and MKDSSP are most similar (see Table 6.12c, appendix for the consensus π-helix assignments of all methods). STRIDE and SHAFT classifications are significantly different from all other methods and their consensus is

0.08. Both methods assign only 36 or 47 π-helices for the complete dataset. The main reason is the hierarchy underlying SHAFT and STRIDE which prefer α- and 3_{10}-helix assignments over π-helix assignments. They find only π-helices whenever there is a significantly higher proportion of $r_i \rightarrow r_{i+5}$ hydrogen bonds within a helical region. Therefore, both tools were excluded for the following analyses.



(a) Pure π-helix in 4iicA@pdb
α: 0.087, π: 0.913

(b) N-Capping π in 3w36A@pdb
α: 0.765 , π: 0.059, 3_{10}: 0.176

(c) C-Capping π in 4rg3A@pdb
α: 0.867, π: 0.133

(d) Kink in 4nbrA@pdb
α: 0.891, π: 0.109

(e) Overlapping π-helix 4q2wA@pdb
α: 0.469, π: 0.375, 3_{10}: 0.156

Figure 4.6: Differences in the assignment of π-helices by different SSAMs. (a) The π-helix in 4iicA@pdb is characterized by predominantly occurring $r_i \rightarrow r_{i+5}$ hydrogen bonds and does not overlap with other helix classes. In contrast, the structures 3w36A@pdb (b) and 4rg3A@pdb (c) contain α-helices which are N- and C-terminally capped by $r_i \rightarrow r_{i+5}$ hydrogen bonds. (d) An α-bulge which leads to a kinked (highlighted in green) α-helical structure in 4nbrA@pdb. (e) A π-helix with overlapping *normal-6 2* which can be found inside an α-helix in 4q2wA@pdb. This Figure is reproduced by permission of Bioinformatics (2019) [7].

Given the fact that we assign also less π-helices, we analyzed potential reasons. We searched for structures with π-helices assigned by MKDSSP (951) but missed by our method. Our prerequisite for π-helical cores are at least two overlapping *normal-6 2* turns. This leads to the omission of π-helical turns which are predominantly assigned C-terminally (179) and N-terminally (54) of α-helices as exemplarily shown in Figures 4.6b and 4.6c. In the case of N-terminally omitted π-helices, the common overlaps are between the last or the first residue of the π-helix and the first of our α- and 3_{10}-helices. Rather than π-helices, those structures might be considered as capping motifs consisting of a *normal-5 1* turn and a succeeding *normal-6 2* turn constituting the helix cap. The corresponding N- and C-terminal residues of the *normal-6* turn are mainly hydrophobic and/or

aromatic and probably lead to a stabilization of the helix terminus.

Additionally, α-bulges within helices consisting of only one $r_i \rightarrow r_{i+5}$ hydrogen bond are not classified as π-helices by SCOT (92 cases). In those cases, the α-helix is split into two parts due to an increased four-residue segment Cα–Cα distance (see Figures 4.6d and 3.9c, for the four-residue segment Cα–Cα distance histogram of π-helices). Finally, we identify π-helices that are differently classified by the other algorithms. One example is given in Figure 4.6e. These cases are especially interesting as they constitute helical stretches which are characterized by overlapping $r_i \rightarrow r_{i+3}$, $r_i \rightarrow r_{i+4}$, and $r_i \rightarrow r_{i+5}$ hydrogen bonds.

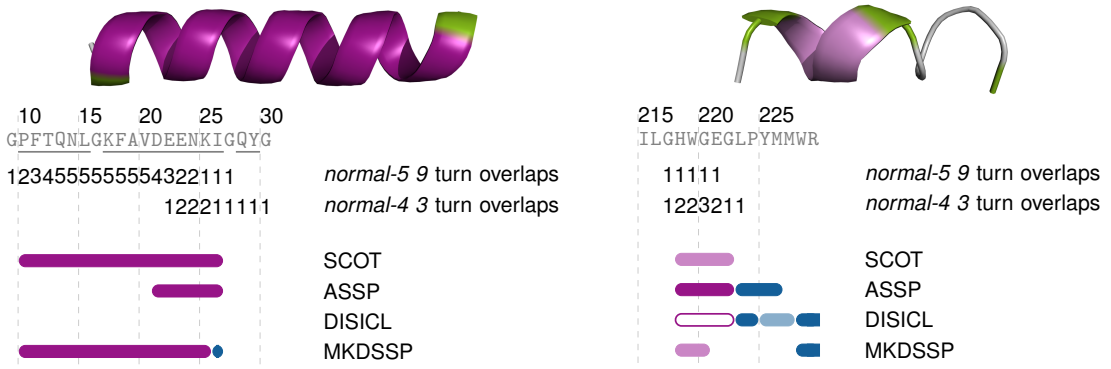
Given these results, we argue that only the combination of hydrogen bonding pattern and geometric criteria provides a comprehensive and reliable classification of π-helices. The lack of single *normal-6* turn helices in the center of helices as well as at the termini of α-helices which are characterized by a high flexibility leads to stable π-helix assignments. The geometric parameters of the π-helices are robust when compared to DISICL and the constant φ and ψ angles point toward a highly unique π-helix classification. The use of our π-helices for, e.g., protein structure alignment is arguable due to the possible overlaps with other helix classes, but their assignment might be interesting for further analyses regarding the evolution and function of this rare SSE class.

4.6.3.3 Left-Handed Helices

Left-handed helices are also rarely occurring in protein structures and their lengths are mostly restricted to some residues [111]. We identified a *normal-4* and a *normal-5* turn class which overlap in left-handed helices and whose dihedral angles agree with those of left-handed helical conformations in proteins. In contrast, we could not identify a hydrogen-bonded *normal-6* turn class with the dihedral angles suitable for the assignment of left-handed π-helices. We used the dataset of Novotny and Kleywegt [111] to compare our approach to ASSP, DISICL, and MKDSSP. DISICL assigns no left-handed classes while ASSP and MKDSSP assign and differenciate left-handed α-, 3_{10}-, and π-helices. MKDSSP classifies the handedness (chirality) with the help of the parameter Vtor. Using this information leads to the classification of 1 residue long left-handed helices at the C-terminus of numerous right-handed helices (6,035 out of 6,136 (98.4 %) left-handed α-helices, 1,772 out of 2,048 (86.5 %) left-handed 3_{10}-helices, and 192 out of 196 (98.0 %) left-handed π-helices are 1 residue long).

Although the results of SCOT for this dataset are comparable to those of the two geometry-based tools, we find some striking differences (see Table 6.14, appendix for the assignment of left-handed helices by all methods for the dataset of Novotny and Kleywegt). ASSP does not classify three of the manually assigned helices and omits three helices additionally identified by DISICL. SCOT misses nine of the manually assigned left-handed helices due to insufficient overlaps of the hydrogen-bonded turns, but assigns two additional left-handed 3_{10}-helices together with DISICL. MKDSSP also finds most of the left-handed helices, but they are in general one residue shorter. Several additional helices were identified but we omitted them in the results due to their shortness (1–2 residues). Obviously, all four methods show discrepancies in the reliable classification of left-handed helices. Nonetheless, there are some considerations which reveal the benefits of the

SCOT-based assignment.



(a) Example of a left-handed α-helix partially composed of D-amino acids in 2q33B@pdb. The Purity for the helix assigned by SCOT is 0.887 left-handed α and 0.113 left-handed 3_{10}.

(b) Example of a left-handed 3_{10}-helix classified in the structure 2dvtA@pdb. The Purity for the helix assigned by SCOT is 0.714 left-handed 3_{10} and 0.286 left-handed α.

Figure 4.7: Examples of rarely assigned left-handed helices. Examples of left-handed helices which occur in protein structures available in the PDB. D-amino acids are underlined. The overlaps of the corresponding *normal* turns are given to illustrate the respective class assignments. DISICL does not assign left-handed classes (purple outline). This Figure is reproduced by permission of Bioinformatics (2019) [7].

The geometric descriptors of the left-handed α- and 3_{10}-helices assigned for the X-ray representatives dataset are the most stable as compared to those of left-handed helices assigned by ASSP (see Table 6.2, appendix for the geometric characteristics of left-handed helices). Additionally, the dihedral angles of the residues in SCOT-assigned helices are not as broadly distributed as for ASSP- and DISICL-assigned helices. The scaled B-factors suggest a higher degree of stability for the left-handed helices identified with the SCOT methodology. In contrast to the observations of Hollingsworth and co-workers [109, 112], we identified multiple left-handed α- and 3_{10}-helices which are more than three residues long. Two examples and their assignment by ASSP, DISICL, MKDSSP, and SCOT are given in Figure 4.7. The left-handed α-helix in Figure 4.7a forms due to the presence of D-amino acids which favor the formation of left-handed helical conformations. Although the average length of assigned left-handed helical conformations rarely exceeds four residues (α-helices are on average one residue longer than 3_{10}-helices as also observed for their right-handed counterparts), we argue that the left-handed helix category is a more recurrent and regular motif than previously anticipated.

Given the sparsity of left-handed helices in the X-ray representatives dataset applied here (8 and 37 left-handed α-helices and 4 and 108 left-handed 3_{10}-helices were identified by ASSP and SCOT, respectively while DISICL and MKDSSP classified altogether 6,534 and 8,407 often short left-handed helices), we searched the entire 2018 copy of the PDB for left-handed helices. To this end, we used SCOT as it is the most restrictive tool which incorporates hydrogen bond stabilization of helices to assign at least 3 residue long left-handed helices.

The set of proteins with at least four-residue long left-handed helices was subsequently used to generate a representative (non-redundant) set of protein structures. Although there is a significant

bias, the high consensus to ASSP and DISICL (see Table 4.6) and the improved recognition of left-handed α- and 3_{10}-helix classes using a hydrogen bonding criterion justify the use of SCOT for this purpose.

The resulting representative set (see Section 2.3.3) was subsequently processed with ASSP, DISICL, MKDSSP, and SCOT. The overall consensus between ASSP, DISICL, and SCOT is high whereas that of MKDSSP is comparatively low to all methods (see Table 4.6, and Tables 6.13a and 6.13b, appendix). This together with the differences with respect to the conformational parameters of the residues in left-handed helices (see Table 6.5, appendix) underlines the basic problem of using Vtor for helix handedness assignment, i.e., the overestimation of this angle in the irregular regions following the C-terminus of right-handed helices. Left-handed helix assignments by SCOT and DISICL are most similar. Differences occur in the class assignment by ASSP and SCOT leading to a low consensus for both left-handed helix classes. A huge proportion of ASSP-assigned α-helices are classified as left-handed 3_{10}-helices by our method based on the $r_i \rightarrow r_{i+3}$ hydrogen bonding pattern. Further class-specific differentiations between ASSP and SCOT assignments can be made (see Table 6.3, appendix for the geometric characteristics of the assigned helices). While the geometric parameters of SCOT-assigned left-handed α-helices are more robust, these of the ASSP-assigned 3_{10}-helices are more stable and show lower B-factors. In contrast, SCOT-assigned right- and left-handed 3_{10}-helices are characterized by higher scaled B-factors which is in compliance with the lower hydrogen bond energies of the underlying turn type (*normal-4 3*) (see Table 6.1, appendix). The dihedral angles of the SCOT-derived left-handed helix classes conform to their counterparts for the right-handed classes. In contrast, the dihedral angles of ASSP left-handed α-helical segments tend toward those of 3_{10}-helices.

H6+H11+H13	SCOT	ASSP	DISICL	MKDSSP
SCOT	1	0.6501	0.8534	0.3730
ASSP		1	0.6704	0.3004
DISICL			1	0.3761
MKDSSP				1

Table 4.6: Consensus of different SSAMs in the assignment of left-handed helices for the non-redundant set of structures with left-handed helices.

4.6.3.4 Polyproline II Helices

Another class of left-handed helices in proteins are PPII helices [113]. In contrast to other helix classes, these helices are characterized by an extended conformation and predominantly occurring Pro residues. Their functional roles in different biological processes have already been shown by different independent investigations [114, 115]. A repetitive hydrogen bonding pattern cannot be found for this helix class. SCOT uses the occurrence of repetitive *open-4 9* turns, which show average four-residue segment Cα–Cα distances of 7.8 Å, to assign PPII helices. ASSP, DISICL, and SEGNO assign PPII helices based on the protein geometry and/or dihedral angles. The consensus between SCOT and the three other methods is considerably low (below $20\,\%$). As already observed in other analyses [49, 50], the assignment of PPII helices is a challenging issue

and highly diverse assignments can be found. SCOT assignments are most similar to these of SEGNO (see Table 6.12e, appendix for the consensus). There are no two tools which resemble each other in the assignment of PPII helices. This can probably be attributed to the occurrence of PPII helices as irregular SSEs which are partially stabilized by water bridges [116] and hydrophobic interactions [117]. Nevertheless, the PPII helices of all methods are dominated by Pro residues. For the X-ray representatives dataset, ASSP, DISICL, SCOT, and SEGNO assigned 3,891, 35,971, 2,752, and 7,458 PPII helices, respectively. The fraction of residues assigned as part of PPII helices is 1.4 % for ASSP, 8.9 % for DISICL, 1.3 % for SCOT, and 2.9 % for SEGNO (see Table 4.4 for residue coverages). ASSP- and SEGNO-assigned PPII helices also show a significantly high occurrence of Lys residues. The stability of the geometric helical descriptors for our PPII helices is between that of DISICL, which shows a high variance within the helical parameters, and that of ASSP and SEGNO, which assign the geometrically most stable PPII helices with the lowest overall B-factors. The B-factors of residues involved in the different PPII helices are considerably higher than the average scaled B-factors for other backbone hydrogen bond-stabilized SSEs (see Table 6.2, appendix).

We analyzed the differences between the SSAMs in more detail. Many of the ASSP- and SEGNO-defined PPII helices overlap with (1,133/933) or are completely included in (110/70) our hydrogen bond-based assigned β-strands. SCOT assignments are designed to exclude any helices occurring within main-chain hydrogen bond-stabilized strand structures. Consequently, many of the geometrically assigned PPII helices are assigned as strands or completely omitted by SCOT. Enabling PPII assignment independent of strands leads to the assignment of 4,559 PPII helices. Compared to the 2,752 helices found previously, this is a huge increase. The insufficient differentiation between PPII helices and β-strands by geometry-based methods partially explains the highest mean B-factor of SCOT-assigned PPII for the dataset. The mean B-factor of all PPII helices decreases from 0.31 to 0.04 if overlapping β-strands and PPII helices are allowed. In consequence, SCOT assigns overall highly stable PPII helices. Intriguingly, these helices show the highest mean BDA with the highest variance. For PPII helices, no maximum four-residue segment Cα–Cα distance was defined to cut bent helices. PPII helices have an extended conformation with helical character. Consequently, the use of Cα–Cα distances for a splitting is difficult and we only split PPII helices with four-residue segment Cα–Cα distances below 7.45 Å (see Figure 3.9d for the Cα–Cα distance histogram). Although SCOT seems to be altogether less restrictive for PPII assignments than other methods when overlapping PPII helices and β-strands are allowed, the percentage of PPII residues is the lowest for our representative dataset. One reason is the exclusion of PPII helix assignment within strands. Another reason is the minimum required length of three residues per helix (as compared to DISICL).

4.6.4 Sheets

The most extended backbone conformation observed in proteins are β-strands (see Table 6.2, appendix for the geometric parameters of all SSE classes). They occur as isolated strands as well as in larger parallel or anti-parallel assemblies of strands (sheets). Our strand assignment is based on the analysis of the underlying hydrogen bonding patterns. In addition, turns are used to split

strands (see Section 3.3.3.3). In contrast to geometry-based methods, SCOT does not identify isolated strands.

The average length of SCOT-assigned strands is similar to that of strands assigned by MKDSSP, SEGNO, and STRIDE. ASSP and DISICL assign overall shorter strands (see Table 6.2, appendix). The dihedral angles of SCOT-assigned strands are in a narrow range although they were not used in the assignment procedure. The BDA distribution is broader than that of ASSP, DISICL, and SEGNO whose strand-assignments are characterized by the lowest average BDAs (see Figure 4.8).

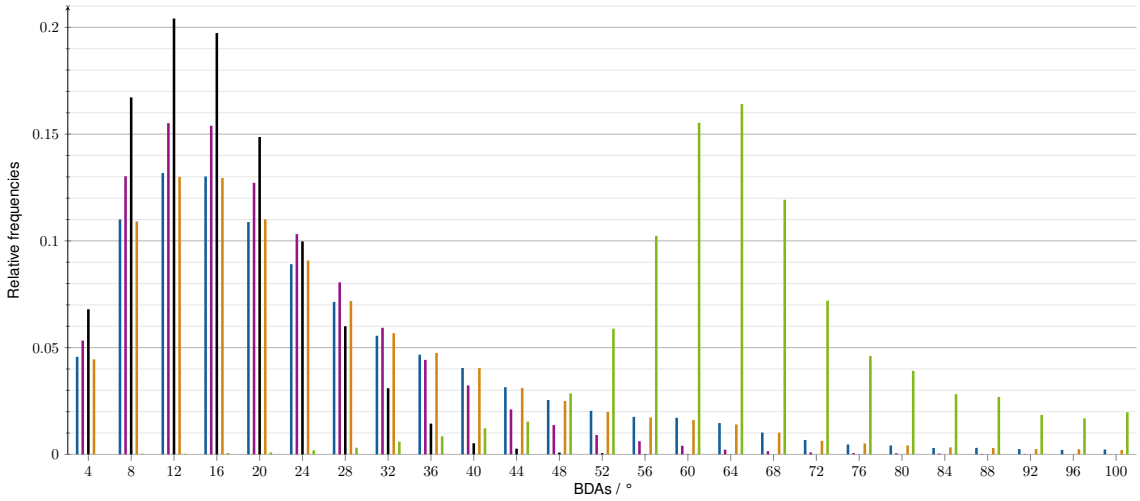


Figure 4.8: BDA histogram of the distances for β-strands assigned by different SSAMs. BDA histogram of the distances for β-strands assigned by SCOT (blue), $\text{SCOT}_{\text{kinked}}$ (purple), ASSP (black), and STRIDE (orange) are given together with the binned BDAs (in degrees) for the SCOT-assigned kink residues (green). This Figure is extracted from [7].

We additionally use four-residue segment Cα–Cα distances to account for highly bent strands and it is possible to use this information to split strands. We assign strand kinks which can be visualized and utilized in a strand post-processing step. This is achieved via analyzing the Cα–Cα distance distributions within strands. Obviously, a distance below $8.5\,\text{Å}$ is atypical for strands (see Figure 3.7 for the Cα–Cα distance histogram) and the residues within such four-residue segments are assigned as strand-kink-residues. An analysis of the BDAs at these residue positions shows high overall BDAs which are on average $59.69° \pm 18.82°$ (see Figure 4.8, appendix). A splitting at kink positions leads to a decrease in the mean BDAs from approximately $23.8°$ to $18.6°$ (see Table 6.2, appendix) which is similar to those of the geometry-based methods DISICL and SEGNO and below those of the hydrogen bond-based classifications. An analysis of the amino acid propensities at different kink positions shows a clear overrepresentation of Gly (and also Ser and Val) residues at the position of the kink whereas flanking regions include Ile, Leu, Val, and Cys (r_{k-1}) and Phe, Tyr, Val, Ile, and Cys (r_{k+1}) residues (see Table 4.8). Our strand kinks can be used to obtain geometrically uniform strands without influencing the residue preferences at the Ncap and the Ccap of strands (see Table 6.11, appendix). In contrast to the geometry-based methods, SCOT assignments focus on finding stable extended conformations as revealed by the significantly lower average scaled B-factor (see Table 6.2, appendix). The use of the kinks to

split strands leads to a geometrically more robust strand assignment without relying on purely geometric criteria. Nevertheless, we decided against an automated splitting of strands as the hydrogen bonding pattern is continuous and only the underlying geometry is irregular.

S0	SCOT	ASSP	DISICL	MKDSSP	PDB	SEGNO	STRIDE
SCOT	1	0.4788	0.5158	0.8982	0.8811	0.5916	0.8567
ASSP		1	0.5001	0.4859	0.4893	0.5439	0.4826
DISICL			1	0.5350	0.5410	0.6565	0.5431
MKDSSP				1	0.9755	0.6065	0.9252
PDB					1	0.6130	0.9067
SEGNO						1	0.6048
STRIDE							1

Table 4.7: Consensus of different SSAMs in the assignment of β-strands for the X-ray representatives dataset. The most similar methods to SCOT are highlighted in blue.

S0	r_{k-3}	r_{k-2}	r_{k-1}	r_k	r_{k+1}	r_{k+2}	r_{k+3}
Ala	0.8824	0.5283	0.6847	1.0215	0.8323	0.7113	0.7762
Cys	1.6903	0.5843	1.8363	0.9108	1.8572	1.0851	1.2938
Asp	0.8094	0.6038	0.2098	1.0401	0.4325	0.8436	0.9293
Glu	0.6165	1.4576	0.4148	0.9458	0.6774	1.0884	0.8144
Phe	1.4921	0.7493	1.0823	0.7728	1.5498	1.2680	1.2232
Gly	2.2651	0.6036	0.3250	1.4509	0.6799	0.8788	0.9352
His	0.6582	0.8806	0.3380	0.8541	0.7205	0.9962	0.7205
Ile	1.1005	1.0732	3.1242	1.0976	1.9737	1.5462	1.4734
Lys	0.6303	1.9415	0.4831	1.0253	0.5521	0.8419	0.7959
Leu	0.9718	0.6450	1.5972	0.8790	0.9913	0.8489	0.8433
Met	0.7564	0.7839	0.7701	0.6214	1.0177	0.9352	0.8664
Asn	1.0037	1.0097	0.3507	0.9904	0.6167	1.0521	0.9855
Pro	0.8573	0.6185	0.9929	0.3483	0.2659	0.1736	1.0309
Gln	0.4684	1.4191	0.4822	0.8419	0.8680	0.9231	0.7302
Arg	0.6882	1.7256	0.4788	1.0449	0.8528	0.9974	0.9825
Ser	0.6650	0.8229	0.5237	1.1674	0.9268	1.0515	0.9726
Thr	0.7435	1.4634	0.9740	0.9689	1.0634	1.1387	1.0917
Val	1.2254	1.1492	3.1287	1.1989	2.2550	1.4791	1.6314
Trp	1.7245	1.0013	0.4450	1.0950	1.2980	1.1867	1.1126
Tyr	1.1806	1.1509	0.7648	1.0714	1.2994	1.3366	0.9430
XXX	0.6721	0.5761	1.0561	0.7149	1.4401	1.0561	1.1521

Table 4.8: Conformational parameter for residues in SCOT-defined strand kink regions. Conformational parameter P as defined by the method of Chou and Fasman [25] for residues in SCOT-defined strand kink regions. This Table is extracted from [7].

The significant amino acid preferences for strands are highly similar for all methods (see Table 6.6, appendix). The only exception is DISICL, whose assigned strands show a high occurrence of Pro residues. The most similar method to SCOT with respect to strand assignment was MKDSSP with a consensus of 90 % (see Table 4.7). The geometry-based methods classify strands that are

assigned as PPII helices by SCOT. This is in line with the observation that ASSP, DISICL, and SEGNO find Pro as overrepresented residue at strand termini (see Table 6.11, appendix for the residue preferences at strand termini). This finding is further discussed in the following section.

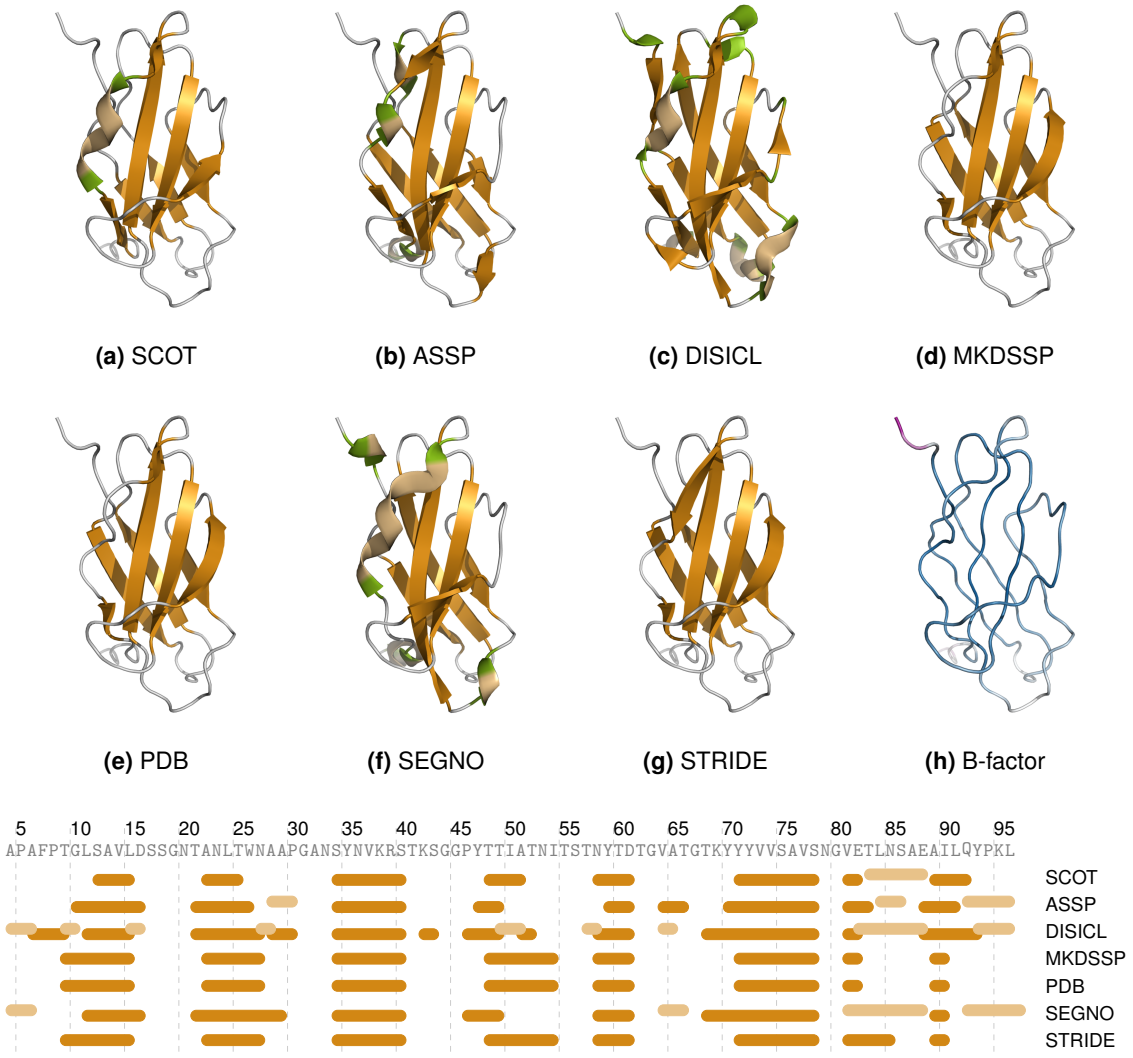
4.6.5 Disagreements in Assigning Extended Conformations

S0+H10	SCOT	ASSP	DISICL	MKDSSP	PDB	SEGNO	STRIDE
SCOT	1	0.5000	0.4845	0.8453	0.8326	0.5888	0.814
ASSP		1	0.4903	0.4868	0.4899	0.5770	0.4851
DISICL			1	0.4752	0.4802	0.6636	0.4848
MKDSSP				1	0.9778	0.5665	0.9326
PDB					1	0.5721	0.9154
SEGNO						1	0.5677
STRIDE							1

Table 4.9: Consensus of different SSAMs in the assignment of extended conformations for the X-ray representatives dataset.

The assignment of β-strands and PPII helices is highly different for ASSP, DISICL, SEGNO, and SCOT (see Table 4.9, and Tables 6.12e and 6.12f, appendix for the consensus of these four methods). Partially, this can be attributed to the hydrogen bond-based β-sheet assignment by SCOT. A hierarchical assignment was applied to avoid the overlap between strands and PPII helices. Without this restriction, SCOT assigns approximately twice as many PPII segments. We compared the PPII helices assigned by the four tools to analyze this observation in more detail. Figure 4.9 gives one example of the highly different per-residue assignments by the methods analyzed herein. The hydrogen bond-based strands assigned by MKDSSP, SCOT, and STRIDE overlap to a high degree whereas ASSP and DISICL define further isolated β-strand structures which partially overlap with PPII helices assigned by other geometry-based methods. Altogether, 24 %, 48 %, and 36 % of the SCOT PPII residues are classified as strand residues by ASSP, DISICL, and SEGNO, respectively. Vice versa, many of the PPII helices assigned by these three tools correspond to SCOT-assigned strand structures. This explains the overall lower scaled B-factor of the PPII helices assigned by geometry-based tools. An analysis of the PPII residues of the different methods shows that there is no clear differentiation between extended PPII helix and strand conformation. This can be partially attributed to the similarities with respect to the dihedral angles and geometric parameters (see Table 6.2, appendix) which limits a purely geometry-based classification.

We argue that our hydrogen bond-based β-sheet classification and exclusion of PPII helices in those residue segments enables a unique assignment of PPIIs. Further analyses will be necessary to reliably and uniquely assign protein segments as PPII helices. A preliminary conclusion is that both SSE classes cannot be distinguished easily and they should both better be referred to as extended conformations.



(a) SCOT **(b)** ASSP **(c)** DISICL **(d)** MKDSSP

(e) PDB **(f)** SEGNO **(g)** STRIDE **(h)** B-factor

(i) Comparison of different assignments.

Figure 4.9: Strand and PPII assignments by different SSAMs. The assignments are given for 3mpcA@pdb. β-strands are represented in orange while PPII assignments are highlighted in light orange. The three-dimensional representations at the top were generated using PyMOL [32]. MKDSSP, PDB, and STRIDE do not support the assignment of PPII helices. The ribbon structure (h) is colored according to the B-factor of the proteins' Cα atoms with a gradient ranging from purple for the highest to white to blue for the lowest value. This Figure is reproduced by permission of Bioinformatics (2019) [7].

4.6.6 Rare Helix Classes

Two further helix classes are discussed in this paragraph. The first one is the 2.2_7-helix which is rarely occurring in proteins. SHAFT assigns 2.2_7-helices based on inverse γ-turns (*normal-3 1*) and assigns them as right-handed γ-helices (helix class 4 according to the PDB classification). Intriguingly, most of those SHAFT-assigned helices can be found within β-strands with mean φ and ψ dihedral angles of $-103.6°$ and $120.2°$. Their assignment can be attributed to the non-

restrictiveness of the hydrogen bond assignment according to Kabsch and Sander [22]. Our investigations of the hydrogen bonding energies and geometry led to the conclusion that the hydrogen-bonded *normal-3* turns identified by the turn classification of Koch and Klebe [88] are partially inappropriate due to geometry highly deviations from the optimum for hydrogen bonding interactions [118, 119]. The original definition of so-called γ-helices relates back to the first secondary structure investigations of Pauling and co-workers [35] who coined the term γ-helix as an alternative to the α-helix [120]. To prevent any confusion, we will name helices composed of γ-turns as 2.2_7-helices according to Donohue [121].

This helix class has been reported for a crystalline peptide structure [122] and was shown to exist in globular protein structures [123, 124]. Its left-handed form is characterized by φ- and ψ-angles of approximately $-80°$ and $60°$ [125]. These dihedral angles correspond to the dihedral angle space of *normal-3 1* turns (corresponding to inverse γ-turns, mean φ = $-82.3°$, ψ = $61.1°$). *Normal-3 2* turns (normal γ-turn, mean φ = $74.3°$, ψ = $-52.0°$) can be used to assign right-handed 2.2_7-helices. Consequently, we used these turn classes to assign 2.2_7-helices whenever we find overlapping turns of this class with a length of at least 2. This methodology did not detect 2.2_7-helices in the complete X-ray representatives dataset, but 2.2_7-helices were classified for some protein structures in the PDB (copy 2018). We used the 2.2_7 assignments for the complete PDB to derive some helix characteristics. In contrast to the γ-helices assigned by SHAFT, the dihedral angles of the SCOT-defined 2.2_7-helices (φ = $-79.6° \pm 26.4°$, ψ = $45.2° \pm 32.8°$) are similar to those assigned earlier by Ramachandran and Chandrasekaran [125]. Rather than a helical character, those elements as assigned by SCOT are characterized by an extended conformation and should be recognized as ribbon structures. Table 4.10 summarizes all occurrences of 3 residue long left- and right-handed 2.2_7-helices as assigned by SCOT. Both helix classes are highly underrepresented. In contrast to the left-handed 2.2_7-helices, the right-handed class can only be found at highly flexible N-termini of NMR ensemble structures and their meaning as a regular SSE class is questionable. However, very short left-handed 2.2_7-helices can be observed in the PDB and are worth further investigations as they are located in stable parts of the proteins. We observe that residues constituting the helices are located in stable parts of the protein but not identified in all known structures of the respective proteins. A less restrictive hydrogen bond definition would lead to their identification in all structures known for the proteins. Nevertheless, we did not change our hydrogen bond criterion for this helix type to circumvent the problems observed for SHAFT (γ-helices within strands).

SCOT is not able to classify the so-called ω-helix in proteins which was initially characterized for synthetic polypeptides [126, 127]. This helix class was investigated by Enkhbayar and co-workers [128]. The authors argue that ω-helices can be found within proteins. The average φ, ψ, and ω angles of $-75°$, $-34°$, and $175°$ of the characterized ω-helices lie well within the standard deviations of those of SCOT-assigned α-helices and a differentiation is, therefore, infeasible (see Table 14, appendix for the dihedral angles of α-helices).

Left-handed 2.2_7 (*normal-3 1*)

1xgo	(5)	P56218	(methionine aminopeptidase)	His173-Asn175	(chain A)
1ysw	(3)	P10415	(Bcl-2)	Asp32-Val34	(chain A)
2o21	(3)	P10415	(Bcl-2)	Asp32-Val34	(chain A)
1zr4	(2)	P03012	(transposon gamma-delta resolvase)	Gln13-Ser15	(chains A, D, E)
2bud	(1)	O02193	(males-absent on the first protein)	Leu369-Gln371	(chain A)
2j28	(168)	P0A7M6	(50S ribosomal protein L29)	Gln39-His41	(chain X)
2kz1	(7)	P01563	(interferon alpha-2)	Gln191-Val193	(chain B)
2qts	(15)	Q1XA76	(acid-sensing ion channel)	Thr295-Asp297	(chain F)
4n43	(10)	Q9WPJ0	(capsid protein VP3)	Val55-Asn57	(chain C)
5oun	(1)	Q12464	(RuvB-like protein 2)	Arg220-Val222	(chain A)

Right-handed 2.2_7 (*normal-3 2*)

1njq	(2)	Q38895	(superman protein)	Ala33-Leu35	(chain A)
2v1n	(1)	O60870	(protein kin homolog)	Leu5-Leu7	(chain A)

Table 4.10: Examples of rarely SCOT-assigned 2.2_7-helices. Examples of rarely assigned left-handed (inverse γ-turns) and right-handed (normal γ-turns) 2.2_7-helices assigned by SCOT. Numbers in parentheses indicate the number of structures of the protein in the entire PDB (2018). This Table is extracted from [7].

4.6.7 Impact of Structure Quality on SSE Assignments

For the validation and evaluation of SSAMs, it is interesting to assess the tools' ability to correctly assign SSEs independent of the structure quality. This evaluation is especially important for the analysis of SCOT as the more restrictive hydrogen bond criterion might lead to failures in identifying the appropriate SSEs in poorly resolved crystal structures. A correlation between resolution and SSE assignment was already observed for other SSAMs [129]. We used a modified version of the dataset of Konagurthu and co-workers [40] (see Section 2.3.4) to calculate the consensus in assigned SSEs for pairs of X-ray structures with high and low resolution. Table 4.11 presents the results for the consensus of right-handed helices and extended conformations for all high and low resolution pairs (for SHAFT only right-handed helices are considered as the `SHEET` information is extracted from the original PDB file). The consensus results are sorted according to the resolution of the low resolution structure. For SCOT and STRIDE, a distinct correlation between resolution and consensus is observed (decreasing consensus with decreasing resolution). The geometry-based methods ASSP, DISICL, and SEGNO together with SHAFT show low consensus values for structures with poor as well as high resolutions pointing toward general inconsistencies of these tools.

The low consensus for SCOT and SHAFT for the structure pair 4k20@pdb and 5cna@pdb results from the fact that the protein mainly consists of β-strands. The two short helices (3 to 6 residues) in the structure bias the results if the β-strands are not considered. Obviously, (completely or partially) hydrogen bond-based assignment methods except for SHAFT are the most robust ones regarding structure quality.

Based on the mean consensus for this dataset, we can state that the impact of structure quality

on SCOT-defined SSEs is low and comparable to that of STRIDE (which is the most quality-independent method herein) and MKDSSP. Obviously, (completely or partially) hydrogen bond-based assignment methods are the more robust ones while geometric SSE assignment criteria are more sensitive toward quality differences in terms of X-ray structure resolution.

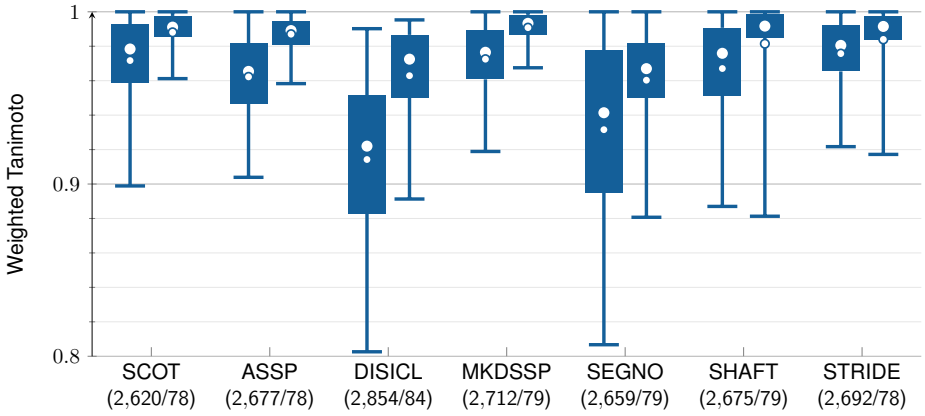
UniProt	PDB-ID (L/H)		SCOT	ASSP	DISICL	MKDSSP	SEGNO	STRIDE	SCOT*	SHAFT
P17802	1wef	1kg2	0.986	0.976	0.987	0.996	0.918	0.997	0.973	0.983
Q9REU3	2awc	3agt	0.987	0.964	0.991	0.967	1.000	0.982	0.973	0.943
Q9AMP1	3aro	3arx	0.997	0.946	0.946	0.984	0.917	0.979	0.994	0.962
P95339	3zow	4jcg	0.988	0.932	0.945	0.939	0.941	0.976	0.975	0.891
P24295	1aup	1bgv	0.993	0.964	0.935	0.986	0.763	0.991	0.987	0.974
P62157	1yru	1fw4	0.988	0.988	0.951	0.961	0.855	0.988	0.976	1.000
P32396	1ld3	1doz	0.994	0.975	0.958	0.981	0.885	0.991	0.988	1.000
P02213	1nwn	3sdh	0.996	0.992	0.982	0.980	0.927	0.998	0.992	1.000
P01009	1psi	3ne4	0.983	0.975	0.897	0.973	0.750	0.975	0.966	1.000
P10933	4af6	3mhp	0.972	0.967	0.908	0.968	0.852	0.991	0.944	0.928
P00390	1grh	3grs	1.000	0.977	0.986	0.993	0.987	0.997	1.000	0.970
P00512	1mto	4i7e	0.956	0.938	0.879	0.934	0.638	0.955	0.912	0.879
Q9WFX3	4gxu	4gxv	1.000	0.941	0.859	0.838	0.864	1.000	1.000	0.756
P02866	4k20	5cna	0.615	0.643	0.659	1.000	0.875	0.958	0.231	0.227
P00698	1bhz	2zq3	0.945	0.950	0.918	0.943	0.960	0.933	0.891	0.943
mean			0.960	0.942	0.920	0.963	0.875	0.981	0.920	0.897
σ			0.097	0.085	0.083	0.040	0.097	0.019	0.193	0.196

Table 4.11: The impact of structure quality on the assignments by different SSAMs. Given is the consensus of right-handed helices and extended conformations for structure pairs with high and low resolution together with the corresponding UniProt accession (UniProt) and PDB-IDs of the structures and the means and the standard deviations (σ) over all structures. The structure pairs are sorted with ascending difference between the resolution of the low and the high quality structures. For SCOT* and SHAFT, assignments solely right-handed helices were considered. This Table is reproduced by permission of Bioinformatics (2019) [7].

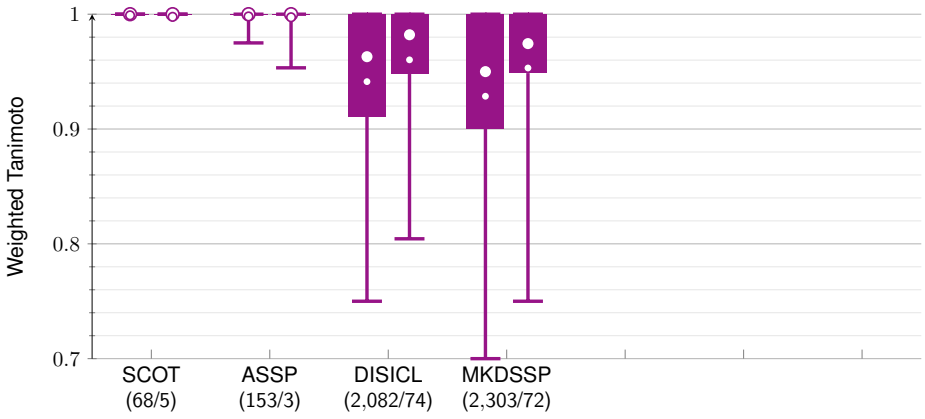
4.6.8 Consistency of Secondary Structure Element Assignments

Having discussed the geometric consistency of SCOT, we now take a look at the consistency of the assigned SSEs within structural ensembles. This issue was already discussed for DSSP [130] and consequently, DSSPcont was introduced 15 years ago as a more robust assignment method [58]. One possibility to assess the impact of protein flexibility on the outcome of SSAMs is the analysis of assignments for multiple models derived from NMR solution structures. Our NMR ensemble dataset comprises 2,856 ensembles of unrelated proteins. We calculated the weighted Tanimoto coefficient per ensemble to assess the robustness of the secondary structure assignments. The boxplots summarizing the consistency for right-handed helices, left-handed helices, and extended conformations are given in Figure 4.10.

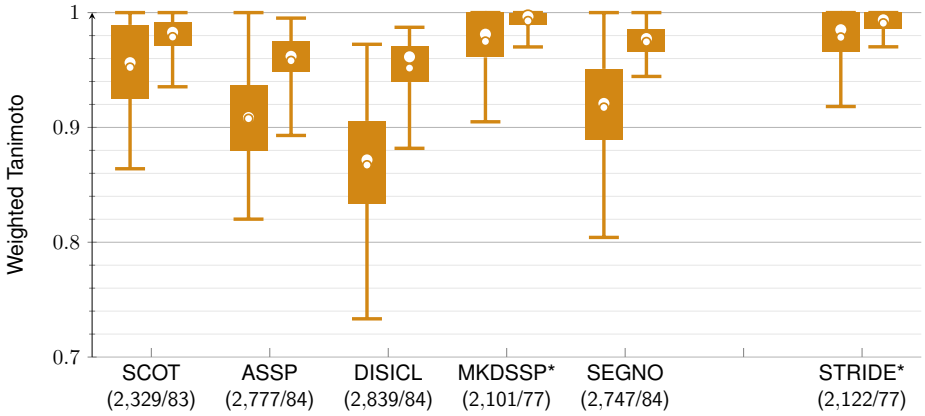
With respect to the mean weighted Tanimoto coefficient, STRIDE assigns the most consistent α-helices of all methods discussed herein followed by MKDSSP, SHAFT, and SCOT (see Figure 6.2a,



(a) Right-handed helices



(b) Left-handed helices



(c) Extended conformations (PPII helices and β-sheets), * no PPII helices classified/supported by the SSAM

Figure 4.10: Boxplots showing the overall consistency of different SSAMs based on the weighted Tanimoto coefficient. Boxplots showing the overall consistency of different SSAMs based on the weighted Tanimoto coefficient for the NMR (left, 2,856 ensembles) and X-ray (right, 84 ensembles) ensembles datasets. The median is indicated by a big and the mean by a small white dot. The numbers ensembles in which SSEs were classified by each SSAM are given in parentheses. Outliers were omitted in favor of a concise visualization. This Figure is reproduced by permission of Bioinformatics (2019) [7].

appendix for the boxplots of the α-helix consistency). A similar trend can be observed for π-helices (see Figure 6.2b, appendix for π-helix consistency boxplots). However, the overall differences are negligible. SCOT, SHAFT, and STRIDE classify the most consistent π-helices although the number of models including this helix category/class is significantly higher for SCOT. For 3_{10}-helices, SEGNO, MKDSSP, STRIDE, and SCOT give the highest mean weighted Tanimoto coefficient (see Figure 6.2c, appendix for 3_{10}-helix consistency boxplots). The left-handed helix assignments by ASSP and SCOT (see Figures 6.2e and 6.2g, appendix for left-handed helix consistency boxplots) are comparable for the NMR ensembles and more robust than those of DISICL and MKDSSP. Left-handed α-helix assignments by ASSP are slightly less consistent than those of SCOT. Probably, the geometry-based left-handed assignment tends to over-emphasize geometry over stability, therefore, lacking some robustness in this evaluation. Altogether, the consistent assignment of left-handed helices underpins their stable nature and meaning for protein structures, i.e., they are no artifacts randomly observed in protein structures. The robustness of the 2.2_7-helix assignments by SCOT and SHAFT was also analyzed. Due to the non-restrictive hydrogen bond criterion, SHAFT identifies left-handed 2.2_7-helices in $2{,}107$ of all $2{,}856$ ensembles with a mean weighted Tanimoto coefficient of 0.89 which is considerably lower than that for the 2.2_7-helices assigned by SCOT in 19 ensembles (see Figure 6.2f, appendix for 2.2_7-helix consistency boxplots). If this secondary structure class is assigned, it is classified consistently and reliably with SCOT. For mixed helices, we observe a consistent assignment by SCOT whereas mixed helices were assigned incoherently with SEGNO (see Figure 6.2d, appendix for mixed helix consistency boxplots). This can be attributed to the fact that SEGNO mixed helices include α-, 3_{10}-, and π-helices whereas SCOT mixed helices comprise only α- and 3_{10}-helices which are solely assigned in cases of equal contributions of both underlying turn types. Intriguingly, SEGNO mixed helices occur in nearly $70\,\%$ of all proteins. SCOT assigns mixed helices in only $14\,\%$ of the ensembles. The most robust PPII assignments are obtained with ASSP, SCOT, and SEGNO (see Figure 6.2h, appendix for PPII helix consistency boxplots). Here, the robustness of SCOT assignments cannot be explained based on the stable hydrogen bonding patterns observed for other SSEs. In contrast the requirement of a consecutive pattern of *normal-4 9* turns, which rarely occurs randomly in coil conformations, is key to the robust assignment.

The strand assignments by STRIDE, MKDSSP, and SCOT are the most consistent (see Figure 6.2i, appendix for strand consistency boxplots) whereas the mean weighted Tanimoto coefficient for ASSP and DISICL is below 0.9 for this dataset. This again underlines the main difference between geometry- and hydrogen bond-based assignments.

A major drawback in using NMR ensemble data is the difficulty in assessing the quality of the structural models. Therefore, we used a second dataset of X-ray structures to validate the general conclusions drawn from our first analysis. Similar trends are observed for this dataset (see Figures 6.2a to 6.2i, appendix for the consistency boxplots for the X-ray ensembles dataset). The consistency of SCOT-assigned 3_{10}-helices for the X-ray structures is worse than for the NMR structures, but not very different from these of the methods with the most consistent assignments. Moreover, MKDSSP seems to be more consistent for the X-ray ensembles whereas STRIDE showed the highest consistency for the NMR ensembles. Similarly, SEGNO assignments were more consistent for the NMR- than for the X-ray structures.

Concerning the consistency of right-handed and left-handed helices as well as extended conformations, we see an overall high consistency of SCOT-assigned SSEs. Although SCOT applies hydrogen bonding patterns and geometric criteria, it shares the robustness of MKDSSP and STRIDE as mainly hydrogen bond-based methods. These data underpin the benefits of hydrogen bond-based assignment methods which not only consider the backbone geometry (a snapshot in protein crystallography), but also the stability of backbone segments. Strikingly, the also hydrogen bond-based method SHAFT is less consistent which can only be attributed to the inconsistent helix terminus assignment (see Tables 6.7 to 6.10, appendix for the conformational parameter of SSE terminal residues). Additionally, an extension by one residue for the Ncap and the Ccap in comparison to all other methods was observed. Including geometry and dihedral angle data as realized by STRIDE and SCOT improves the geometric regularity of SSEs without neglecting backbone flexibility. This underlines the high-wire act of obtaining consistent SSE assignments, i.e., assignments that tolerate minor spatial fluctuations in the protein backbone maintaining stable geometric parameters for the SSEs.

When the methods are ranked according to the mean weighted Tanimoto coefficient per SSE class, SCOT assignments yield ranks 1 to 4 for the NMR ensembles dataset and ranks 1 to 3 for the X-ray ensembles dataset, with 3_{10}-helices as the only exception (rank 5). Even rare SSE assignments were shown to have a high consistency, underlining the reliability of the SCOT-assigned SSE classes.

Summarizing this section, we observe a distinct trend in SSE assignment consistency. While hydrogen bond-based methods lead to the most consistent assignments, the residue-based SSE classification highly fluctuates within structural ensembles for geometry-based methods. SCOT assigns helices and strands based on hydrogen bond as well as geometric criteria. It classifies different SSE classes with a reliable consistency. The average weighted Tanimoto coefficient is located in between that of geometry and hydrogen bond-based methods. Altogether, MKDSSP, SCOT, and STRIDE lead to highly consistent assignments.

4.6.9 Impact of Secondary Structure Assignments on Alignment Quality

Finally, we used the SSAMs to discuss their impact on the SSE-based alignment of protein structures. We evaluated the ability of two different structural alignment tools to find similarities between proteins of one CATH superfamily and different sequence clusters, and to match proteins with the same CATH topology, but different superfamily classes. The tools' results were investigated for the secondary structure assignments obtained using all SSAMs.

The first study applies the UCSF Chimera MatchMaker [100] method. A pairwise sequence alignment of two protein structures is achieved using both, residue similarity and secondary structure similarity (helix, strand, other). The impact of secondary structure information on the initial sequence alignment can be adjusted according to the nature of the dataset. We evaluated the secondary structure contribution to the initial sequence alignment from 0.1 to 1 in 0.1 steps. The contribution leading to a matching of all pairs and the lowest root-mean-square-deviation

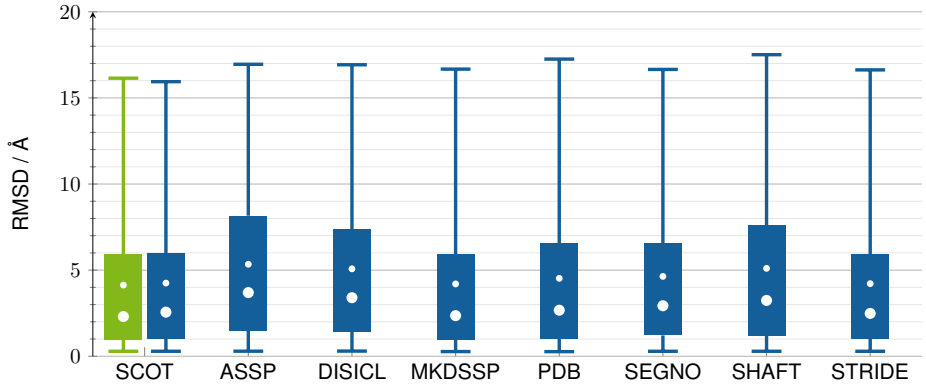


Figure 4.11: Boxplots for the RMSD values (Cα atoms) of the UCSF Chimera MatchMaker [100] superpositions of protein pairs with the same CATH superfamily. The impact of different SSE assignments by seven SSAMs and the SSEs of the original PDB file were analyzed. The SSE contribution to the initial sequence alignment was set to 0.8. The results for SCOT with split helices and strands at kink positions and without π-helices are shown in green. Outliers were omitted in favor of a concise visualization.

(RMSD) of matched residues (without iterations) was applied. An optimum contribution of 0.8 was assessed for the CATH superfamily dataset. It is noteworthy that the method's ability is restricted to sequential structure alignments. Therefore, we only analyzed the performance for protein pairs belonging to the same superfamily. Figure 4.11 depicts the boxplots of the RMSD values obtained for the protein pairs of the dataset using different SSAMs. We observe lower RMSD values for MKDSSP, SCOT, SEGNO, and STRIDE whereas ASSP-, DISICL-, and SHAFT-based matches are characterized by higher RMSD values for comparable alignment lengths. To further evaluate the suitability of the different assignments for a sequence- and SSE-guided structure alignment, we counted the number of cases in which one assignment leads to an RMSD at least 5 Å below that based on the other methods. In on average 12 cases for MKDSSP and SCOT and 13 cases for STRIDE, another assignment method is the better choice to obtain an accurate alignment in terms of RMSD. This average rises to 27, 31, 47, 58, and 64 for SEGNO, the PDB assignment, DISICL, SHAFT, and ASSP, respectively. On average 42 pairs for MKDSSP and 41 pairs for SCOT and STRIDE are matched with a significantly lower RMSD than observed for any other assignment method. Consequently, SCOT is a suitable alternative to MKDSSP and STRIDE for the use in sequence- and SSE-guided structure alignments.

As the UCSF Chimera MatchMaker solely uses SSE information for the initial sequence-based alignment, we applied LOCK2 [101] to assess the impact of secondary structure definitions on a tool that uses a vector-based SSE representation to structurally align proteins. The method generates alignments and reports the RMSD values, a score (which can be normalized to reflect the per-SSE-score), and the fraction of matched (aligned) SSEs. We analyzed different versions of SCOT SSE assignments. Besides the original assignment, we omitted π-helices from the comparisons as they constitute rare SSEs which frequently overlap with other right-handed helices. Finally, the SCOT helices and strands were split according to the detected kinks. All of these changes led to improved alignments with an overall lower RMSD and higher scores per matched SSE pair retaining the overall fraction of matched SSEs. A small increase in the fraction of matched

SSEs was observed for the CATH topology dataset (see Table 4.12 for the LOCK2 output for different SCOT options) which can be attributed to the splitting procedure. These parameters did also not influence the outcome for the MatchMaker comparisons (see Figure 4.11).

Topology		**Original**	**W/O π-helices**	**W/O π-helices, with split strands at kink positions**	**W/O π-helices, with split strands and helices at kink positions**
RMSD	mean	2.4461	2.4374	2.4333	2.4357
	σ	0.4051	0.4101	0.4124	0.4114
FASSE	mean	0.7027	0.7036	0.7038	0.7008
	σ	0.1787	0.1783	0.1783	0.1784
SCORE	mean	24.3128	24.5210	24.6209	24.7068
	σ	4.8349	4.9831	5.0478	4.9862
total failed		2	7	5	3
Superfamily					
RMSD	mean	1.3095	1.3030	1.2998	1.2994
	σ	0.7691	0.7621	0.7583	0.7578
FASSE	mean	0.9234	0.9232	0.9232	0.9226
	σ	0.1239	0.1229	0.1226	0.1231
SCORE	mean	32.4865	32.6869	32.7790	32.7718
	σ	4.8837	4.7569	4.6889	4.6163
total failed		7	21	14	7

Table 4.12: SCOT assignment optimization for a vector-based structural alignment of CATH topology and CATH superfamily pairs with LOCK2. Mean and standard deviation of RMSD / Å, fraction of aligned SSEs (FASSE), and per-SSE-score for LOCK2 alignments of CATH topology and CATH superfamily pairs are given for SSE assignments with different versions of SCOT. Additionally, the number of pairs which could not be compared by LOCK2 are given. This Table is extracted from [7].

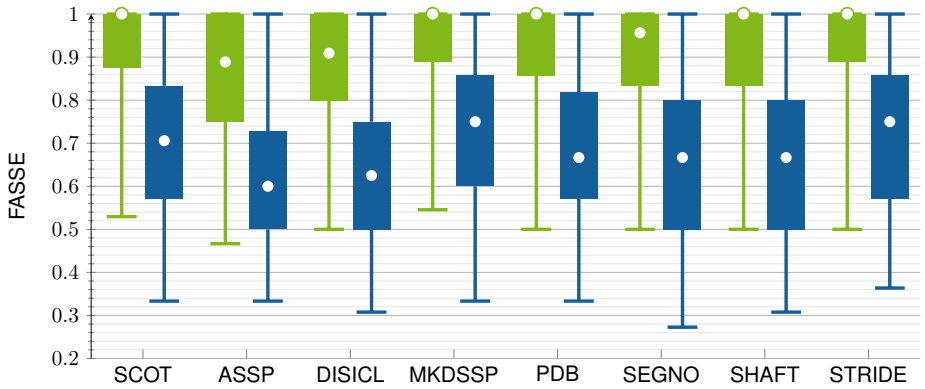


Figure 4.12: Boxplots for the fraction of aligned SSEs (FASSE) with LOCK2 for the CATH topology (blue) and the CATH superfamily (green) datasets. Given are the distributions of the fraction of aligned SSEs using different SSAMs. Outliers were omitted in favor of a concise visualization. This Figure is extracted from [7].

Next, we analyzed which SSAM is best suited for an SSE vector-based structure alignment. To

this end, we compared the LOCK2 normalized scores per SSE using different SSAMs. The results for the topology dataset are given in Table 4.13. SCOT assignments led to the best scores for the largest number of matches. Nevertheless, this finding might be insignificant due to only minor score differences. Consequently, we calculated the differences in the scores per match obtained for the different SSAMs. For per-SSE-score differences above 5, the SSAM with the lower score was declared to be outperformed. ASSP, DISICL, SEGNO, and SHAFT are most frequently outperformed by other methods (for 9.7 % to 10.9 % of all pairs). This is in line with the finding that a lower fraction of SSEs can be matched for these assignments (see Figure 4.12 for boxplots of the fraction of aligned SSEs with LOCK2) which hampers the reliable superposition of some topology pairs based on the vectorized helices and strands. In contrast, MKDSSP, SCOT, and STRIDE assignments lead to a high fraction of matched SSEs per topology pair. They are on average only rarely outperformed by other assignment methods. Nevertheless, there is one notable difference between the three methods. The number of cases where MKDSSP and STRIDE are outperformed by the geometry-based methods is significantly higher than that for SCOT. In other words, our combined geometry- and hydrogen bond-based approach offers a promising compromise between both approaches and enables the reliable SSE-based comparisons of similar CATH domains. This holds true for both, the CATH topology as well as the CATH superfamily pairs (see Table 4.13).

Obviously, the definition of not only SSE types provided by MKDSSP and STRIDE, but also PPII helices and left-handed helices can be used to obtain trustworthy alignments of related domain pairs. Examples are given in Figures 4.13 and 4.14 (alignments of two CATH topology and superfamily domain pairs using different SSE assignments) which underline the benefits of SCOT assignments. LOCK2 alignments which led to good scores with geometry-based, but not hydrogen bond-based SSE assignments and vice versa, showed high scores using SCOT-assigned SSEs.

In summary, SCOT provides reliable SSE assignments to guarantee for good alignment quality when applied for SSE-based sequence-guided and structure-guided protein structure alignments. The inclusion of rare helix classes does not hamper the alignment and might even lead to the identification of yet undiscovered similarities between proteins.

4.6.10 Runtime and Memory Consumption

We analyzed the Geometry (see Section 4.3.2.1) and the Consistency (see Section 4.3.2.4) Observer with respect to their memory and runtime consumptions. The memory footprint of both Observers was not quantifiable (0.0 %) by the use of the command `top`. As SNOT is not parallelized, all runtimes were achieved using a single thread.

The runtime of the Geometry Observer was evaluated with the help of the X-ray representatives (see Section 2.3.2) dataset containing 3,597 single protein chains. All protein chain files were stored in separate files of a total size of 1.1 GB. The runtime of the Geometry Observer was 1 min 38 s using default settings. The CPU usage was 80 % on average. This indicates that the file I/O dominates the runtime. We performed the same analysis but switched from the three parallel hard drives (RAID level 0) to a single solid-state drive (SSD). The runtime decreased to 47 s and

Score at least 5 higher →

Topology	SCOT	ASSP	DISICL	MKDSSP	PDB	SEGNO	SHAFT	STRIDE	Mean	Best	Score at least 5 lower ↓
SCOT	0	59	58	38	51	50	53	46	44.375	87	
ASSP	23	0	41	36	46	37	48	39	33.750	49	
DISICL	34	45	0	39	56	37	59	42	39.000	55	
MKDSSP	29	55	49	0	34	49	46	33	36.875	52	
PDB	21	46	39	14	0	40	23	28	26.375	35	
SEGNO	26	40	36	33	48	0	49	40	34.000	41	
SHAFT	26	46	46	21	22	47	0	31	29.875	40	
STRIDE	30	49	49	26	38	45	42	0	34.875	42	
Mean	23.625	42.500	39.750	25.875	36.875	38.125	40.000	32.375			
Worst	30	71	69	28	52	63	57	42			

Score at least 5 higher →

Superfam	SCOT	ASSP	DISICL	MKDSSP	PDB	SEGNO	SHAFT	STRIDE	Mean	Best	Score at least 5 lower ↓
SCOT	0	87	86	37	58	49	68	62	55.875	265	
ASSP	25	0	64	46	61	38	60	62	44.500	95	
DISICL	24	59	0	33	56	30	60	43	38.125	102	
MKDSSP	13	78	78	0	27	41	41	43	40.125	141	
PDB	20	71	77	17	0	43	24	47	37.375	121	
SEGNO	19	60	57	32	54	0	47	51	40.000	147	
SHAFT	31	70	81	27	24	48	0	46	40.875	164	
STRIDE	22	82	59	26	44	43	43	0	39.875	130	
Mean	19.250	63.375	62.750	27.250	40.500	36.500	42.875	44.250			
Worst	64	330	247	69	123	116	111	128			

Table 4.13: The impact of different SSAMs on the per-SSE-scores of LOCK2 alignments for the CATH topology and CATH superfamily (Superfam) pairs. We counted the number of times the per-SSE-score obtained with one method was at least 5 higher (columns) or lower (rows) than that of the LOCK2 alignment using different SSAMs. Additionally, we counted the number of times an assignment method led to the best (columns) and worst (rows) alignments in terms of the per-SSE-scores. This Table is reproduced by permission of Bioinformatics (2019) [7].

the CPU usage increased to 97 % on average. Both underlines that the file I/O is a major bottleneck even when using the SSD.

The same was observed for the Consistency Observer. We analyzed the runtime for the NMR ensembles dataset (see Section 2.3.5) consisting of 2,856 ensembles with a total of 56,189 models for each of the seven SSAMs (ASSP, DISICL, MKDSSP, SCOT, SEGNO, SHAFT, and STRIDE). The total required file space for all files (393,323) was 63 GB. The runtime with default settings on the hard drives took 2 h 57 min 57 s. However, the speed-up when using an SSD is much higher here. The use of an SSD led to a runtime of 47 min 14 s. The CPU usages were similar compared to the usages observed in the runtime analysis of the Geometry Observer.

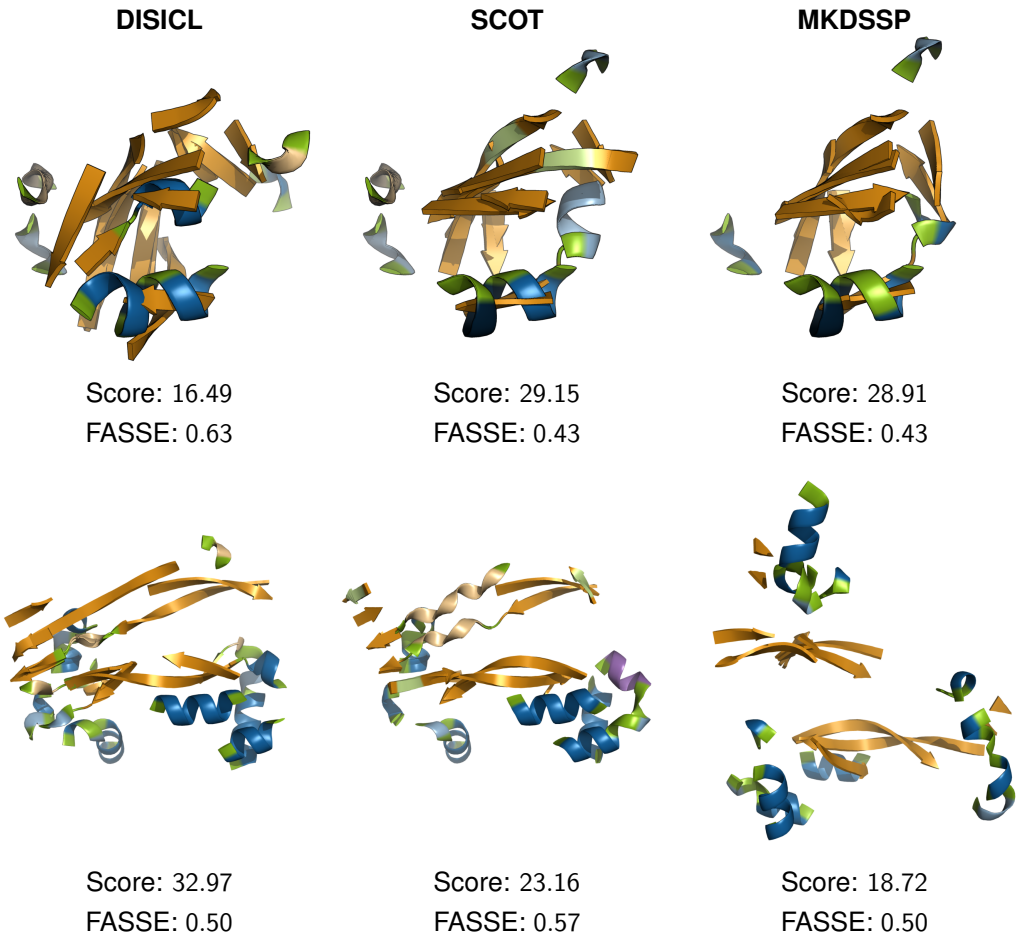


Figure 4.13: LOCK2 alignments of two CATH topology domain pairs. LOCK2 alignments of the CATH topology domain pairs of Barwin-like endoglucanases (5b6cA02@cath (3.10.330.10) and 1o54A01@cath (3.10.330.20)) (top) and Penicillin-binding protein 2a, domain 2 (4mnrA01@cath (3.90.1310.10) and 3oc2A01@cath (3.90.1310.30)) (bottom). The alignments are given for the SSE classifications with DISICL, SCOT, and MKDSSP together with the corresponding scores for the matched SSEs and fractions of aligned SSEs (FASSE). This Figure is reproduced by permission of Bioinformatics (2019) [7].

4.7 Discussion

We introduce SNOT as a novel and comprehensive tool for the in-depth evaluation, analysis, and comparison of secondary structure assignments. It provides six different Observers for the analysis of geometric properties, the consistency with respect to conformational flexibility, the sequence coverage, the consensus of two secondary structure assignments, the overlaps of different SSE types and classes, and the statistics on underlying residue types. Apart from the relevance of all of these criteria, the major advantage of SNOT is their fusion in a single and intuitively to use tool. In addition, it is not limited to specific SSE types or classes. Instead, the introduction of additional SSE type and class groups (e.g., right-handed helices) supports the comparison of SSAMs with differences in their supported SSE types and classes.

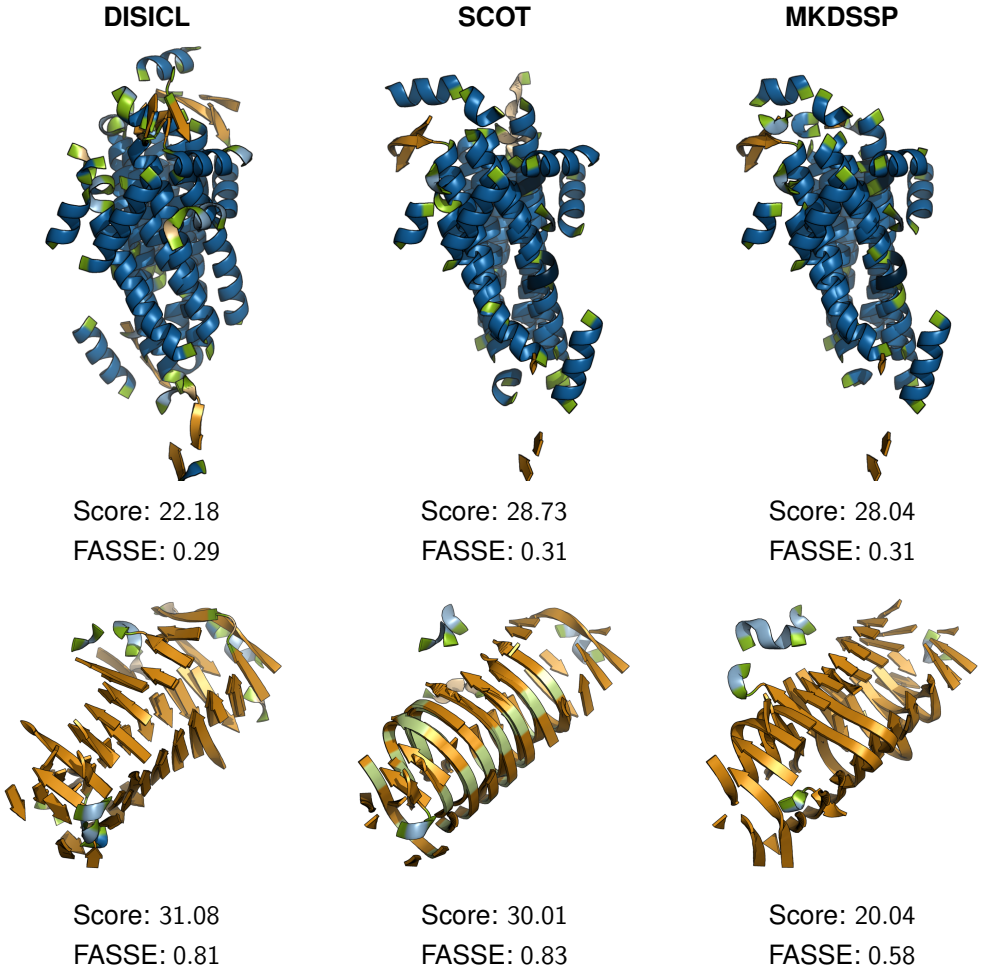


Figure 4.14: LOCK2 alignments of two CATH superfamily domain pairs. LOCK2 alignments of the CATH superfamily domain pairs of Fumarase C (chain A, domain 2, 1gkmA02@cath (1.20.200.10) and 3qbpB02@cath (1.20.200.10)) (top) and Pectate Lyase C-like (1k5cA00@cath (2.160.20.10) and 4u49B02@cath (2.160.20.10)) (bottom). The alignments are given for the SSE classifications with DISICL, SCOT, and MKDSSP together with the corresponding scores per matched SSE and fractions of aligned SSEs (FASSE). This Figure is extracted from [7].

All benefits of SNOT are demonstrated by the comparison of SCOT to other methods which make use of a multitude of general approaches to indicate distinct backbone conformations. The results presented herein shed light on the benefits of both, SNOT and SCOT. Despite the difficulty in *correctly* assigning SSEs (as we cannot provide a *correct answer*), SNOT provides multiple objective criteria to evaluate SSAMs. The geometric uniformity of different SSE classes, the dependency of the assignment on structure quality, the consistency of SSE characterization throughout structural ensembles can be analyzed with SNOT. The impact of helix and strand definitions on the quality of sequence- and SSE-guided and SSE-based structural alignments were evaluated. With respect to these criteria, we compared SCOT to the broadly applied and widely accepted methods MKDSSP and STRIDE, but also to a range of purely geometry-based methods which do not depend on distinct hydrogen bond definitions. The latter methods are clearly superior concerning the geometric regularity of the SSEs which is in accordance with other studies [52, 65].

However, their robustness with respect to structure quality and flexibility is significantly lower. The hydrogen bond-based methods MKDSSP and STRIDE provide the most robust classifications and are well suited for SSE-based structure comparisons but show high BDAs of the assigned helices and strands. SHAFT, as an alternative hydrogen bond-based approach for the classification of helices, is also characterized by geometric inconsistencies, restricted to the assignment of α-, 3_{10}-, and π-helices, and less suitable for SSE-based protein comparisons. In contrast to SHAFT, SCOT bridges the benefits of geometry-based and hydrogen bond-based methods by using hydrogen bond and geometry information to gain insights into the structural space of proteins. Its dual character enables robust classifications of SSEs without significant influence on the geometric regularity of assigned SSEs. In consequence, SCOT is perfectly suited to automatically assign SSEs for subsequent helix- and strand-based alignments with methods, such as LOCK2 [101].

Our analyses also underline the necessity for a more comprehensive evaluation of SSAMs to facilitate the researchers' choice for one method. Although good benchmark studies exist and highlight multiple aspects of SSE assignments [129, 131, 16, 80, 132], we show that deeper investigations are necessary to solve the remaining problems of SSE assignment.

Remaining challenges, such as π- and PPII assignments, the consistent classification of left-handed helices, and the reliable differentiation (if ever possible) between β-strands and PPIIs cannot be fully understood as long as there is no common sense concerning their assignment. Unfortunately, this leads to difficulties in the exhaustive analysis of such underestimated SSEs.

Further investigations on these SSE types may also lead to new criteria by which SSAMs can be evaluated and which, thus, motivate their integration into SNOT. Another future development could also consider the integration of the calculation of standard deviations and means for angular values according to Hughes [95] and Batschelet [96]. In contrast to SCOT, the support for parallelization is comparatively barely expedient as the file I/O is already the major factor of the already practical runtimes here.

In summary, SNOT is a comprehensive tool to evaluate a multitude of different objective characteristics of secondary structure assignments combined in six Observers. It demonstrated its benefits for the evaluation of SCOT and six other SSAMs, namely, MKDSSP, STRIDE, ASSP, DISICL, SEGNO, and SHAFT.

Furthermore, this evaluation clearly reveals the benefits of SCOT by bridging the gap between geometric irregularities of hydrogen bond-based assignments and the missing robustness of geometry-based methods. Therefore, SCOT is not restricted to single application domains and facilitates the reliable characterization of backbone geometries for multiple purposes.

"Science, my lad, is made up of mistakes, but they are mistakes which it is useful to make, because they lead little by little to the truth."

Jules Verne, A Journey to the Center of the Earth

5

SLOT | Searching for spatial SSE arrangements

5.1 Introduction

Proteins are the fundamental elements of chemical biology. Their three-dimensional structure provides insights into the core mechanisms of life and death. Since the first protein structures were determined, the correlation between a protein's structure and its function has become a lively field of research in structural biology. In addition, the similarity between proteins with respect to a certain structural layer plays a major role in the classification of proteins or the prediction of a protein's function. For instance, BLAST [39] can be used to find similarities in and to align amino acid sequences, the UCSF Chimera MatchMaker [100] determines alignments additionally based on the SSE sequence, the CATH database [38] contains protein domains in a hierarchical classification scheme, and TM-align [11] aligns proteins and evaluates their similarity on an overall fold level up to the quaternary structures.

In 1985, Hol said "One is struck by the ever-increasing number of examples where amino acid sequences are vastly different and three-dimensional structures are remarkably similar." [133] and emphasized the interest in similarities on a particular structural level in combination with differences with respect to all others. 30 years later, Koch and Waldmann proposed that a similar spatial arrangement of SSEs in the proximity of the binding site (*ligand-sensing cores*) can recognize

similar scaffolds in disregard of the overall fold [15]. Scaffolds in their sense are molecules or inhibitors without terminal side-chains [134]. This concept is based on the PSSC approach (Protein Structure Similarity Clustering) [15, 135, 136], which identifies structural similarities as well as dissimilarities. Until today, two common *ligand-sensing cores* have been exploited for new drugs in the structure-based design and proposed in the literature [33, 34], both with relevance to cancer treatment. In 2010, Dekker et al. proposed a common *ligand-sensing core* in two proteins that led to the discovery of Palmostatin B as an APT1 inhibitor [34]. Two years later, Willmann et al. proposed another common *ligand-sensing core* which spans three proteins. It led to the discovery of Namoline as an LSD1 inhibitor for the impairment of prostate cancer cell growth [33].

The PSSC approach used for their discovery incorporates DaliLite [137] for its structural comparisons, although it does not explicitly focus on SSEs. However, there are 40 different secondary structure comparison methods (SSCMs) published in the literature (see Table 5.1). The most frequent data structures for the representation of SSEs and their arrangements are vectors and graphs. Especially graphs have a long history and a wide range of applications in the field of chemical biology, due to their intuitive and already application-like setup, and their ability to be easily visualized. In this regard, the comparison of graphs to find similarities is usually based on the (maximum common) subgraph isomorphism problem. Willett already discussed the relevance of this problem for the field of structural biology and the corresponding algorithms for the matching of biological structures [138]. Nevertheless, this problem remains NP-complete in general [31] and, therefore, the determination of the maximum common subgraph (MCS) of two given graphs is a challenging task.

This chapter is organized as follows: Section 5.2 gives a more detailed view on the state of the art of the published SSCMs listed in Table 5.1. These are discussed with respect to their applicability for the search for common *ligand-sensing cores*. In addition, the PSSC approach used to identify the published common *ligand-sensing cores* of LSD1 and APT1 is presented in more detail. Section 5.3 motivates SLOT and its benefits, and describes its workflow from parsing, model building, model comparison, to scoring. Furthermore, different developed modeling algorithms using graphs or histograms for the representation of the arrangement of SSEs are introduced. Section 5.6 evaluates the different modeling and model comparison algorithms of SLOT and its performance in comparison to other SSCMs. The latter utilizes two datasets of domain pairs and a query- and target-based dataset containing the aforementioned common *ligand-sensing cores* in a representative set of protein structures. This section also evaluates different SSAMs with respect to the requirements of SLOT. Finally, Section 5.7 discusses the value of the concept of *ligand-sensing cores* for the rational drug design, the benefits of SLOT, and the open challenges.

5.2 State of the Art

Although the concept of *ligand-sensing cores* does not explicitly require dissimilarity with respect to the spatial arrangement of SSEs on the overall fold level, such a similarity always implies similarity on the binding site level. Thus, it is reasonable to claim for this dissimilarity-similarity combination on both levels, as new insights are more likely to be gained fulfilling this condition because a

Method	SSE representation	Non-sequential	Year	AV
MICAN [139]	Vectors	●	2013	●
CLICK [140]	Graphs	●	2011	●
GANGSTA+ [141]	Graphs	●	2008	●
ProSMoS [142]	Vectors	●	2007	●
SSM [143]	Points/coordinates	●	2005	●
LOCK2 [101]	Vectors	●	2004	●
MASS [144]	Least squared lines	●	2003	●
TOP [145]	Vectors	●	2000	●
CSR [146]	Points/coordinates	●	1998	●
GANGSTA [147]	Graphs	●	2006	●
KENOBI/K2 [148]	Other	●	2000	●
VAST [149]	Vectors	●	1996	●
Smolign [150]	Distance matrices	●	2012	
SA Tableau Search [151]	Orientation matrices/tableaus	●	2010	
QP Tableau Search [152]	Orientation matrices/tableaus	●	2009	
TABLEAUSearch [153]	Vectors	●	2008	
FASE [154]	Points/coordinates	●	2006	
FLASH (OPAAS) [155]	Vectors	●	2003	
GRATH [156]	Graphs	●	2003	
Method of Alesker et al. [157]	Vectors	●	1996	
Method of Koch et al. [158]	Graphs	●	1996	
SARF2 [159]	Vectors	●	1996	
COSEC [160]	Vectors	●	1995	
PROTEP [161]	Graphs	●	1993	
deconSTRUCT [162]	Other		2010	●
CBA [163]	Graphs		2006	●
Matras [164]	Vectors		2003	●
PrISM [165]	Distance plots/matrices		2000	●
DEJAVU [166]	Other		1997	●
SSAPc [167]	Distance plots/matrices		1992	●
SSAP[168]	Distance plots/matrices		1989	●
Samira-VP [169]	Vectors		2017	
TS-AMIR(flexible) [170]	Vectors		2014	
MIRAGE-align [171]	Vectors		2012	
TS-AMIR[172]	Vectors		2012	
TetraDA [173]	Residue strings		2005	
Topsalign [174]	Graphs		2003	
SSEA [175]	Sequence		1999	
LOCK [176]	Vectors		1997	
POSSUM [177]	Graphs		1990	

Table 5.1: SSCMs and their features. SSCMs grouped by their ability to compare SSEs in a sequential or non-sequential (●) fashion, and their availability as a standalone executable (●), as a web service (●), or being not available at all. The basic representation of an SSE and the year of publication is given for each SSCM additionally.

multitude of approaches for the identification of the overall fold similarity already exist (see Table 5.1). In general, there is no SSCM available that combines these two steps of the search for common *ligand-sensing cores*, i.e., the comparison of the overall fold and the comparison of the binding sites if the overall folds are different.

However, there is a limited number of SSCMs available that enable a sequence-independent identification of structural similarities with respect to the SSE arrangements, but only LOCK2 allows the use of externally provided SSE annotations. Although some of the SSCMs presented in Table 5.1 support external SSE annotations, such as SSAPc or Matras, they require SSE annotations in the DSSP file format. However, the use of the DSSP file format by other SSAMs has several limitations. First, the file format contains a multitude of information, e.g., solvent accessibility, which may neither be calculated nor supported/provided by other SSAMs, e.g., SCOT. Second, it does not support PPII helices. Third, it does not support overlapping SSEs. Especially SCOT supports the assignment of overlapping right-handed α- and π-helices. The benefits of this feature have already been demonstrated in Section 4.6.3.2. Nevertheless, there is no scientific standard for the output of an SSAM in general.

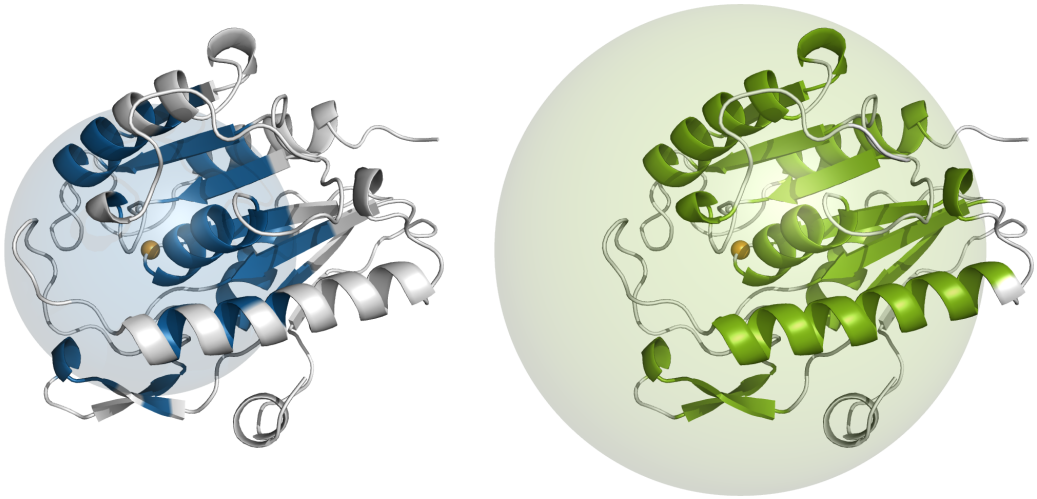
5.2.1 PSSC

The discovery of the common *ligand-sensing cores* for LSD1 [33] and APT1 [34] was based on an approach called PSSC (Protein Structure Similarity Clustering) [15, 135, 136]. An input query protein is structurally aligned against all proteins of the PDB with Dali/FSSP [178] and the Combinatorial Extension (CE) algorithm [179], to generate a hitlist with decreasing structural similarity. This hitlist is filtered with respect to pharmaceutically relevant superfamilies with a low sequence similarity (of up to $20\,\%$ to obtain interesting cases). The remaining hits are visually inspected. Promising hits are superposed with respect to their *ligand-sensing cores* and an RMSD of at most $4\,$Å–$5\,$Å.

The extraction of a *ligand-sensing core* is described most detailed by Dekker et al. [34]. The authors chose Ser 114 of APT1 as the center around which the *ligand-sensing core* was defined. They placed a sphere of $15\,$Å around this center and included all SSEs that were partly or entirely within this sphere. These SSEs were then cut to a sphere of $25\,$Å around the aforementioned center (see Figure 5.1).

From a computational point of view, the main drawback of the PSSC approach is the high emphasis on the visual inspection. This emphasis is also underlined by the authors [33, 34]. The reasons are that, on the one hand, it can hardly be automated, and that, on the other hand, the precise criteria of the visual inspection remain concealed. Both facts hamper the use of this approach in a direct comparison to other tools.

Therefore, we decided to compete the SLOT approach with the following briefly introduced tools.



(a) 15 Å **(b)** 25 Å

Figure 5.1: Visualization of the definition of the *ligand-sensing core* in 1fj2A@pdb by Dekker et al. [34]. The center is represented by an orange sphere. We used the coordinates of the Cα of Ser 114. (a) Visualization of the zone of 15 Å (blue sphere) around the center and the residues of SSEs within this zone are highlighted in blue. (b) The extended zone of 25 Å (green sphere) around the center, the previously selected and the additionally extended residues of the SSEs are highlighted in green. We classified the SSEs with MKDSSP (see Section 4.4.3).

5.2.2 DaliLite

DaliLite [137] is the main algorithm behind the CATH database [38] and a sub-routine of the previously introduced PSSC approach. It uses Cα–Cα distance matrices, which are then decomposed into elementary contact patterns and combined into sets of common pairs in both matrices. A Monte Carlo algorithm is used to explore the search space and iteratively improve the current solution of mapped residue pairs. Starting with an initial seed, the algorithm performs two basic operations, i.e., the expansion of the temporary solution with residue pairs based on the elementary contacts and the trimming or removal of any tetrapeptide fragments. The outcome is an alignment plus an additive similarity score, namely, the Z-score.

5.2.3 LOCK2

LOCK2 is a superposition script and part of FoldMiner [101]. It represents user-provided SSEs by vectors between the centroid of terminal residues (2 of strands and 4 of helices). In each iteration, a pair of SSEs from a query protein is superimposed on a pair of SSEs of a target protein. The resulting global protein superposition is scored by a dynamic programming algorithm. It allows gaps within overlapping vectors. In contrast to its predecessor LOCK [176], it is a non-sequential alignment algorithm.

5.2.4 TM-align

TM-Align [11] has been used in numerous applications related to structural protein comparison (e.g., to find the closest structural homologue to a predicted structure [180]), but has also been used with respect to binding sites [4]. It combines the TM-score rotation matrix and dynamic programming. At first, three initial structural alignments are calculated based on the residues in SSEs, a gapless threading of the smaller against the larger structure, and a combination of the first two with a gap-opening penalty. Finally, the rotation matrix is used to rotate the structures and to obtain a similarity score matrix by the use of a heuristic iterative algorithm.

5.3 SLOT

Although there are more than 40 SSCMs published in the literature, only LOCK2 fulfills the minimum requirements, i.e., the availability as a standalone tool, the use of external SSE annotations, and the non-sequential matching. We have developed SLOT (Secondary structure Layer One Two) to fulfill these and additional individual requirements to discover similar SSE arrangements. It consists of several different modules for the input, modeling, comparison, and output. These can be configured using a configuration file described in Section 5.3.1.1. The following sections are dedicated to the modules of the workflow. These are named with respect to their name in the configuration file and presented in the order of their usage in the workflow.

SLOT processes one or two datasets of proteins. Each dataset consists of the following modules: an Identifier (see Section 5.3.2), a Protein (parser) (see Section 5.3.4), an optional Pocket (parser) (see Section 5.3.5), a Collector (see Section 5.3.3), a Modeler (see Section 5.3.6), and an optional Model Writer (see Section 5.3.7). The Identifier identifies the proteins of the dataset by their PDB-IDs. The Protein parser parses all proteins from a specified directory using the PDB-IDs to select and identify the protein files. The optional Pocket parser reads pocket files defining multiple pockets for each protein. The Collector collects the SSEs from either the entire protein, a protein chain, or the parsed pockets of a protein in (SSE) collections. These collections are processed by the Modeler which creates a representation/model for each collection. The optional Model Writer exports entire models for visualization purposes, or certain characteristics for further analyses.

We present five graph- and one histogram-based modeling algorithms. The most sophisticated and promising candidate is the SegmentedV1DM algorithm described in Section 5.3.6.4, which is incorporated in the final version of SLOT. It uses undirected complete labeled graphs for the representation of SSEs. The geometry of each SSE is mimicked by so called segmentation points along an SSE's axis. The similarity of two such graphs is based on the MCS. Its determination utilizes an optimized maximum clique detection algorithm in an appropriately defined modular product graph of the two input graphs. A different approach are the turn histograms introduced in Section 5.3.6.6. These represent the occurrences of turns in histograms and more or less implicitly capture the geometry of a protein and its SSEs.

The workflow contains three more modules. The Comparator (see Section 5.3.8) compares the models in a pairwise manner and determines their maximum common substructure which is also referred to as matching. These matchings can be exported by the optional Match Writer (see Section 5.3.8) for visualization purposes, for instance. Finally, the Judge (see Section 5.3.10) judges or scores each matching and provides several scores in an output file.

SLOT is implemented without the use of external libraries for maximum versatility and to be able to create a tailor-made tool.

SLOT is written in C++ and parallelized using OpenMP.

5.3.1 Input

In contrast to SCOT and SNOT (see Chapters 3 and 4), all parameters of SLOT for the parsing, modeling, model comparison, and writing can be set up via a configuration file. Thus, the executable solely requires this configuration file and the number of threads for parallel execution.

5.3.1.1 Configuration Files

The SLOT configuration files are text-based files which contain all parameters for the input, processing, and output of SLOT. A configuration is a hierarchical tree consisting of vertices annotated with properties. Each line in the configuration file contains a parameter or the name of a new vertex. The differentiation between parameters and vertices is based on the colon. The hierarchy and affiliation of the parameters and vertices are established by white-space indentations, similar to the way the programming language Python represents the hierarchy of classes, functions, or code blocks, for instance. Based on the indentation length d of a vertex or a parameter, it is added to its parent with indentation length $d - 1$.

The configuration file can contain the definition of one or two datasets. If only one dataset is specified containing n entries, all entries are compared to each other in an all-against-all fashion but without identity pairings leading to $\frac{n(n-1)}{2}$ pairwise comparisons. Otherwise, if two datasets with n and m entries are defined, $n \cdot m$ pairwise comparisons are performed by the workflow.

A reduced example of a configuration file is shown in Figure 5.2. An exhaustive example is given in Figure 6.3 of the appendix.

5.3.2 Identifier

The Identifier identifies the proteins of a dataset by their PDB-IDs. These PDB-IDs are used throughout the entire workflow to identify PDB or pocket files and internally to identify and label proteins and models. They can be specified via a white-space-delimited string, a file containing a PDB-ID in each line, or by the files within a directory. The latter lists all files of a directory

```
dataset1
 name:ligand-sensing cores lsd1
 identifier
  string:1gos 2bxr 2ejr
 protein
  directory:/datasets/proteins/
 collector
  protein
 modeler
  max-edge-distance:20
 model-writer
  pymol
   directory:/results/models/

comparator
 vertices
 edges
  distance-deviation-dynamic:0.1
  distance-deviation-static:2

judge
 file:/results/scores.txt
```

Figure 5.2: Reduced example of a configuration file.

with a specific file extension and uses their file names as PDB-IDs. All three ways can be used simultaneously for a maximum of flexibility. However, at least one is required for all further processing steps.

5.3.3 Collector

The Collector collects sets of SSEs in so called collections. A collection contains a set of SSEs, the corresponding model (e.g., an SSE graph), and the respective protein. Each collection also contains a collection ID which consists of the PDB-ID of the protein and a sequential number (e.g., 1gos_2). There are three Collectors available: protein, chain, and pocket. The protein Collector collects all SSEs of a protein in one collection. The chain Collector creates a separate collection for each chain and adds all SSEs belonging to the chain to that collection. The pocket Collector collects the SSEs from the parsed pockets (see Section 5.3.5) of a protein creating a collection for each pocket. A pocket K is defined on a set of residues $R_K \subseteq P$ of a protein P. A pocket residue (sequence) padding can optionally be specified to also include the neighboring residues of all $r \in R_K$. For instance, if a padding of 2 is given and if $r_i \in R_p$ is a pocket residue, we also take $r_{i-2}, r_{i-1}, r_{i+1}$, and r_{i+2} into account. Let R_K include all initial and neighbor-padding residues of a pocket K. The corresponding collection to pocket K contains all SSEs of protein P which share at least one residue in R_K. For every Collector, the SSE types (helices, sheets, or turns) to be collected have to be specified (e.g., helices:y). Finally and similarly to the Identifier (see

Section 5.3.2), multiple Collectors of different types (e.g., protein or chain) can be simultaneously specified in the configuration file.

5.3.4 Protein

The protein parsing procedure of SLOT requires standard PDB files as input and is similar to the one described for SNOT in Section 4.3.1.1. All files with a given file extension (`.pdb`) and with a PDB-ID as the file name provided by the Identifier (see Section 5.3.2) are parsed from a specified directory.

5.3.5 Pocket

The parsing of pockets is optional. Similar to the Protein parser (see Section 5.3.4), the Pocket parser identifies files by the PDB-IDs provided by the Identifier (see Section 5.3.2) in a given directory and with a given file extension. The major difference in the handling of files is that multiple pocket files can be provided for a single PDB-ID. Each of which defines a single pocket. These files have to be continuously numbered starting with `1`. An optional pocket number delimiter (e.g. _) can be specified if a delimiter for the PDB-ID and pocket number is used (e.g. `1gos_1.pdb`). A maximum number of pockets can optionally be defined to limit the number of parsed pockets even if more are available.

The pocket files themselves have to be in the PDB file format containing `ATOM` lines only. All residues specified in these lines are added to a pocket.

5.3.6 Modeler

We have developed six different algorithms to model a set of SSEs and to search for (structural) similarities. Five of these models are based on undirected labeled graphs representing the SSEs and their arrangement in the three-dimensional space and in some cases the geometry of the SSEs themselves. The first three algorithms, namely, StaticV1D1 (see Section 5.3.6.1), StaticV2D1 (see Section 5.3.6.2), and StaticV3D1 (see Section 5.3.6.3), use a static or fixed number of vertices to represent each SSE (1, 2, and 3) plus a single distance at the edges. The last two graph-based algorithms, namely, SegmentedV1DM (see Section 5.3.6.4) and SegmentedVSD1 (see Section 5.3.6.5), segment each SSE to mimic its axis. The first one uses a single vertex and distance matrices at the edges whereas the second one utilizes multiple vertices and single distances at the edges. The final modeling algorithm, i.e., Turn Histograms (see Section 5.3.6.6), is based on histograms in which the number of turns for each category, length, and class are counted separately.

The graph-based algorithms are given in the order of their development. All models, whether it is a graph or a histogram, have a model ID which corresponds to the collection ID it is originated from.

5.3.6.1 Graph StaticV1D1

The StaticV1D1 modeling algorithm creates an undirected labeled graph $G = (V, E)$ for each Collection of SSEs. We represent each SSE (helix or strand) by a vertex $v \in V$. Each v is labeled with an integer code representing the type t (0 for helices, 1 for strands) and the classification c (e.g., 1 for right-handed α-helices) of an SSE, and a point in the three-dimensional space. The code combines t and c to reduce number of criteria to be checked to determine whether two vertices are compatible or not. It is defined by Equation 5.1. We use the classification number of right-handed α-helices (1) also for right-handed π-helices. The point is based on the coordinates of the median residue's Cα atom if the length of the SSE is odd. If this residue is missing, we use the coordinates of the Cα atom of its successor. If the length of the SSE is even and if none of the two central residues is missing, we use the mean of their Cα atom coordinates. Otherwise, we simply return the coordinates of the residue's Cα atom that is not missing.

$$\mathrm{code}_{\mathrm{S}}(t, c) := (c \cdot 10) + t \tag{5.1}$$

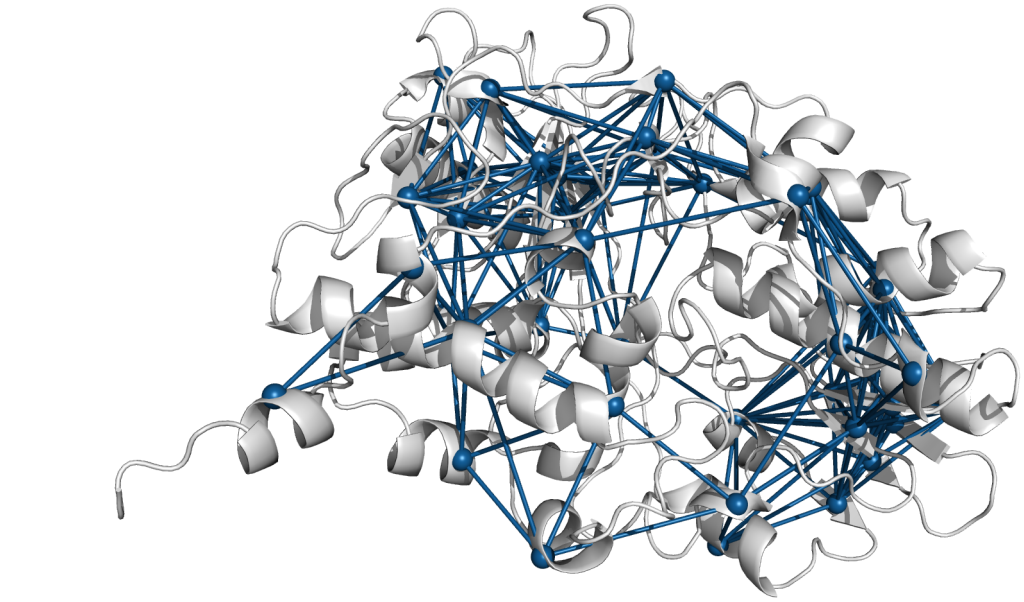
After all vertices are created, we connect them by adding edges to the graph. Let $u, v \in V$ be two vertices of G. u and v are adjacent, i.e., $(u, v) \in E$, if the Euclidean distance between the points of their labels is less than or equal to a certain threshold, namely, the connectivity distance. Thus, vertices are only adjacent to vertices of their corresponding Euclidean neighborhood in the three-dimensional space. This means that missing edges between vertices representing distant SSEs allow for a certain degree of flexibility in the determination of the MCS of two graphs. Figure 5.3 illustrates the graph for 1gosA@pdb for three different connectivity distances (16 Å, 20 Å, and 24 Å).

Vertex Compatibility

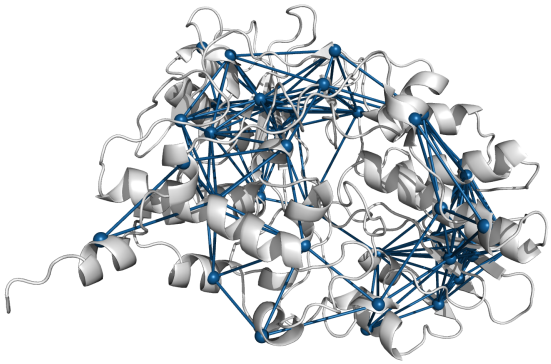
The vertex compatibility is solely based on the SSE type and class code of the labels of the vertices under investigation. Two vertices u, v are compatible if they are labeled with the identical code or if the labels represent right-handed helices of which one is of class mixed and the other of class α or 3_{10}. Therefore, only vertices representing SSEs of the same type and classification are compatible with two exceptions. Right-handed α- and π-helices share the same code and hence are compatible. In addition, right-handed mixed helices are compatible to right-handed α- and 3_{10}-helices. Please note that the points all vertices are labeled with are only used for the labeling process of the edges and for visualization purposes. They are explicitly not considered in the determination of the compatibility.

Edge Compatibility

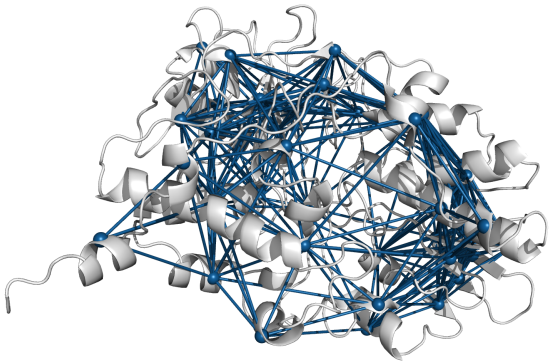
The edge compatibility reflects the similarity of the arrangements of two pairs of SSEs represented by the incident vertices of two edges. Let d_1 and d_2 be the distances of the labels of the two edges under investigation. We use two parameters, i.e., $static_{dst}$ and $dynamic_{dst}$, to determine the



(a) 20 Å



(b) 16 Å



(c) 24 Å

Figure 5.3: Visualization of the graph for 1gosA@pdb created by the StaticV1D1 algorithm. Visualization of the graph (blue) for 1gosA@pdb (gray) created by the StaticV1D1 algorithm (see Section 5.3.6.1) for different edge connectivity distances. The SSEs used for the visualization and the creation of the graph were assigned with SCOT.

allowed deviation between these distances (see Equation 5.2).

$$\text{allowed}_{\text{dst}}(d_1, d_2) := (\min(d_1, d_2) \cdot dynamic_{dst}) + static_{dst} \tag{5.2}$$

$$\text{deviation}_{\text{dst}}(d_1, d_2) := |d_1 - d_2| \tag{5.3}$$

The final compatibility is given in Equation 5.4. Two edges are compatible if the difference between

the distances of their labels is less than or equal to the allowed deviation.

$$\text{compatible}_{\text{dst}}(d_1, d_2) := \text{deviation}_{\text{dst}}(d_1, d_2) \leq \text{allowed}_{\text{dst}}(d_1, d_2) \tag{5.4}$$

5.3.6.2 Graph StaticV2D1

The StaticV2D1 algorithm is very similar to the StaticV1D1 algorithm described in Section 5.3.6.1. It also creates an undirected labeled graph $G = (V, E)$ for each Collection of SSEs, uses the same labels for vertices (code, point) and edges (distance), and also defines the adjacency of vertices via edges based on a connectivity threshold. The only difference is that each SSE is represented by two vertices $u, v \in V$ to incorporate its length and also to fix its orientation in the three-dimensional space. We use the coordinates of the Cα atom of the SSE's N-terminal residue for u and the coordinates of the Cα atom of its C-terminal residue analogously for v. If one of these residues is set es missing, we use the successor of the N-terminal residue instead. We use the predecessor of the C terminal residue analogously.

All vertices are connected via edges if the Euclidean distance between the points of their labels is less than or equal to a connectivity distance (see Section 5.3.6.1 for more details). Figure 5.4 illustrates the graph for the same protein (1gosA@pdb) for three different connectivity distances (16 Å, 20 Å, and 24 Å).

Vertex Compatibility

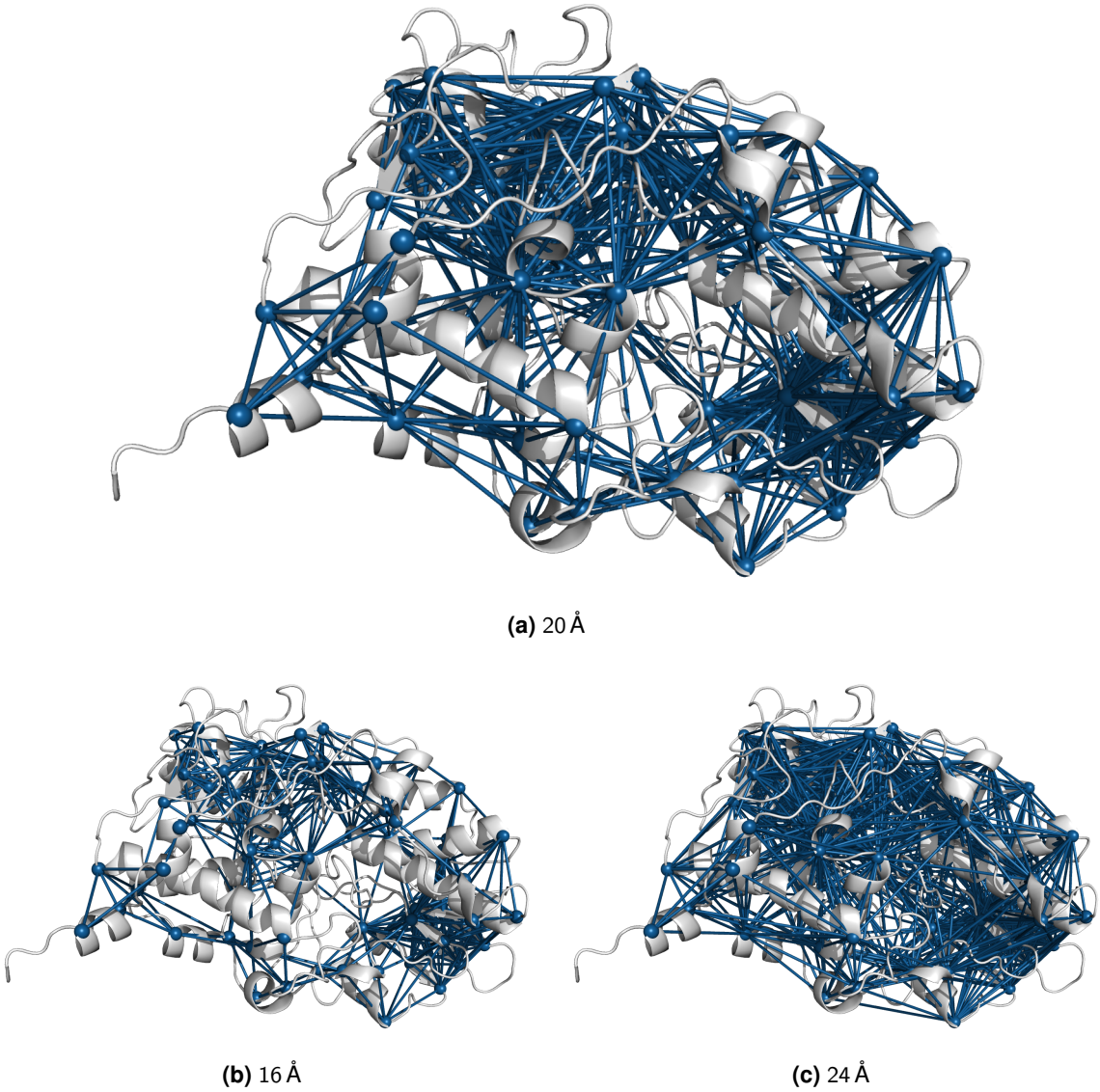
The vertex compatibility is identical to the one described for the StaticV1D1 modeling algorithm.

Edge Compatibility

The edge compatibility is identical to one the described for the StaticV1D1 modeling algorithm.

5.3.6.3 Graph StaticV3D1

The StaticV3D1 modeling algorithm is a combination of both previously described modeling algorithms, namely, StaticV1D1 and StaticV2D1 (see Sections 5.3.6.1 and 5.3.6.2). In accordance, it also uses an undirected labeled graph for the representation of the SSEs of a Collection. In contrast, however, each SSE S is represented by three vertices $u, v, w \in V$. These vertices are labeled with the points based on the N- (u) and C-terminus (w), and its center (v) to incorporate the geometry of S and to allow for a higher degree of partial-SSE matching. Furthermore, these points are not solely based on the coordinates of the Cα atom of the corresponding residue, but on the mean of the residue's backbone atoms (N, Cα, C). The rules for the replacement of missing



(a) 20 Å

(b) 16 Å

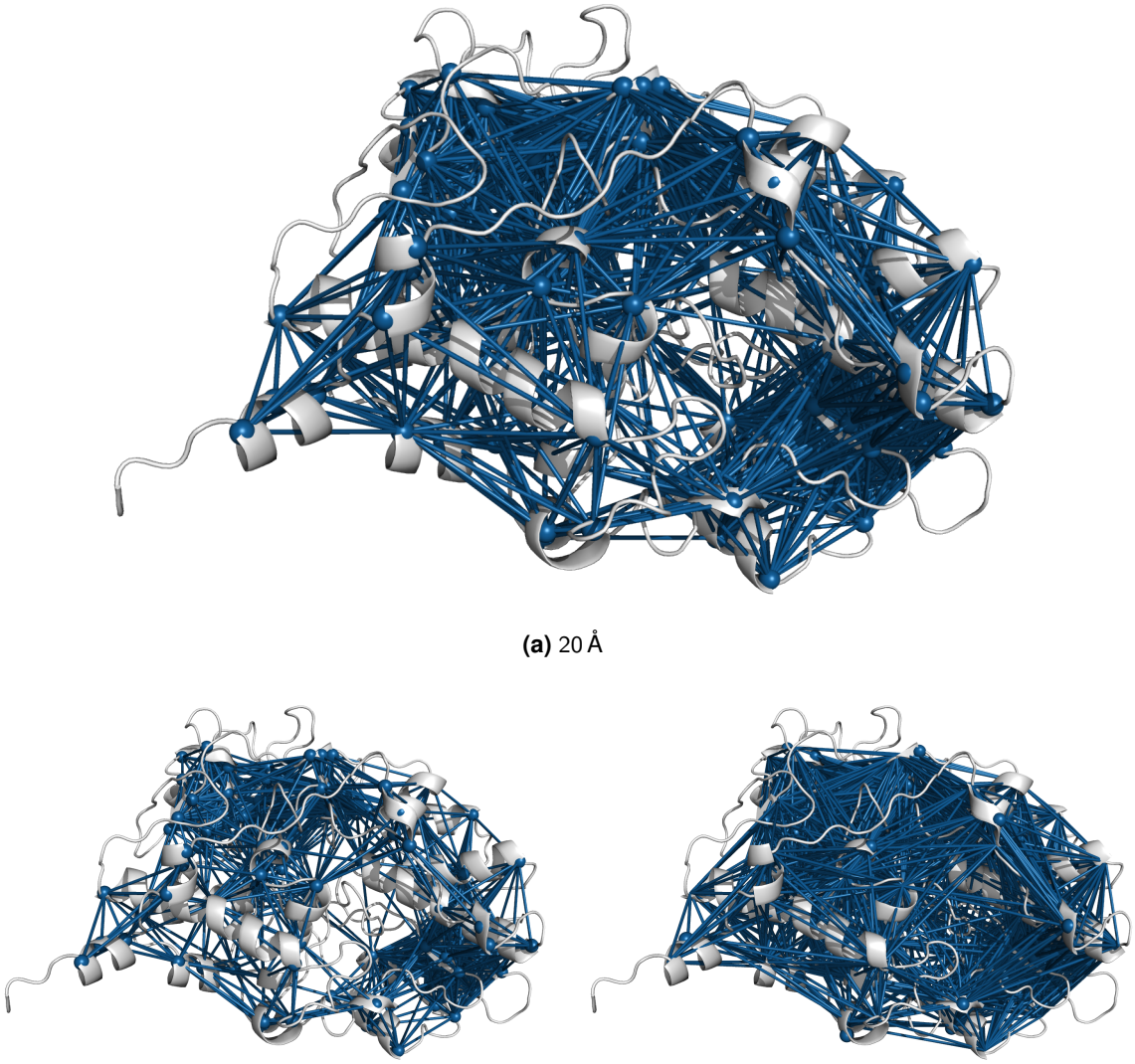
(c) 24 Å

Figure 5.4: Visualization of the graph for 1gosA@pdb created by the StaticV2D1 algorithm. Visualization of the graph (blue) for 1gosA@pdb (gray) created by the StaticV2D1 algorithm (see Section 5.3.6.2) for different edge connectivity distances. The SSEs used for the visualization and the creation of the graph were assigned with SCOT.

residues (StaticV1D1 and StaticV2D1) and the definition of the central point (StaticV1D1) also apply here.

Two vertices $u, v \in V$ are connected via an edge $(u, v) \in E$, if the Euclidean distance between the points of their labels is less than or equal to a connectivity distance (see Section 5.3.6.1 for more details). The final graph for protein 1gosA@pdb for three different connectivity distances (16 Å, 20 Å, and 24 Å) is illustrated in Figure 5.5.

Vertex Compatibility



(a) 20 Å

(b) 16 Å

(c) 24 Å

Figure 5.5: Visualization of the graph for 1gosA@pdb created by the StaticV3D1 algorithm. Visualization of the graph (blue) for 1gosA@pdb (gray) created by the StaticV3D1 algorithm (see Section 5.3.6.3) for different edge connectivity distances. The SSEs used for the visualization and the creation of the graph were assigned with SCOT.

The vertex compatibility is identical to the one described for the StaticV1D1 modeling algorithm.

Edge Compatibility

The edge compatibility is identical to the one described for the StaticV1D1 modeling algorithm.

5.3.6.4 Graph SegmentedV1DM

The SegmentedV1DM algorithm is the one incorporated in the final version of SLOT. In contrast to the previous algorithms, it utilizes *complete* undirected labeled graphs for the representation of the SSEs. The algorithm creates a graph $G = (V, E)$ for each Collection. Each SSE S of a Collection is represented by one vertex $v \in V$. Each vertex v is labeled with the type and classification code of S according to Section 5.3.6.1 and a set of segmentation points STP. These points mimic the geometry and the axis of each SSE S.

Let r_i be the N-terminal residue of S. If S is a helix, for every second residue (r_i, $r_{i+2}, \ldots$) in the sequence, we create a segmentation point p. Each point p is based on the mean of the N and C atom coordinates of the following residues including r_i that are required to form a helix coil. This number of residues/atoms is individual for every helix class due to different underlying turns and corresponding dihedral angles. Table 5.2 gives the residues and atoms per coil with respect to a helix' class. For instance, at each second residue we consider the following 11 backbone atoms in an right-handed α-helix. Thus, p for residue r_i is based on the coordinates of the N and C atoms of residues r_i, r_{i+2}, r_{i+4} and the N atom of residue r_{i+6}. If the helix is shorter than a single coil, p is based on the mean of all chain-trace backbone atoms of all of its residues. In case S is a strand, we create a segmentation point p for every residue r_i based on the mean N atom coordinates of r_i and its successor r_{i+1}. Missing residues at termini are not taken into account. If the strand consists of a single residue, p is set to the mean of the N and the C atom coordinates of that single residue. Figure 5.6 illustrates the segmentation points for a right-handed α-helix and a strand assigned by SCOT in 1gosA@pdb.

Helix	Residues per coil	According to	Backbone atoms per coil
RH α	3.6	Bamford et al. [181]	11
RH 3_{10}	3.0	Donohue [121]	9
RH π	4.4	Low and Grenville-Wells [182]	13
PPII	3.0	Crick and Rich [183]	9
2.2_7	2.0	Donohue [121]	6
other	3.6		11

Table 5.2: Numbers of residues and atoms per coil for different helix classes used for the calculation of segmentation points in helices. The number of atoms per coil corresponds to the number of residues multiplied by 3 (number of chain-trace backbone atoms, i.e., N, Cα, and C). A more detailed list including the mean dihedral angles and the residues per coil can be found in Ramachandran and Sasisekharan [120].

All SSEs for which a minimum required number of segmentation points was created ($|STP|$) are represented by a vertex $v \in V$. This minimum threshold can be adjusted for helices and strands separately and is set to 2 for both by default.

The next step of the algorithm is dedicated to the connection of all pairwise distinct vertices with each other to create a complete graph. Let $u, v \in V$ be two vertices and STP_u be the segmentation points of the label of u and STP_v of v, respectively. Let $e = (u, v) \in E$ be the directed edge with source u and target v. The labeling of the opposite edge $(v, u) \in E$ required for the

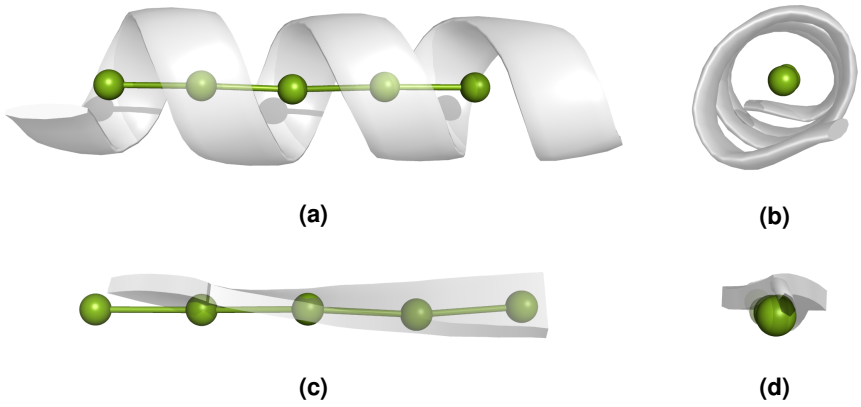


Figure 5.6: Visualization of the segmentation points for a helix. Visualization of the segmentation points (green spheres) for a right-handed α-helix on residues 159–171 (a, b) and for a strand on residues 245–249 (c, d) in 1gosA@pdb. Both are given in two perspectives, orthogonal and parallel to the respective SSE axis. The straight lines between the segmentation points are added to emphasize deviations from the ideal trace. The SSEs used for the visualization and the creation of the segmentation points were assigned with SCOT.

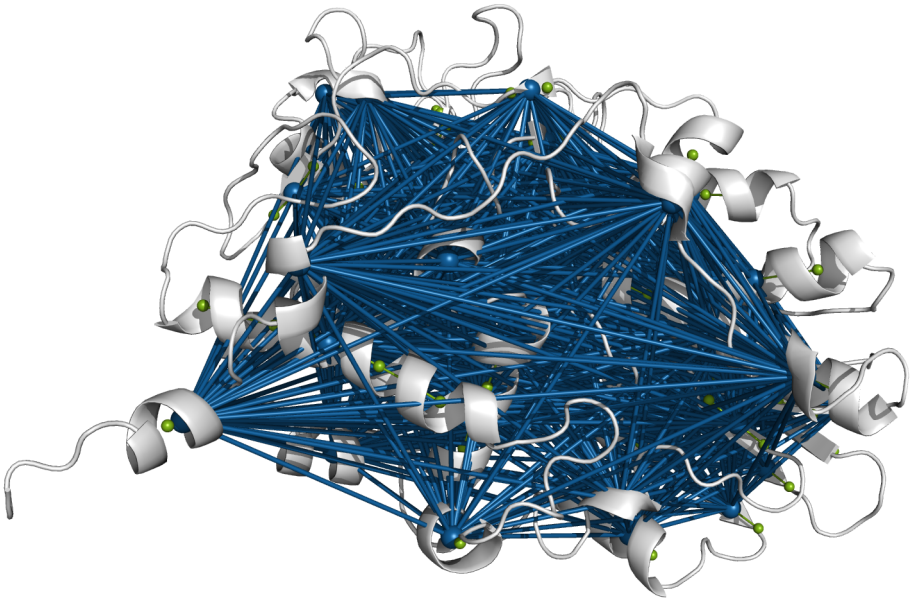
omnidirectionality is explained later. We label e with a distance matrix $M_{u,v}$ of size $|STP_u| \times |STP_v|$. $M_{u,v}$ contains all pairwise distances between a segmentation point in STP_u and one in STP_v. For instance, $M_{u,v}$ contains at indices $1, 1$ the distance between the first segmentation point of STP_u and the first one of STP_v. An example is given in Figure 5.8c.

As the segmentation points for an SSE are given in sequence direction we transform $M_{u,v}$ whenever the corresponding SSEs are arranged in an opposite direction or an anti-parallel sense. If the sum of distances at indices $1, 1$ and $|STP_u|, |STP_v|$ is smaller than the sum of distances at indices $1, |STP_v|$ and $|STP_u|, 1$, the sense is anti-parallel. In other words, if the sum of distances front to front and back to back are longer than the corresponding cross distances, the sense is anti-parallel. In this case, we horizontally mirror or flip $M_{u,v}$. For instance, all distances in the first column are exchanged with the values of the last column. Finally, the opposed directed edge to $e = (u, v)$, i.e., $(v, u) \in E$, is labeled with the transposed matrix $M_{v,u} := M_{u,v}^{\top}$.

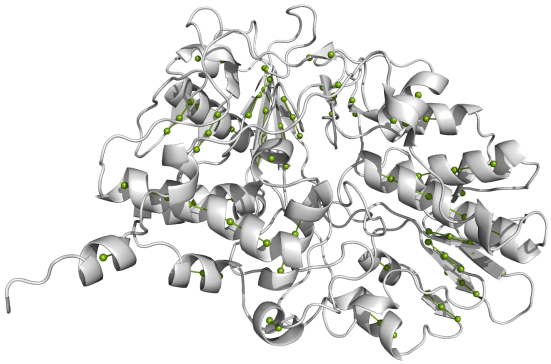
A visualization of a graph for 1gosA@pdb is given in Figure 5.7. The coordinates of each vertex correspond to the median segmentation point or the mean of the two central segmentation points of its label if $|STP| \bmod 2 = 0$.

Vertex Compatibility

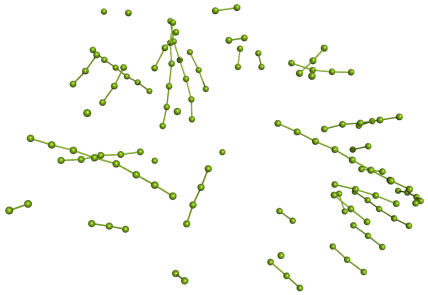
The compatibility of two vertices u, v is based on two criteria. First, the vertex compatibility criteria defined by the StaticV1D1 algorithm (see Section 5.3.6.1) also applies here, i.e., the compatibility with respect to the corresponding SSE types and classes. Second, the difference in the number of segmentation points of each label has to be within a certain threshold. Let STP_u and STP_v be the segmentation points of the labels of the vertices u and v respectively. We use two parameters, i.e., $static_{STP}$ and $dynamic_{STP}$, to determine the allowed deviation between the



(a) Protein, graph, and segmentation points



(b) Protein and segmentation points



(c) Segmentation points only

Figure 5.7: Visualization of the graph and the segmentation points for 1gosA@pdb created by the SegmentedV1DM algorithm. Visualization of the graph (blue) and the segmentation points (green) for 1gosA@pdb (gray) created by the SegmentedV1DM algorithm (see Section 5.3.6.4). The segmentation points of each SSE are connected by straight lines to indicate their affiliation. The SSEs used for the visualization and the creation of the graph were assigned with SCOT.

numbers of segmentation points $|STP_u|$ and $|STP_v|$.

$$\text{allowed}_{\text{stp}}(|STP_u|, |STP_v|) := (\max(|STP_u|, |STP_v|) \cdot dynamic_{STP}) + static_{STP} \tag{5.5}$$

$$\text{deviation}_{\text{stp}}(|STP_u|, |STP_v|) := ||STP_u| - |STP_v|| \tag{5.6}$$

$$\text{compatible}_{\text{stp}}(|STP_u|, |STP_v|) := \text{deviation}_{\text{stp}}(|STP_u|, |STP_v|) \leq \text{allowed}_{\text{stp}}(|STP_u|, |STP_v|) \tag{5.7}$$

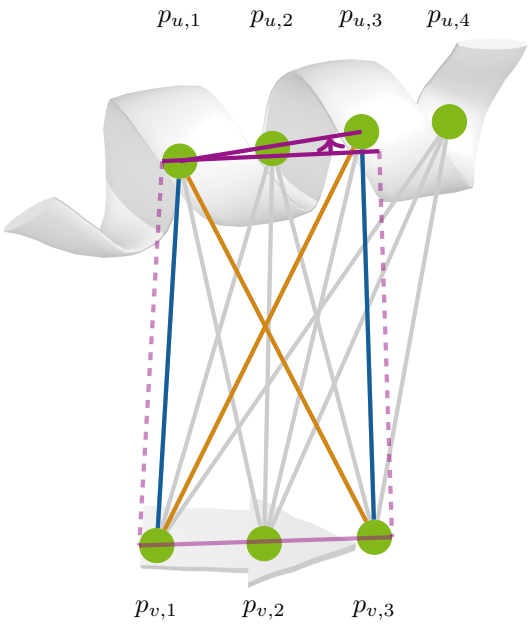
If the deviation $\text{deviation}_{\text{stp}}(|STP_u|, |STP_v|)$ (see Equation 5.6) is at most the allowed deviation $\text{allowed}_{\text{stp}}(|STP_u|, |STP_v|)$ (see Equation 5.5) both vertices u and v are compatible (see Equation 5.7). Note that in contrast to the definition of $\text{allowed}_{\text{dst}}$ in Section 5.3.6.1, the definition here is based on the maximum instead of the minimum.

Edge Compatibility

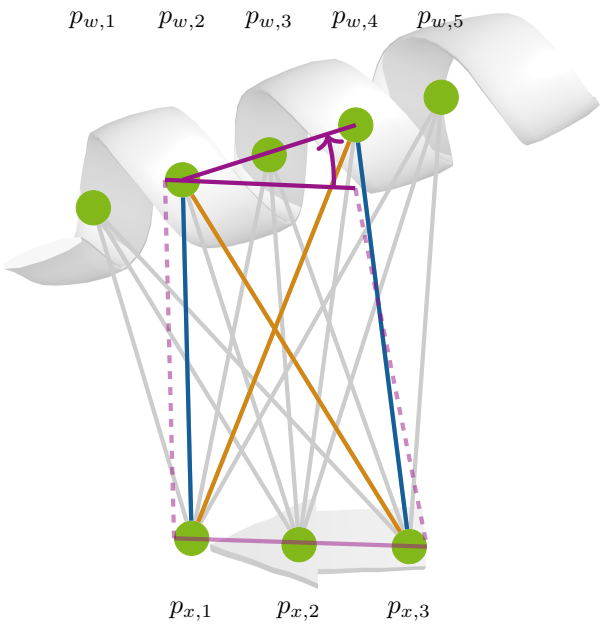
The compatibility of edges is based on the distance matrices of their labels and the segmentation points of the labels of the incident vertices. Let u, v, w, x be vertices, $(u, v), (w, x)$ be two edges, $M_{u,v}, M_{w,x}$ the corresponding distance matrices, and $STP_u, STP_v, STP_w, STP_x$ the corresponding segmentation points. In a figurative sense, we try to match subsets of segmentation points of STP_u, STP_v and the according distances to subsets of STP_w, STP_x (see Figure 5.8).

Algorithm 1 shows the determination of the compatibility in detail. At first, we determine the minimum numbers of rows m_r and columns m_c of $M_{u,v}$ and $M_{w,x}$ based on the numbers of segmentation points. The displacement allows a sub-matching and is defined by a static and dynamic parameter with respect to m_r and m_c. In our example in Figure 5.8, $static_{dsp} = 1, dynamic_{dsp} = 0, minimum_{mtc} = 3, m_r = 4$, and $m_c = 3$. This means, that only 3 of a minimum of 4 rows are taken into account at the shown step. Although $static_{dsp} = 1$, 3 instead of 2 columns are taken into account for the strands, as the minimum number of matched segmentation points $minimum_{mtc}$ is set to 3. For all displacements d_r and d_c, we calculate the numbers of rows n_r and columns n_c (which correspond to the number of segmentation points to be matched). We iterate d_r and d_c from the maximal displacement to 0, which means that at first the minimal numbers of rows n_r and columns n_c are considered (bottom up). For all possible combinations of n_r and n_c consecutive indices, we compare the outer and cross distances and the angle between the straight lines defined by the corresponding segmentation points. In correspondence to the pseudo-code, the indices of the current iteration of the example are set to $i = 1, j = 1, k = 2, l = 1$. The outer distances are located at indices i, j $(1, 1)$ and $i + n_r, j + n_c$ $(3, 3)$ in $M_{u,v}$, and k, l $(2, 1)$ and $k + n_r, l + n_c$ $(4, 3)$ in $M_{w,x}$ (highlighted in blue). The cross distances analogously at $i, j + n_c$ $(1, 3)$ and $i + n_r, j$ $(3, 1)$ in $M_{u,v}$ (highlighted in orange), for instance. For both distances, we have a static and a dynamic parameter to define the allowed deviations. The compatibility with respect to these params is given in lines 37 and 40 of Algorithm 1. If the deviations of these distances are within the allowed tolerances, we calculate the angle between the vectors defined by the corresponding segmentation points.

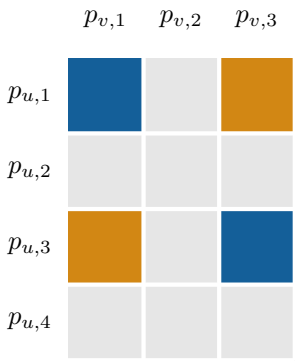
For the current indices, the vectors for the first edge are defined by $p_{u,i}, p_{u,i+n_r} \in STP_u$ and $p_{v,j}, p_{v,j+n_c} \in STP_v$. The vectors of the other edge are defined analogously. The allowed deviation between the angles span by these vectors can be adjusted by three parameters. These parameters allow a fixed degree of deviation (static) plus a temperature factor that allows higher deviations the lower the minimum of n_r and n_c is.



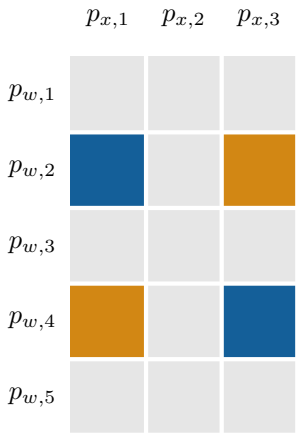
(a) Compatibility criteria for edge (u, v).



(b) Compatibility criteria for edge (w, x).



(c) Distance matrix $M_{u,v}$ of the directed edge (u, v).



(d) Distance matrix $M_{w,x}$ of the directed edge (w, x).

Figure 5.8: Visualization of the compatibility criteria for two edges. Visualization of the compatibility criteria for two edges $(u, v), (w, x)$ based on the distance matrices of their labels $M_{u,v}, M_{w,x}$ and the segmentation points $STP_u, STP_v, STP_w, STP_x$ (green spheres) of their incident vertices. The matrix entries corresponding to the distances in the figures above are highlighted in the same colors.

In a nutshell, the fewer indices or segmentation points are matched or the shorter the corresponding vectors are, the higher the allowed deviation for the difference in the angles is. The exact calculation is given in lines 20 and 21 of Algorithm 1. If the distance and the angle deviations are within their allowed tolerances, both edges are compatible. Otherwise, the displacements d_r and d_c are reduced which results in increased n_r and n_c values, i.e., the number of rows and columns to be matched. If no iteration reports compatibility of the edges, the edges are incompatible.

```
1: function CompatibleEdges(M_{u,v}, M_{w,x}, STP_u, STP_v, STP_w, STP_x)
2:     m_r := min(|STP_u|, |STP_w|)
3:     m_c := min(|STP_v|, |STP_x|)
4:     displacement_r := max(minimum_{mtc}, min(m_r − 1, (m_r · dynamic_{dsp}) + static_{dsp}))
5:     displacement_c := max(minimum_{mtc}, min(m_c − 1, (m_c · dynamic_{dsp}) + static_{dsp}))
6:     for d_r ∈ {displacement_r, ..., 0} do
7:         n_r := m_r − d_r
8:         for d_c ∈ {displacement_c, ..., 0} do
9:             n_c := m_c − d_c
10:            for i ∈ {1, ..., |STP_u| − n_r} do
11:                for j ∈ {1, ..., |STP_v| − n_c} do
12:                    for k ∈ {1, ..., |STP_w| − n_r} do
13:                        if CompatibleOut(M_{u,v}[i, j], M_{w,x}[i + n_r, j + n_c])) then
14:                            if CompatibleCrs(M_{u,v}[i, j + n_c], M_{w,x}[i + n_r, j]) then
15:                                for l ∈ {1, ..., |STP_x| − n_c} do
16:                                    if CompatibleOut(M_{u,v}[k, l], M_{w,x}[k + n_r, l + n_c])) then
17:                                        if CompatibleCrs(M_{u,v}[k, l + n_c], M_{w,x}[k + n_r, l]) then
18:                                            a_{u,v} :=Angle(p_{u,i}, p_{u,i+n_r}, p_{v,j}, p_{v,j+n_c})
19:                                            a_{w,x} :=Angle(p_{w,k}, p_{w,k+n_r}, p_{x,l}, p_{x,l+n_c})
20:                                            t := max(0, limit_{agl} − min(n_r, n_c))
21:                                            allowed_{agl} := static_{agl} + (t^2 · factor_{agl})
22:                                            if |a_{u,v} − a_{w,x}| ≤ allowed_{agl} then
23:                                                return true
24:                                            end if
25:                                        end if
26:                                    end if
27:                                end for
28:                            end if
29:                        end if
30:                    end for
31:                end for
32:            end for
33:        end for
34:    end for
35:    return false
36: end function
37: function CompatibleOut(d_1, d_2)
38:    return |d_1 − d_2| ≤ (min(d_1, d_2) · dynamic_{out}) + static_{out}
39: end function
40: function CompatibleCrs(d_1, d_2)
41:    return |d_1 − d_2| ≤ (min(d_1, d_2) · dynamic_{crs}) + static_{crs}
42: end function
```

Algorithm 1: Algorithm of SegmentedV1DM to determine the edge compatibility. Algorithm of SegmentedV1DM (see Section 5.3.6.4) to determine the compatibility of two edges $(u, v), (w, x) \in E$ of a graph $G = (V, E)$ based on the distances matrices $M_{u,v}, M_{w,x}$ of their labels and the segmentation points $STP_u, STP_v, STP_w, STP_x$ of their incident vertices $u, v, w, x \in V$. The function Angle returns the angle between two straight lines defined by the first and second pair of passed points.

5.3.6.5 Graph SegmentedVSD1

The SegmentedVSD1 algorithm uses the segmentation of SSEs introduced by the SegmentedV1DM algorithm (see Section 5.3.6.4), but represents all segmentation points explicitly as vertices instead of indirectly at a vertex' label.

This algorithm also creates a complete undirected labeled graph $G = (V, E)$ for each Collection. Each SSE S is segmented. For each segmentation point $p \in STP$ of this segmentation, a vertex $v \in V$ is added to G and to a temporary set of vertices V_{STP}. The label of v consists of two values and p. The first value is the SSE type class code identical to the one described for the StaticV1D1 algorithm in Section 5.3.6.1. The second value is a vertex index within $\{1, \ldots, |STP|\}$ that corresponds to the index of p in STP. For instance, the terminal segmentation points in STP have indices 1 and $|STP|$ respectively. All vertices $u, v \in V_{STP}$ representing the segmentation points of S are connected by edges (u, v). These edges are labeled as intra, with a connection length, and the Euclidean distance between the points of the labels of the incident vertices. The connection length is the absolute value of the difference of the vertex indices of their incident vertices u and v. In other words, it corresponds to the number of segmentation points between the points corresponding to u and v. For instance, let $p_1, p_2, p_3, p_4 \in STP$ be segmentation points and $u, v, w, x \in V_{STP}$ the corresponding vertices. The connection length for edge (u, v) is 1 whereas it is 2 for the edge (u, w).

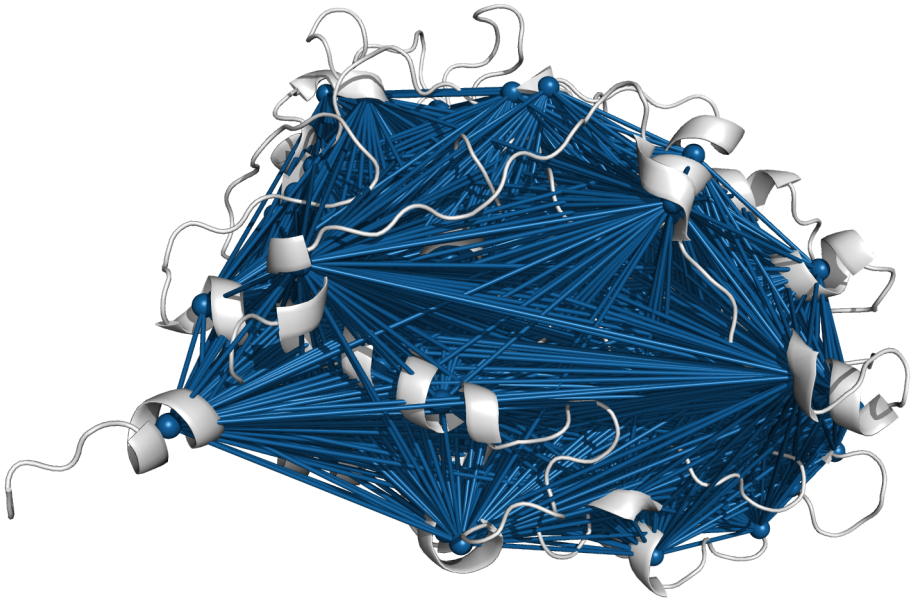
After the intra-labeled edges are created, all vertices V_{STP} are connected by edges to the rest of the vertices V/V_{STP} of G. These edges are labeled as inter and also with the Euclidean distance between the points of the labels of the incident vertices. Summing up, there are two different types of edges in this complete graph G. The inter-labeled edges are labeled with a distance and the intra-labeled edges are additionally labeled with a connection length. Figure 5.9 shows a visualization of the graph for 1gosA@pdb.

Vertex Compatibility

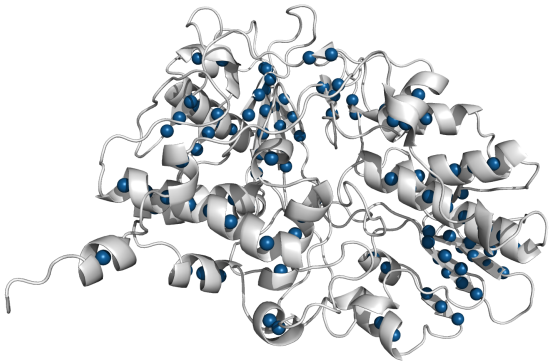
The vertex compatibility is identical to the one described for the StaticV1D1 modeling algorithm.

Edge Compatibility

The compatibility of edges is based on the connection length for pairs of intra edges and the Euclidean distance for all other combinations. For intra-labeled edges, the connection lengths of both edges have to be identical to be compatible. For all other combinations, the compatibility is based on the two distances d_1 and d_2 and which is identical to the one described for the StaticV1D1 modeling algorithm.



(a) Protein and graph



(b) Protein and vertices

(c) Vertices only

Figure 5.9: Visualization of the graph for 1gosA@pdb created by the SegmentedVSD1 algorithm. Visualization of the graph (blue) for 1gosA@pdb (gray) created by the SegmentedVSD1 algorithm (see Section 5.3.6.5). The SSEs used for the visualization and the creation of the graph were assigned with SCOT.

5.3.6.6 Turn Histograms

The algorithm to create turn histograms bins the occurrences of different turns in histograms. For each Collection, we create a separate histogram H in which all occurrences of hydrogen-bonded turns (*normal* and *reverse*) are counted. Similarly to the SSE type class code described in Section 5.3.6.1, we create a turn code based on a turn's category t, length l, and class c (see Equation 5.8).

$$\mathrm{code}_{\mathrm{T}}(t, l, c) := (c \cdot 100) + (l \cdot 10) + t \tag{5.8}$$

For a turn, this code is the key by which it is counted in H. Thus, for each turn, a code is generated and the corresponding number of occurrences in H is incremented by 1.

5.3.7 Model Writer

There are multiple Model Writers available to export the created models for the visualization and/or further analyses. There are three Model Writers available for the export of graphs created by the SegmentedV1DM algorithm (see Section 5.3.6.4). The PyMOL (see Sections 5.3.7.1) and the Chimera (see Sections 5.3.7.2) Model Writer provide an export of the graph itself for visualization purposes. The Segmentation Model Writer (see Section 5.3.7.3) provides the distances between the segmentation points for further analyses.

5.3.7.1 PyMOL

The PyMOL Model Writer exports a graph $G = (V, E)$ created by the SegmentedV1DM algorithm (see Section 5.3.6.4) to a file to be visualized in PyMOL [32]. The file is named by the model ID of the graph and saved to a specified directory. The default file extension is `.py`.

All vertices $v \in V$ are exported as spheres and all edges $e \in E$ as cylinders. The segmentation points STP_v of each vertex $v \in V$ are exported as small spheres. Each consecutive pair of segmentation points $p_j, p_{j+1} \in STP_v$ with $j \in \{1, \ldots, |STP_v| - 1\}$ is connected via a cylinder to indicate their affiliation. This is especially useful to visualize the independent parts of split SSEs (e.g., using the `--split-kinked-sses` argument provided by SCOT, see Chapter 3). The color and diameter of each group of elements, i.e., vertices, edges, segmentation points, and segmentation point edges, can be set separately. Furthermore, each group is assigned a separate layer which allows the visualization of vertices only, for instance.

Figure 5.6 contains several images based on model files created by this PyMOL Model Writer. The figures of all other modeling algorithms (e.g., Figure 5.3) were created with a modified version without the support for segmentation points. All of these figures were created using default values for the diameters and the colors.

5.3.7.2 Chimera

The Chimera Model Writer for the visualization in UCSF Chimera [17] is very similar to the previously described PyMOL Model Writer (see Section 5.3.7.1) besides syntax modifications. There are two main differences. First, the default file extension is `.bld`. Second, all elements are put on a single layer which does not allow to hide a single element group, such as the segmentation points. Apart from that, the set of parameters and the visualization of the graph itself are identical.

5.3.7.3 Segmentation

The Segmentation Model Writer provides the distances between all pairs of consecutive segmentation points in a single file for further evaluation. Let $G = (V, E)$ be a graph created by the SegmentedV1DM algorithm (see Section 5.3.6.4), $v \in V$ be a vertex, S the corresponding SSE, and STP_v the segmentation points v is labeled with. For each consecutive pair of segmentation points $p_j, p_{j+1} \in STP_v$ with $j \in \{1, \ldots, |STP_v| - 1\}$, the Euclidean distance between p_j and p_{j+1} is written to a separate line to the file including the graph ID, the type (0 for helices, 1 for strands) and the classification of S, and the index j. All values are delimited by commas by default.

The written distances of the Segmentation Model Writer are used in the evaluation described in Section 5.6.2.

5.3.8 Comparator

The Comparator compares two data structures and determines their maximum common substructure. In Section 5.3.8.1 the determination of this substructure is explained for graphs and in Section 5.3.8.2 for histograms.

5.3.8.1 Graphs

The measure of similarity of two graphs G_1, G_2, created by any of the graph-based modeling algorithms, is based on their maximum common subgraph (MCS) G_{MCS}. We perform two steps to determine G_{MCS}. First, we create a modular product graph G_P of G_1 and G_2 according to Definition 2.1.11. The edge and vertex compatibilities are defined in the corresponding Sections 5.3.6.1 to 5.3.6.5 with respect to each modeling algorithm. Second, we search for maximal cliques in G_P. Each clique in G_P corresponds to an MCS of G_1 and G_2. This correlation and this procedure was first described by Levi [30].

We use a modified version of the clique-detection algorithm by Tomita et al. [28] which is a further development of the original and well-established algorithm by Bron and Kerbosch [29]. Algorithm 2 shows our modified implementation.

$$m := \max(minimum_{mtc}, (\max(G_1, G_2) \cdot dynamic_{mtc}) + static_{mtc}) \tag{5.9}$$

At first, we calculate the minimum size m for an MCS to be reported according to Equation 5.9. Whenever the current branch of the recursion tree is not able to find a clique of at least size m or the size c of the largest reported clique so far, it is not longer followed (see line 6 of Algorithm 2). The main function takes a graph $G = (V, E)$ and m and calls the recursive function initializing the temporary result R with $\emptyset$, the possible extensions P with V, the excluded set X with $\emptyset$, and c with 0. If P and X are empty, R is reported as a clique. Otherwise, a pivot vertex p is selected from

```
1: function CLIQUE(G, m)
2:     CLIQUE(∅, V, ∅, m, 0)
3: end function
4:
5: function CLIQUE(R, P, X, m, c)
6:     if |R| + |P| > max(m, c) then
7:         if |P| = 0 ∧ |X| = 0 then
8:             c := |R|
9:             Report R as a clique
10:        else
11:            P_X := P ∪ X
12:            p = PIVOT(P_X, P)
13:            P_p := P \ NEIGHBORS(p)
14:            for v ∈ P_p do
15:                R_N := R ∪ v
16:                P_N := P ∩ NEIGHBORS(v)
17:                X_N := X ∩ NEIGHBORS(v)
18:                CLIQUE(R_N, P_N, X_N, m, c)
19:                P = P \ v
20:                X = X ∪ v
21:            end for
22:        end if
23:    end if
24: end function
25:
26: function PIVOT(P_X, P)
27:    c := 0
28:    for v ∈ P_X do
29:        n := |NEIGHBORS(v) ∩ P|
30:        if n > c then
31:            p := v
32:            c := n
33:        end if
34:    end for
35:    return p
36: end function
```

Algorithm 2: Modified clique detection algorithm based on the algorithm by Tomita et al. [28]. The main CLIQUE function in line 1 calls the recursive function in line 5 and initializes the parameters for the recursion. The recursive CLIQUE function processes three vertex sets, namely, the temporary result (R), the possible extensions (P), and the excluded set (X). The function NEIGHBORS returns the adjacent vertices of the passed vertex.

$P \cup X$ with the maximal number of neighbors in P. This step is the main difference between the original algorithm by Tomita et al. and the one by Bron and Kerbosch, i.e., the choice of using a pivot vertex p instead of arnitrarily selecting a vertex $v \in P$. The effect of this pivoting is exemplarily shown in Figure 5.10 where all recursive calls on an input graph are shown for both algorithm variants. For all following steps, please be referred to Algorithm 2. Note that if the maximum clique is not distinct, we report the first detected one only.

5.3.8.2 Histograms

The maximum common similarity of two given histograms H_1, H_2 is a histogram H_M. Let K_1 be the set of keys and c_1 the counter function of H_1, and K_2, c_2 of H_2 analogously. Then, $K_M = K_1 \cup K_2$ is the set of keys and c_M with $\forall k \in K_M : c_M(k) = \min(c_1(k), c_2(k))$ the counter function of H_M. In other words, the histogram H_M contains the minimum value of the counters for each of the keys of H_1 and H_2.

This algorithm does not provide any parameters for the comparison of histograms.

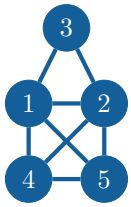
5.3.9 Match Writer

The Match Writers provide a visualization of the MCS, calculated by the Comparator (see Section 5.3.8), based on two graphs created by any of the graph-based modeling algorithms (see Section 5.3.6).

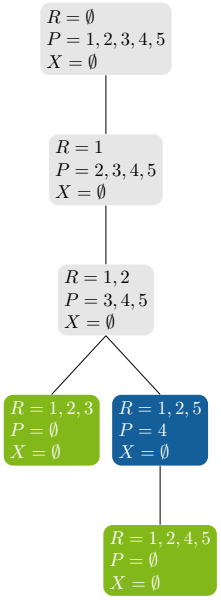
5.3.9.1 PyMOL

The PyMOL Match Writer provides a visualization script for PyMOL [32] to highlight pairs of matched SSEs and aligns the two proteins with respect to these matched SSEs. It processes pairs of graphs $G_1 = (V_1, E_1), G_2 = (V_2, E_2)$ created by any of the graph-based modeling algorithms of Section 5.3.6 and their corresponding MCS $G_{MCS} = (V_{MCS}, E_{MCS})$ determined by the Comparator (see Section 5.3.8.1). For each such triumvirate, a script file with the graph IDs as the file name (e.g., `1gos-2bxr.py`) is created. Each vertex $v_m \in V_{MCS}$ corresponds to a pair of compatible vertices (v_1, v_2) with $v_1 \in V_1, v_2 \in V_2$. Each vertex v_1 and v_2 corresponds to an SSE S_1 and S_2, respectively. We color every such pair of SSEs in the same distinct color. The rest of the protein ribbons of the respective proteins are colored in white and gray respectively, and set semi-transparent. The protein colors and the transparency can be set individually. The distinct colors for the pairs of matched SSEs are fixed. The alignment is based on the minimal RMSD with respect to the terminal residues of each SSE pair. Please note that the alignment is calculated utilizing the `pair_fit` command of PyMOL and explicitly not by the script or SLOT itself.

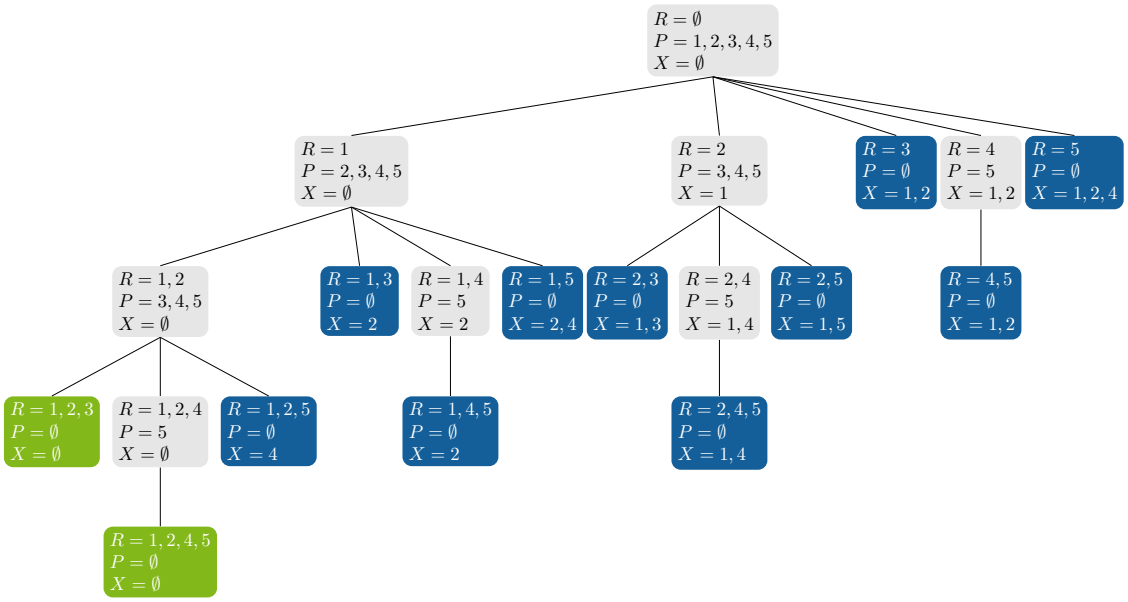
All visualizations of matchings shown in the results section (see Section 5.6), Figure 5.20, for instance, were created using the PyMOL Match Writer.



(a) Input graph



(b) Tree of recursive calls of our algorithm based on Tomita et al. [28].



(c) Tree of recursive calls of the algorithm by Bron and Kerbosch [29].

Figure 5.10: Visualization of the recursive calls for two clique detection algorithms. Visualization of the recursive calls for two clique detection algorithms on the graph shown in (a). In Subfigures (b) and (c), the nodes of unfinished calls are highlighted in gray, nodes of finished calls that reported a maximal clique in green, and nodes of finished calls that did not lead to a maximal clique in blue.

5.3.9.2 Chimera

Similar to the Chimera Model Writer (see Section 5.3.7.2), the Chimera Match Writer is also based on its PyMOL counterpart described in Section 5.3.9.1. It enables the visualization of a matching of

two graphs in UCSF Chimera [17]. In contrast to its counterpart, the default file extension is `.cmd` and the command `match` is used as it the internal command of Chimera to calculate an alignment that minimizes the RMSD between matched atom pairs.

5.3.10 Judge

The Judge judges two models, i.e., graphs or histograms, based on their common substructure determined by the Comparator (see Section 5.3.8), calculates different scores, and provides an output scores file. Section 5.3.10.1 is dedicated to graph-based comparisons and describes the scoring with respect to MCSs. Section 5.3.10.2 covers the scoring of histogram similarities.

5.3.10.1 Graphs

The Judge for graphs examines the MCS $G_{MCS} = (V_{MCS}, E_{MCS})$ of two graphs $G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$. It creates a column-based scores output file containing a line for each such triple. The first two columns contain the graph IDs of G_1 and G_2, followed by their sizes $|V_1|$ and $|V_2|$, the size of the MCS $|V_{MCS}|$, and five scores based on these values. The first two scores set the size of the MCS in relation to each graph's size (see Equation 5.10). Let $s_1 = \mathrm{score}(V_{MCS}, V_1)$ and s_2 defined analogously be these scores. The last three scores are the minimum, the mean, and the maximum of s_1 and s_2 given in Equations 5.11 to 5.13.

$$\mathrm{score}(V_{MCS}, V) := \begin{cases} 0 & |V| = 0 \\ |V_{MCS}|/|V| & \text{otherwise} \end{cases} \tag{5.10}$$

$$\mathrm{score}_{\min}(s_1, s_2) := \min(s_1, s_2) \tag{5.11}$$

$$\mathrm{score}_{\mathrm{avg}}(s_1, s_2) := \frac{s_1 + s_2}{2} \tag{5.12}$$

$$\mathrm{score}_{\max}(s_1, s_2) := \max(s_1, s_2) \tag{5.13}$$

We especially provide all scores for G_1 and G_2 in a single line instead of using two separate lines for each graph to reduce the file size and data redundancy, such as the size of the MCS $|V_{MCS}|$. The final score file also contains a heading in the first line.

5.3.10.2 Histograms

The Judge for histograms is similar to the one for graphs described in Section 5.3.10.1. It evaluates a triple of a histogram H_M containing the similarity of two input histograms H_1 and H_2 (see

Section 5.3.8.2. However, instead of using the sizes of the graphs as the basis for the calculation of the scores, the sums of all counters ($\sum_{k \in K} c(k)$) of each histogram are the basis here.

5.4 Application of SSCMs

All SSCMs were applied with default settings in a serial manner.

5.4.1 DaliLite

We use version v4 of DaliLite [137]. If not otherwise stated, we use the Z-score for our analyses.

5.4.2 LOCK2

The LOCK2 [101] algorithm is used which is the successor of the LOCK [176] algorithm. In contrast to Section 4.6.9, the scores are not normalized with respect to the number of matched SSEs here, but used without any modifications.

5.4.3 TM-align

We use version 20170708 of TM-align [11]. The score used in our analyses corresponds to the TM-score of TM-align.

5.5 Analysis of SSCMs

The parameter optimization of SLOT as well as the analysis of SLOT and three other SSCMs use the area under the receiver-operating-characteristic (ROC) curve (AUC) as the key criterion. ROC curves were plotted using the KNIME [184] ROC Curve node to analyze the sensitivity and specificity (see Equations 5.14 and 5.15). Let P be the number of positives, N be the number of negatives, TP and FP be the numbers of true positives or false positives, respectively. Let TN and FN be defined analogously.

$$sensitivity := \frac{TP}{P} = \frac{TP}{TP + FN} \tag{5.14}$$

$$specificity := \frac{TN}{N} = \frac{TN}{TN + FP} \tag{5.15}$$

A ROC curve plots the sensitivity in relation to the specificity for different parameter values or snapshots. The AUC values were calculated for the resulting ROC curves.

We used the CATH topology and the ECOD subset dataset (see Section 2.3.7) for our analyses. Both datasets consist of related protein domain pairs. Each such pair was defined as active whereas all other possible domain pairings were defined as inactive or decoy in the all-against-all comparison.

5.6 Results

This section covers the comparison and evaluation of the different modeling algorithms and the optimization of the one finally incorporated in SLOT. As SLOT utilizes graphs, aspects with respect to their comparison are also investigated in this section. In addition, several SSAMs are evaluated to select the most suitable one for the requirements of SLOT. The performance of SLOT is compared to three other SSCMs for two different applications, i.e., the retrieval of structurally related domain pairs and the identification of common of *ligand-sensing cores*.

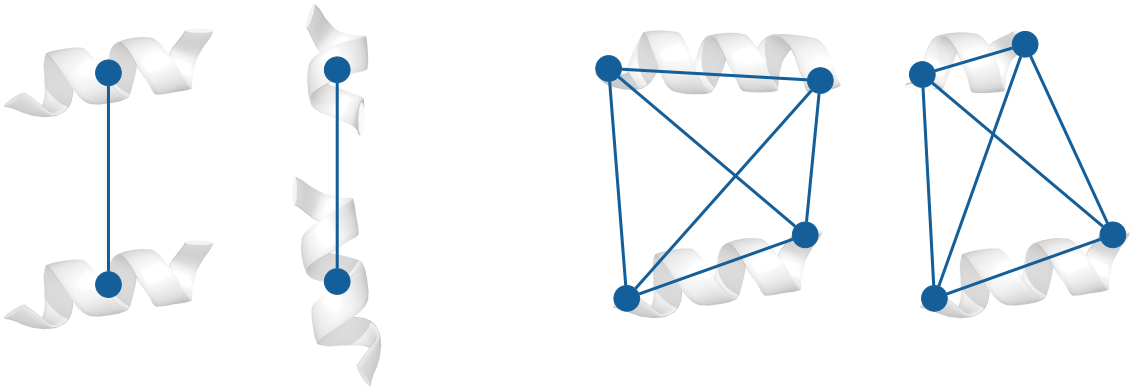
5.6.1 The Progress in the Modeling of SSEs

Although most of the introduced modeling algorithms use a graph to represent the spatial arrangement of SSEs, they differ in how these SSEs and their spatial relationships are represented.

The first modeling algorithm, namely, StaticV1D1 (see Section 5.3.6.1), represents each SSE by a vertex including a code to reflect the SSE's type and class. The representation of the geometry of an SSE is broken down to the coordinates of the Cα atom of its median residue. The vertices representing SSEs with coordinates within a certain distance are connected via edges which are labeled with that distance. The blind spot of this algorithm is the use of a single coordinate to represent each SSE. The coordinate allows a localization of an SSE in the three-dimensional space, but provides no information about its orientation or length. Figure 5.11a depicts an example of two pairs of SSEs, whose corresponding vertices and edges are compatible although the underlying SSEs differ in length and orientation.

The StaticV2D1 algorithm (see Section 5.3.6.2) addresses this challenge by introducing a two-vertex representation of each SSE. The vertices are labeled with the Cα atom coordinates of the terminal residues. However, Figure 5.11b shows the blind spot of this way of representation, i.e., differences in the lengths of similarly oriented SSEs. The two upper helices are oriented and arranged similarly with respect to their respective pair-partner, but due to the differences in lengths the upper right vertices are not part of the matching or the MCS, respectively. However, the image of the upper right helix in Figure 5.11b also reveals another challenge. The Cα atom coordinates of the terminal residues may be positioned at opposite sides of the helix axis.

Therefore, the StaticV3D1 algorithm (see Section 5.3.6.3) addresses this issue by calculating the



(a) StaticV1D1 **(b)** StaticV2D1

Figure 5.11: Visualization of pitfalls in the vertex and edge compatibility defined by the StaticV1D1 and StaticV2D1 algorithm. Visualization of two pairs of adjacent vertices (blue) based on (a) the StaticV1D1 (see Section 5.3.6.1) and (b) the StaticV2D1 algorithm (see Section 5.3.6.2). In (a) the pairs of vertices are compatible although the orientation of the corresponding SSEs differ. In (b) the effect of SSE lengths of otherwise similarly arranged SSEs is demonstrated.

mean of the coordinates of the backbone atoms instead of using the coordinates of the Cα atoms. It represents each SSE by three vertices with the coordinates obtained from the terminal residues and the central residue. However, the higher accuracy in representing an SSE and the support for SSE sub-matching is achieved at high costs, i.e., the higher runtime (see Section 5.6.7). The higher number of vertices representing an SSE leads to larger graphs with respect to the number of vertices and edges. This consequently results in larger and usually denser product graphs, which increases the runtime. Nevertheless, the results on different datasets were not promising for any of the algorithms discussed so far, which is why further algorithms have been developed.

The SegmentedV1DM reduces the number of required vertices and also introduces more criteria to the definition of edge compatibility. The geometry of an SSE is represented by segmentation points. Figure 5.12 shows a helix in 1c02A@pdb and the corresponding segmentation points following the helix axis.

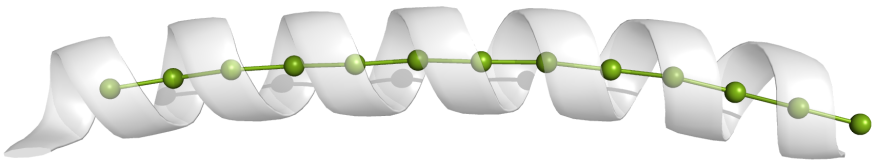
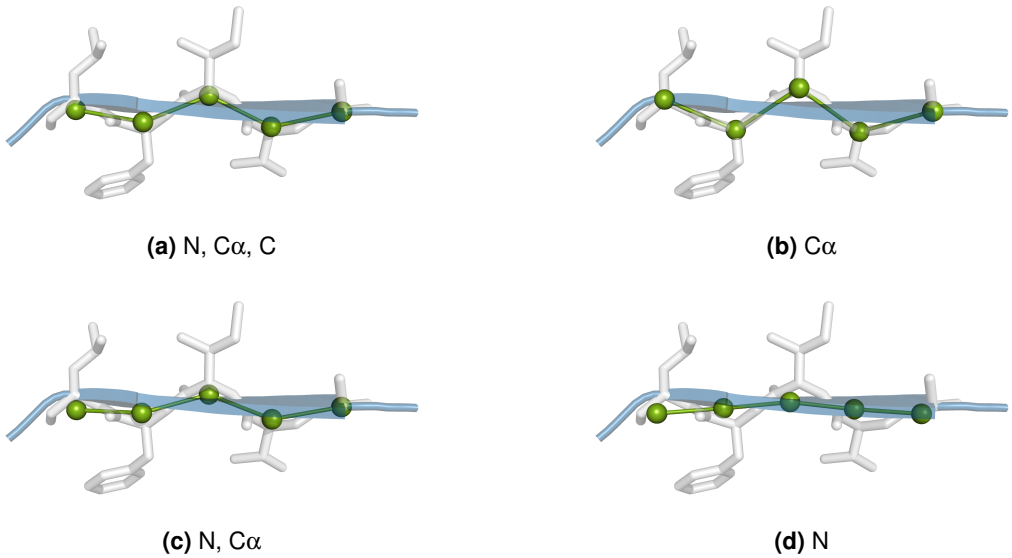


Figure 5.12: Visualization of segmentation points representing the bent shape of a helix. Visualization of the segmentation points following the geometric shape of a SCOT-assigned bent right-handed α-helix in 1c02A@pdb, residues 135–163.

We have developed four different ways to obtain the segmentation points in strands which are exemplarily shown for a strand in 1fj2A@pdb in Figure 5.13. Similar to helices, we initially used the mean of the coordinates of the backbone atoms. This resulted in a *zigzag*-like positioning of

the segmentation points with respect to the strand axis. Solely using the coordinates of the Cα atoms considerably increases this effect. The use of the mean based on the coordinates of the N and the Cα atoms smoothens the representation. Finally, calculating the mean based on the coordinates of the N atoms of two neighboring residues almost perfectly follows the strand axis. Thus, both, helices and strands, can be rotated around their axes without a significant influence on their segmentation points. For instance, if the strand in Figure 5.13b is rotated along the strand axis by 180°, the segmentation point trace would change from a W- to an M-shaped orientation. In contrast, this rotation does not have any effect on the segmentation points in Figure 5.13d.



(a) N, Cα, C **(b)** Cα

(c) N, Cα **(d)** N

Figure 5.13: Visualization of a strand and four different segmentation point representations. Visualization of a SCOT-assigned strand in 1fj2A@pdb defined on residues 18–22 and four different segmentation point placements. Each placement is based on the coordinates of different backbone residue atoms. The protein structure of the strand is shown by the residues as sticks and the backbone trace as cartoon in blue. The straight lines between the segmentation points are added to emphasize deviations from the ideal trace.

To circumvent the drawback of the StaticV1D1 algorithm shown in Figure 5.11a, the required number of segmentation points for an SSE to be represented by a vertex should be at least 2.

The SegmentedVSD1 algorithm (see Section 5.3.6.5) is the last graph-based modeling algorithm. Instead of representing the geometry of an SSE implicitly by segmentation points, each such point is explicitly represented by a vertex. There are several reasons motivating this way of modeling. First, the higher the number of vertices used to represent an SSE the more detailed the score based on the size of an MCS is. Second, the SegmentedV1DM algorithm does not support the matching of multiple SSEs on different parts of a single SSE. Third, the selected (segmentation point/distance matrix) indices for a vertex to create edge compatibility may differ between different compatibility checks for the edges to its neighbors. However, the considerable increase of the graph sizes created by the SegmentedVSD1 increased the runtimes for the comparisons from minutes/hours to weeks (see Section 5.6.7).

The Turn Histograms (see Section 5.3.6.6) are not limited to SSEs but consider all turns of a protein. In contrast to the other algorithms, they represent the arrangement of SSEs implicitly

by counting the turns that forge the SSE arrangement. Therefore, an alignment of two matching proteins cannot be calculated.

In summary, the SegmentedV1DM algorithm led to the most convincing results in general and was, therefore, chosen for SLOT.

5.6.2 Selecting an SSAM based on Segmentation Point Distances

One of the weak spots of the StaticV2D1 and StaticV3D1 algorithms is that the distances between the vertices representing an SSE can differ to a huge extend which hampers the matching per se and the sub-matching in particular. For the SegmentedV1DM algorithm, uniform distances between the segmentation points are of upmost importance for the matching procedure. They facilitate lower allowed deviations which result in lower runtimes and, more importantly, increase the accuracy of the found matching procedure. Therefore, we analyzed the distances between two neighboring segmentation points within an SSE. Figures 5.14 and 5.15 show the boxplots for these distances obtained for the X-ray representatives dataset for the seven different SSAMs introduced in Chapter 3 and evaluated in Chapter 4.

For the remainder of this section, helix classes refer to right-handed helix classes.

The most common SSEs in a folded protein chain are α-helices and strands. Therefore, their accurate representation is of major importance. The use of the SCOT SSE classification with split SSEs at kinked positions leads to the most stable segmentation point distances for α-helices together with ASSP. However, for most of the SSAMs, the distances for these helices show the lowest deviations compared to other helix classes and also strands. The boxplots for 3_{10}-helices show much higher deviations in general. Nevertheless, the SCOT-assigned 3_{10}-helices are still the second most stable among all SSAMs. For the π-helices, we see that the combination of a hydrogen bonding pattern and geometric criteria provides a comprehensive and reliable classification (see Section 4.6.3.2) which results in the lowest deviations for SCOT. The boxplot for STRIDE is solely based on two segmentation point distances and, thus, lacks statistical significance.

When it comes to strands, SCOT is the second most stable SSAM whereas ASSP, which was similarly well-performing on α-helices, shows the highest deviations here. Although the segmentation point distances for mixed and PPII helices classified by SCOT show the highest deviations (see Figure 6.4, appendix), SCOT assigns the overall geometrically most stable SSEs. More explicitly, no other SSAM shows a similarly good overall performance. This underlines the benefits of SCOT for the structural alignment of proteins once again, which was already shown for LOCK2 in Section 4.6.9.

The boxplots for SCOT show that deviations for the segmentation point distances are relatively low and, therefore, underline that their calculation is accurate.

For the remainder of this chapter, the option to split SSEs at kink positions (`--split-kinked-sses`) is always used for the SCOT-based SSE annotations.

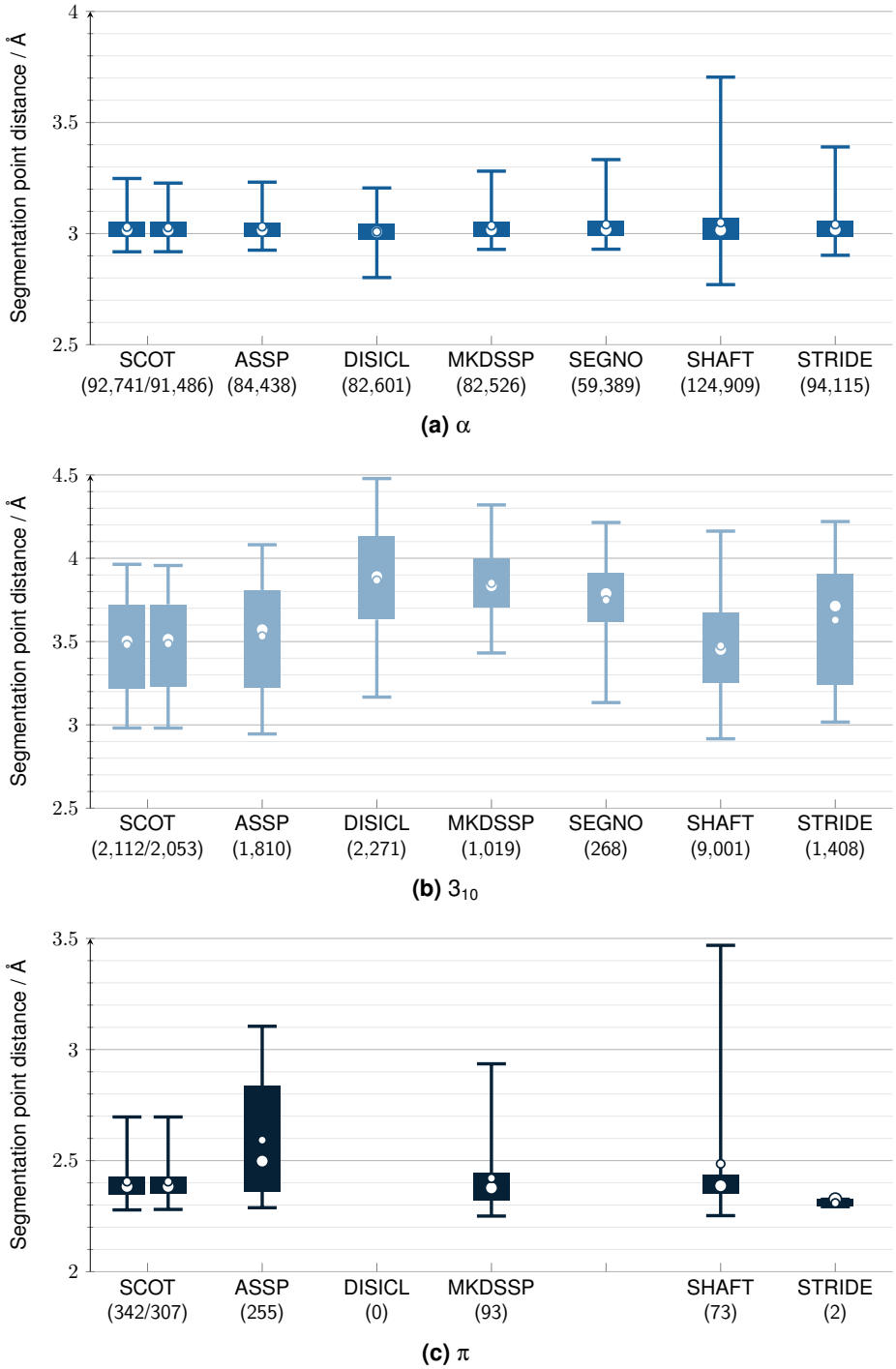


Figure 5.14: Boxplots showing the segmentation point distances in right-handed helices for different SSAMs. Boxplots showing the segmentation point distances between two neighboring points in right-handed α-, 3_{10}-, and π-helices obtained for the X-ray representatives dataset for different SSAMs. For SCOT, these distances are given for the standard settings (left) and for the split helices at kink positions (right). The numbers of analyzed distances are given in parentheses. The assignment of π-helices is not supported by SEGNO. The median is indicated by a big and the mean by a small white dot. Outliers were omitted in favor of a concise visualization.

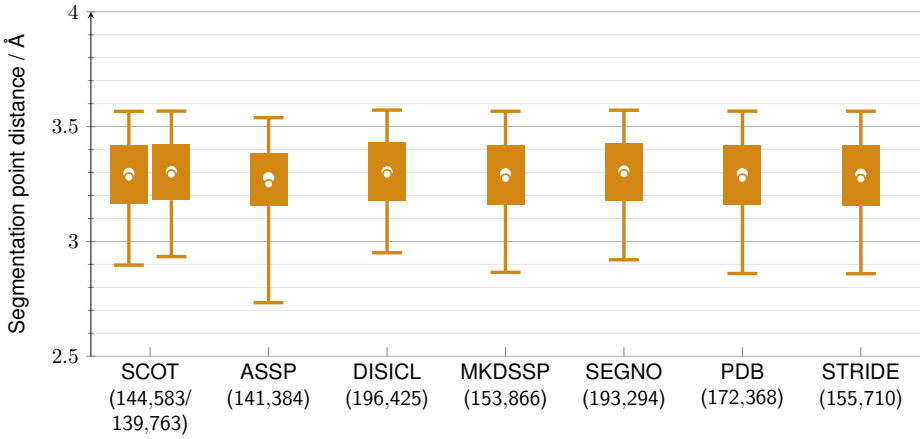


Figure 5.15: Boxplots showing the segmentation point distances in strands for different SSAMs. Boxplots showing the segmentation point distances between two neighboring points in strands of β-sheets obtained for the X-ray representatives dataset for different SSAMs (see Section 4.4). For SCOT, these distances are given for the standard settings (left) and for the split strands at kink positions (right). The median is indicated by a big and the mean by a small white dot. Outliers were omitted in favor of a concise visualization.

5.6.3 Parameter Optimization

The SegmentedV1DM modeling algorithm (see Section 5.3.6.4) and the Comparator for graphs (see Section 5.3.8.1) require several parameters to be set. The parameters of the Comparator and the vertex compatibility were set to intuitively appropriate values. We required an MCS to be of size at least 3 or of size at least $30\,\%$ of the size of the larger input graph. The two parameters for the vertex compatibility were set in such a way that the numbers of segmentation points of two vertices did not differ by more than $40\,\%$ with respect to the higher number of segmentation points. For all other parameters, e.g., the parameters for the edge compatibility, an optimization was performed.

This optimization was performed with the help of the CATH topology dataset (see Section 2.3.7.1) with annotated SSEs by SCOT. We optimized the parameter values to maximize the calculated scores and the separation from decoy pairs for as many active pairs as possible. All parameter sets and the resulting runtimes, the scores, and the AUC values are listed in Table 5.3.

Each set was evaluated with respect to the $\mathrm{score}_{\mathrm{min}}$, the runtime, and by visual inspection. The visual inspection ensured that found positive domain pair matches show the appropriate alignments. Furthermore, the parameters to be adjusted for the next set and their new values were also derived by this inspection. Parameter set 7 was chosen as it is on par with the best performing parameter sets with respect to accuracy. In contrast, however, it requires a very low runtime. It is highlighted in blue in Table 5.3. These parameter values were used to obtain all of the following results of this chapter. If not stated differently, the following evaluations of SLOT were based on the $\mathrm{score}_{\mathrm{min}}$.

Parameter	Set 1	Set 2	Set 3	Set 4	Set 5	Set 6	Set 7	Set 8
$static_{mtc}$	0	0	0	0	0	0	0	0
$dynamic_{mtc}$	0.3	0.3	0.3	0.3	0.3	0.3	0.3	0.3
$minimum_{mtc}$	3	3	3	3	3	3	3	3
$static_{STP}$	0	0	0	0	0	0	0	0
$dynamic_{STP}$	0.6	0.6	0.6	0.6	0.6	0.6	0.6	0.6
$static_{out}$	1	1	1	1	1	1	1	1
$dynamic_{out}$	0.2	0.15	0.2	0.15	0.15	0.12	0.15	0.12
$static_{crs}$	1	1	1	1	1	1	1	1
$dynamic_{crs}$	0.2	0.15	0.2	0.15	0.15	0.12	0.15	0.12
$static_{agl}$	25	30	30	25	20	25	25	20
$limit_{agl}$	1	1	1	1	1	1	1	1
$factor_{agl}$	5	5	5	5	5	5	5	5
$static_{dsp}$	1	1	1	1	1	1	1	1
$dynamic_{dsp}$	0.3	0.3	0.25	0.25	0.25	0.25	0.2	0.2
$minimum_{STP}$	2	2	2	2	2	2	2	2
Runtime	39 min48 s	29 min19 s	53 min54 s	22 min58 s	20 min56 s	17 min54 s	11 min59 s	8 min39 s
AUC $score_{min}$	0.8447	0.8455	0.8459	0.8489	0.8418	0.8466	0.8433	0.8263
AUC $score_{max}$	0.7972	0.8019	0.7952	0.8074	0.8028	0.8081	0.8114	0.8027
AUC $score_{avg}$	0.8409	0.8427	0.8429	0.8458	0.8374	0.8435	0.8398	0.8235

Table 5.3: Parameter sets and the resulting runtimes, the scores, and the AUC values of the parameter optimization. For all sets, the minimum number of segmentation points for an SSE to be represented by a vertex was set to 2 for both, helices and strands.

5.6.4 Hunting for Domain Pairs

After we had selected an SSAM (see Section 5.6.2) and had optimized the parameters (see Section 5.6.3), we evaluated the performance of SLOT with the help of the CATH topology and the ECOD subset dataset. Both datasets comprise protein domain pairs (see Section 2.3.7 for more details). We analyzed SLOT in comparison to LOCK2, DaliLite, and TM-align. Due to the ability of SLOT and LOCK2 to use external SSE annotations, the SSE annotations by SCOT and by MKDSSP were used as input. Table 5.4 contains the AUC and the corresponding runtimes for single thread execution. The runtimes are given for single thread execution because not all SSCM used herein provide parallelization support. In consequence, the runtimes for SLOT are based on the parallel execution using 30 threads. These runtimes were converted to the runtime for the single thread execution. In contrast to Section 4.6.9, the scores reported by LOCK2 are not normalized with respect to the number of matched SSEs here.

SLOT performs best with SCOT instead of MKDSSP with respect to the AUC. The good performance of DaliLite is comprehensible as it is the method behind the CATH database. However, it suffers from the highest loss of performance among the SSCMs for the transition to the ECOD subset dataset (0.1319). This finding indicates that DaliLite is highly optimized for the scope of the CATH database.

	AUC			Runtime		
SSCM	**CATH**	**ECOD**	$\lvert\Delta\rvert$	**CATH**	**ECOD**	**RF**
SLOT (SCOT)	0.9244	0.8849	0.0395	11 h 29 min 0 s	1 h 13 min 0 s	9.4384
SLOT (MKDSSP)	0.9156	0.8674	0.0482	9 h 13 min 0 s	1 h 0 min 30 s	9.1405
LOCK2 (SCOT)	0.9198	0.8739	0.0459	12 h 0 min 39 s	4 h 1 min 7 s	2.9888
LOCK2 (MKDSSP)	0.9187	0.8842	0.0345	10 h 9 min 2 s	3 h 25 min 13 s	2.9678
DaliLite	0.9316	0.7997	0.1319	8 h 42 min 18 s	2 h 9 min 43 s	3.9734
TM-align	0.9556	0.9265	0.0291	7 h 57 min 25 s	2 h 0 min 9 s	3.9734
Turn Histograms	0.8686	0.8593	0.0093	29 s	13 s	2.2308

Table 5.4: The AUC and the runtimes for each SSCM obtained for the CATH topology and the ECOD subset dataset. The differences in the AUC values for the two datasets is given in column $|\Delta|$. The runtimes are given for single thread execution. Column RF contains the ratio between the runtimes for each dataset.

The boxplots in Figure 5.16 also underline the coherence between DaliLite and the CATH database. DaliLite calculates the lowest scores (0 in most cases) for the decoy domain pairs. However, a good separation of actives from decoy was observed for all SSCMs, except for the Turn Histograms.

In comparison to the AUC values in Table 5.4, the boxplots more clearly demonstrate the benefits of SCOT- compared to MKDSSP-based SSE annotations for both, SLOT and LOCK2. Particularly, the mean and the average values for the decoy pairs were significantly lower when using SCOT.

One striking characteristic of the results of SLOT compared to those of the other SSCMs is that a considerable number of domain pairs defined as decoy were scored as high as active pairs. This is indicated by the comparatively high upper whisker of the boxplot for SLOT. Therefore, we analyzed such pairs to answer the question, of whether these were false positives by SLOT or false negatives by the other SSCMs. Table 5.5 and Table 6.15 of the appendix contain excerpts of the results for the queries 1cc8A00@cath and 4f01B01@cath, respectively. In both tables, the scores calculated by SLOT are at least 0.75 which classified the corresponding domain pairs as actives according to the boxplots in Figure 5.16a. In contrast, the scores for such pairs by all other SSCMs were low and in the value range obtained for the decoy domain pairs (see Figures 5.16c, 5.16f, and 5.16e). The question becomes even more interesting due to the fact that some of the matched domains differed on their architecture level considering their CATH-IDs, such as 1cc8A00@cath (3.30.70.100) and 1s2oA02@cath (3.90.1070.10).

We used the PyMOL Match Writer (see Section 5.3.9.1) of SLOT to superpose the matchings for the domains of Table 5.5 in Figure 5.17a and of Table 6.15 of the appendix in Figure 6.5a of the appendix. Both figures show clear structural similarities of 6 and respectively 7 matched SSEs among all protein domains, which were solely found by SLOT. One possible explanation are the non-sequential similarities of the domains, which is depicted by the topology diagrams for each query and one associated target domain. The corresponding rows in Table 5.5 and Table 6.15 of the appendix are highlighted in blue.

The runtimes given in Table 5.4 reveal that the factor between the runtimes for the CATH topology and the ECOD subset dataset of SLOT is by far the highest (9.4384) compared to the ones of the

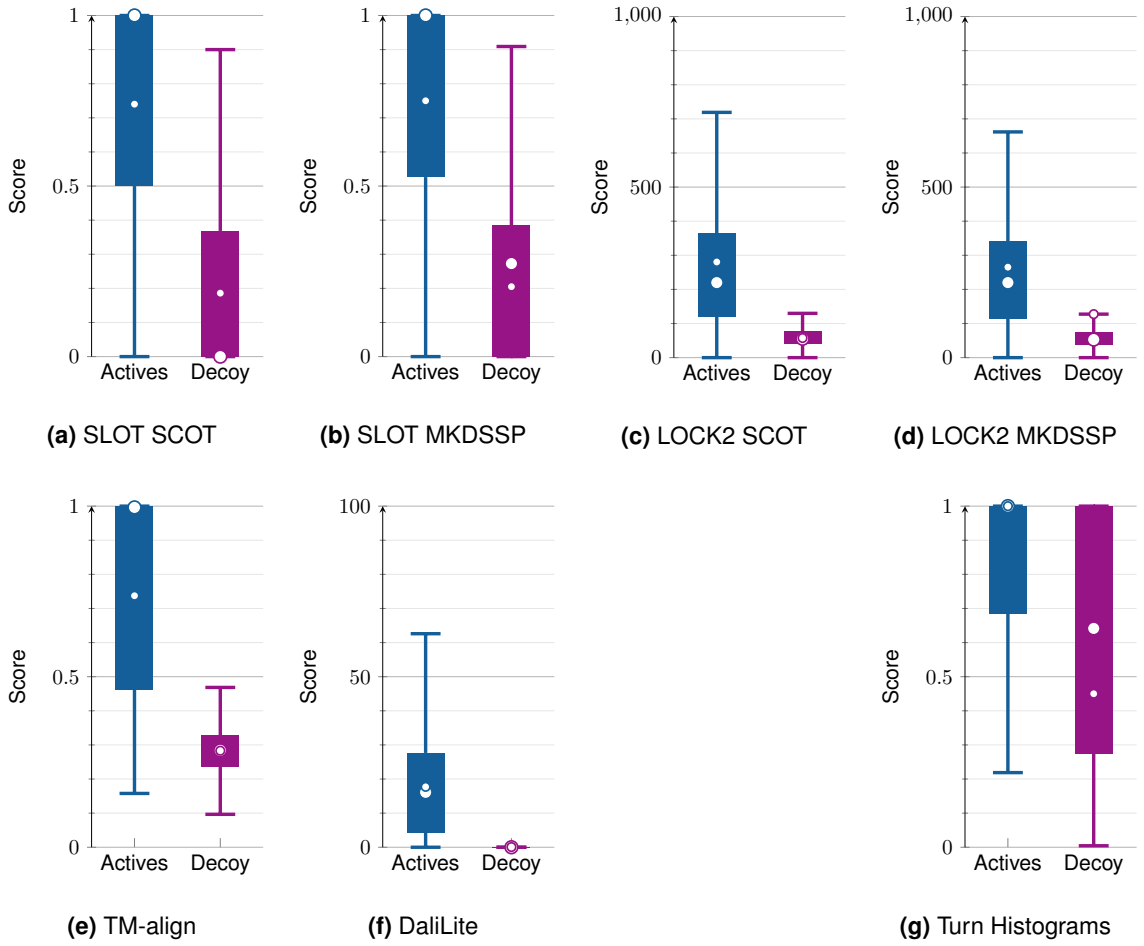


Figure 5.16: Boxplots showing the score distributions of each SSCM for the CATH topology dataset. Boxplots showing the score distributions of each SSCM for the actives (blue) and decoy (purple) obtained for the CATH topology dataset. SLOT and LOCK2 were applied using SCOT with SSEs split at kink positions and MKDSSP-based SSE annotations. The median is indicated by a big and the mean by a small white dot. Outliers were omitted in favor of a concise visualization.

other SSCMs. Assuming that the correlation observed for SLOT, i.e., a higher degree of similarity leads to an increase of the runtime, also holds true for other SSCMs, this high factor also points toward the ability of SLOT to find yet unrevealed structural similarities in the CATH topology dataset. This also hints at the different methodologies behind the two datasets.

The Turn Histograms were introduced in Section 5.3.6.6. They are discussed separately as they do not focus on SSEs alone, but take the turns of the entire protein into account. Due to the character of histograms in general, the runtimes were below half a minute for both datasets. But in contrast to the other SSCMs, the file I/O massively dominated their runtimes ($> 80\,\%$). The AUC was 0.8686 for the CATH topology dataset and 0.8593 for the ECOD subset dataset. Compared to the other SSCMs, the Turn Histograms had the lowest $|\Delta|$. In addition, it is the only SSCM presented herein for which a clear separation of actives and decoy was not possible (see Figure 5.16g).

PDB-ID	CATH-ID	SLOT		LOCK2		DaliLite	TM-align
		SCOT	MKDSSP	SCOT	MKDSSP		
1j5yA02	3.30.1340.20	0.7500	0.7143	136.35	130.63	4.50	0.6207
1xppD00	3.30.1360.10	0.7500	0.8333	141.50	142.01	4.50	0.5601
2gjvA00	3.30.2000.10	0.7500	0.8571	102.79	104.92	3.40	0.5575
1v4pC01	3.30.980.10	0.7500	0.8333	124.27	112.05	0.00	0.5475
4bndA02	3.30.1240.20	0.7500	0.8571	136.20	133.37	3.40	0.5255
4acvB00	3.30.2000.30	0.7500	0.8333	95.55	78.23	2.10	0.5068
1s2oA02	3.90.1070.10	0.7500	1.0000	119.41	118.67	2.40	0.5022
4mo0A00	3.30.780.10	0.7500	0.8571	142.45	144.02	3.20	0.5013
2pwwA00	3.30.310.10	0.7500	0.5556	76.37	81.19	2.00	0.4944
1vkwA02	3.40.109.30	0.7500	1.0000	136.42	125.64	3.60	0.4784
1i4jA00	3.90.470.10	0.7500	0.8333	103.92	76.38	0.00	0.4530
4g6tA00	3.30.1460.10	0.7500	0.6250	101.77	100.39	0.00	0.4518
3a2eA00	3.30.430.20	0.7500	0.8571	90.19	83.23	0.00	0.4094
2hzmA02	2.20.140.20	0.7500	0.8333	104.78	101.49	0.00	0.3854
3d4eA01	3.30.1450.10	0.8750	0.8333	67.48	74.63	0.00	0.3086
3vz9B00	3.30.457.50	0.7500	0.5714	82.79	84.06	0.00	0.3018

Table 5.5: List of protein domain pairs with high scores obtained by SLOT for query domain 1cc8A00@cath. List of domain pairs with high scores obtained by SLOT using SCOT-based SSE annotations for the query domain 1cc8A00@cath (3.30.70.100), for which low scores were obtained by all other SSCMs. The attribution of high and low is based on the boxplots shown in Figure 5.16. A topologically distant domain with respect to the query is highlighted in blue and the corresponding topology diagrams are presented in Figure 5.17.

In summary, in comparison to LOCK2, DaliLite, and TM-align, SLOT was the only SSCM able to find domain pairs in the CATH topology dataset independent of the sequence direction of the matched SSEs.

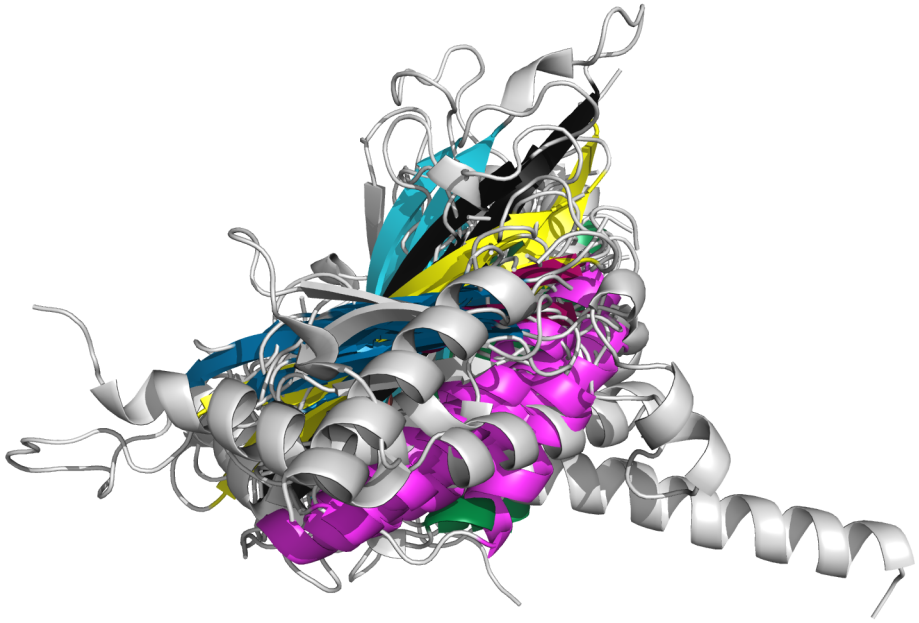
5.6.5 Searching for Ligand-Sensing Cores

The search for *ligand-sensing cores* was performed on both versions the LSC query target dataset (see Section 2.3.8.3), i.e., chains and pockets. We searched for matches among the targets for the *ligand-sensing core* queries of LSD1 (2ejrA@pdb) and APT1 (1fj2A@pdb).

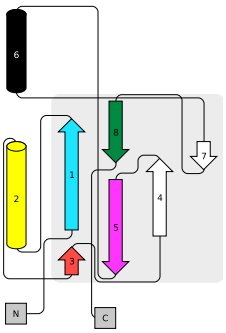
5.6.5.1 Chains

All SSCMs were used to search the targets for similarities to the two queries. Table 5.6 contains the scores and ranks for each of the two query chains.

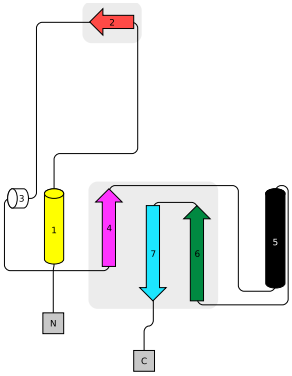
If the identity matches are excluded from the results, all SSCMs reported the targets for LSD1 among their first 10 out of 3,614 ranks. TM-align particularly reported all targets for LSD1 in front



(a)



(b) Topology diagram of 1cc8A00@cath.



(c) Topology diagram of 1s2oA02@cath.

Figure 5.17: Superposition and topology diagrams for the query domain 1cc8A00@cath and a topologically distant domain. (a) Superposition according to SLOT with SCOT-based SSE annotations of the query domain 1c88A00@cath and the domains listed in Table 5.5. The topology diagrams are given for the query (b) and a high scored (matched), but topologically distant domain (c). The diagrams were created using Pro-origami [185]. Matched SSEs are highlighted in the same color in all figures.

of all other matches. However, this trend slightly differs for the APT1 query. Although the targets were still within the top $1.5\,\%$ of the results for LOCK2, DaliLite, and TM-align, the ranks according to SLOT dropped massively. Considering that LOCK2 and DaliLite provide scores that are not normalized with respect to the size of the protein, i.e., the larger the input structures and their structural similarity the higher the score, their scores indicate that the input sizes differ. In fact, the number of SCOT-assigned SSEs which were modeled in 1fj2A@pdb (14) was lower compared to 2ejrA@pdb (43). Furthermore, the matchings calculated by SLOT contained 11 respectively 23 SSEs. Thus, an appropriate filtering of the results and/or the use of another score, e.g., $\mathrm{score}_{\mathrm{avg}}$,

Query	Target	SLOT		LOCK2		DaliLite		TM-align	
PDB-ID	PDB-ID	Rank	Score	Rank	Score	Rank	Score	Rank	Score
2ejrA	1gosA	9	0.5349	10	692.08	11	31.3	5	0.7431
2ejrA	1gosB	6	0.5349	9	737.25	9	31.6	4	0.7491
2ejrA	2bxrA	11	0.5349	4	739.84	3	35.4	2	0.8279
2ejrA	2bxrB	3	0.5581	6	739.70	5	35.3	3	0.8276
1fj2A	1k8qA	743	0.4074	21	360.69	53	13.7	18	0.7054
1fj2A	1k8qB	820	0.3929	28	358.61	54	13.7	19	0.7050

Table 5.6: The ranks and scores for the queries 2ejrA@pdb and 1fj2A@pdb and the targets with a similar *ligand-sensing core* calculated by the SSCMs on the LSC query target chains dataset. The ranks are given with respect to each query.

would provide different rankings. Furthermore, the chance of finding structural similarities increases with fewer criteria to fulfill, i.e., the number of SSEs to be matched. Therefore, the comparatively lower scores calculated by SLOT must not necessarily correlate with a poorer performance, but with structural similarities that were not identified by other the SSCMs. This was already observed for the CATH domain pairs discussed in Section 5.6.4.

The results by SLOT were filtered for each query separately. For LSD1, only matches for which the score was above 0 were kept, resulting in 29 matches (see Table 5.7). The matches of APT1 additionally had to contain at least 10 SSEs due to the high number of high-scoring chain pairs. More filtering criteria led to the exclusion of APT1 whose rank was 106 out of 142 matches (see Table 6.16, appendix).

We added enzyme commission (EC) numbers to all of these matches to highlight the diversity of the enzyme-catalyzed reactions of the corresponding target chains. All chains which were found as similar to LSD1 are oxidoreductases (EC number 1.-.-.-) that differ with respect to their substrates. For APT1, the found matches contained members of all top level codes except 7, i.e., oxidoreductases (1), transferases (2), hydrolases (3), lyases (4), isomerases (5), and ligases (6).

In addition to the EC numbers, UniProt [186] accession numbers were used to identify known bioactive molecules in version 21 of the ChEMBL database [187]. The ChEMBL contained bioactivity data for 6 targets of LSD1 and 11 targets of APT1.

To assess the structural similarity among the found matches of each query, they were superimposed using the SLOT PyMOL Match Writer script (see Section 5.3.9.1). The superpositions are shown in Figure 5.18. The very high structural similarity is evident for both sets of query and similar target chains.

5.6.5.2 Pockets

In contrast to the previous analyses based on the chains, the search for the target pockets was performed with SLOT only. Although the LSC target pocket dataset contained three pockets for

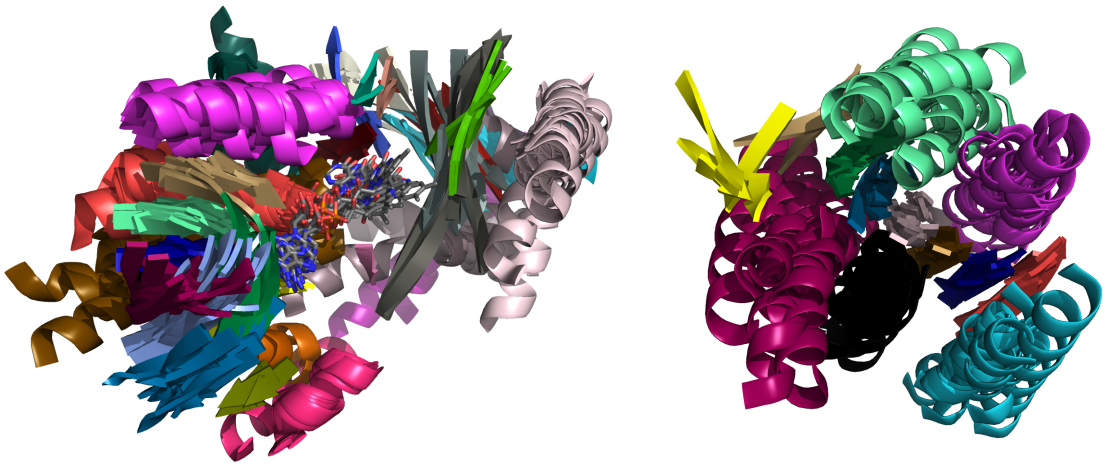
Rank	Query		Target					Matching	
	PDB-ID	SSEs	PDB-ID	EC No	UniProt	ChEMBL	SSEs	SSEs	Score
1	2ejrA	43	2ejrA	1.-.-.-	O60341		43	43	1.0000
2	2ejrA	43	1rsgA	1.5.3.11	P50264		42	26	0.6047
3	2ejrA	43	2bxrB	1.4.3.4	P21397	●	35	24	0.5581
4	2ejrA	43	1h83A	1.5.3.11	O64411	●	36	24	0.5581
5	2ejrA	43	2bxrA	1.4.3.4	P21397	●	36	23	0.5349
6	2ejrA	43	1gosB	1.4.3.4	P27338	●	35	23	0.5349
7	2ejrA	43	1gosA	1.4.3.4	P27338	●	33	23	0.5349
8	2ejrA	43	1s3eA	1.4.3.4	P27338	●	39	22	0.5116
9	2ejrA	43	1zovA	1.5.3.1	P23342		31	17	0.3953
10	2ejrA	43	4h1bA				30	15	0.3488
11	2ejrA	43	3rp8A		A6T923		32	15	0.3488
12	2ejrA	43	3axbA		Q9YCJ0		34	15	0.3488
13	2ejrA	43	3aljA	1.14.12.4	Q988D3		33	15	0.3488
14	2ejrA	43	2bcgG		P39958		35	15	0.3488
15	2ejrA	43	2h88A	1.3.5.1	Q9YHT1		46	16	0.3478
16	2ejrA	43	1pn0A	1.14.13.7	P15245		48	16	0.3333
17	2ejrA	43	4rekA	1.1.3.6	P12676		31	14	0.3256
18	2ejrA	43	2qa1A	1.14.13.-	Q93LY7		42	14	0.3256
19	2ejrA	43	4rg3A	1.14.13.22	C0STX7		37	13	0.3023
20	2ejrA	43	4ntcA		E9RAH5		27	13	0.3023
21	2ejrA	43	4h7uA	1.1.99.29	Q3L245		36	13	0.3023
22	2ejrA	43	3pl8A	1.1.3.10	Q7ZA32		38	13	0.3023
23	2ejrA	43	2gqwA	1.18.1.2	Q52437		37	13	0.3023
24	2ejrA	43	2aqjA		P95480		39	13	0.3023
25	2ejrA	43	1rp0A		P15245		24	13	0.3023
26	2ejrA	43	1mo9A	1.8.1.5	Q56839		41	13	0.3023
27	2ejrA	43	1kdgA	1.1.99.18	Q01738		32	13	0.3023
28	2ejrA	43	1fl2A	1.8.1.-	P35340		28	13	0.3023

Table 5.7: Complete list of the matches calculated by SLOT for the query chain 2ejrA@pdb. Complete list of the matches calculated by SLOT for the query chain 2ejrA@pdb in the LSC query target chains dataset for which the score was not 0. The information of these matches were extended by the EC numbers (EC No), the UniProt accession numbers [186] (UniProt), and the availability (●) of bioactive molecules in the ChEMBL database [187]. Rows highlighted in blue contain query target chain pairs of the *ligand-sensing cores*.

each target (a total of 10,818 pockets), we focussed on the primary pockets.

We observed similar relative rankings of the target's primary pockets compared to the analysis with respect to the chains for both queries (see Table 5.8).

However, the results of ATP1 contain more than 3,500 matches with a higher score than the matches of the target pockets. Approximately 800 of these matches contain more matching SSEs (> 5) compared to the matches of the target pockets. The best approximately 50 matches were assigned a score of at least 0.7 which is twice as high as the highest score of the target pockets.



(a) LSD1 **(b)** APT1

Figure 5.18: Superposition of the LSD1 and APT1 query chains and all targets for which a bioactive molecule was available. Superposition of the LSD1 query chain 2ejrA@pdb and all target chains listed in Table 5.7 (a) and of the APT1 query chain 1fj2A@pdb and all target chains listed in Table 6.16 of the appendix (b) for which a bioactive molecule was available in the ChEMBL database [187].

Query		Target		Matching		
PDB-ID	SSEs	PDB-ID	SSEs	Rank	SSEs	Score
2ejrA0	19	1gosA0	27	5	10	0.3846
2ejrA0	19	1gosB0	28	39	10	0.4255
2ejrA0	19	2bxrA0	19	2	11	0.5789
2ejrA0	19	2bxrB0	27	4	11	0.4074
1fj2A0	9	1k8qA0	14	3,252	5	0.3571
1fj2A0	9	1k8qB0	15	4,700	5	0.3333

Table 5.8: Ranks and scores for the primary pocket of each of the queries and their target's most similar pockets. Ranks and scores for the primary pocket of each of the queries, i.e., 2ejrA@pdb and 1fj2A@pdb, and their target's most similar pockets calculated by SLOT on the LSC query target pockets dataset. The ranks are given with respect to each query.

The first match with a score of less than 0.5 was ranked at position $1{,}188$.

5.6.6 On the Uniqueness of the MCS

Maximal clique detection algorithms, such as the one by Tomita et al. [28], determine all maximal cliques in a graph G. This means that the reported cliques must not necessarily be of the same size. In the example depicted in Figure 5.10, the reported cliques are of size 3 and 4. However, our modified version of the algorithm, only reports the first detected maximum clique, which is the one of size 4 in the given example. It is sufficient to report only the maximum clique because our scoring solely takes the size of the clique/MCS into account and we are interested in the maximal possible structural similarity of two graphs. Nevertheless, the MCS of two given graphs is not

necessarily unique.

We analyzed the uniqueness of the MCS on five datasets, i.e., the LSC query target dataset using chains and pockets, the CATH topology and superfamily datasets, and the ECOD subset dataset. We disabled our optimization of the algorithm and reported all MCSs instead of one representative MCS.

Figure 5.19 shows the boxplots for the number of MCSs and their sizes for all pairs of graphs, for which at least one MCS of a size of at least 3 was found (blue boxplots). For the green boxplots, $\mathrm{score}_{\min}$ had to be at least 0.5 additionally. This means that the size of the MCSs for two graphs G_1 and G_2 is at least half the size of the smaller of G_1 and G_2. Thus, these pairs can be considered as relevant matches for further analysis.

Although the number of MCSs for relevant matches (green) was significantly lower, there was still a considerable number of MCSs detected. As domains contain relatively few SSEs, the sizes of multiple MCSs obtained for these datasets were small as well. This usually facilitates the discovery of multiple MCSs. Nevertheless, the results on the LSC chain dataset demonstrates that large MCS sizes not necessarily guarantee uniqueness.

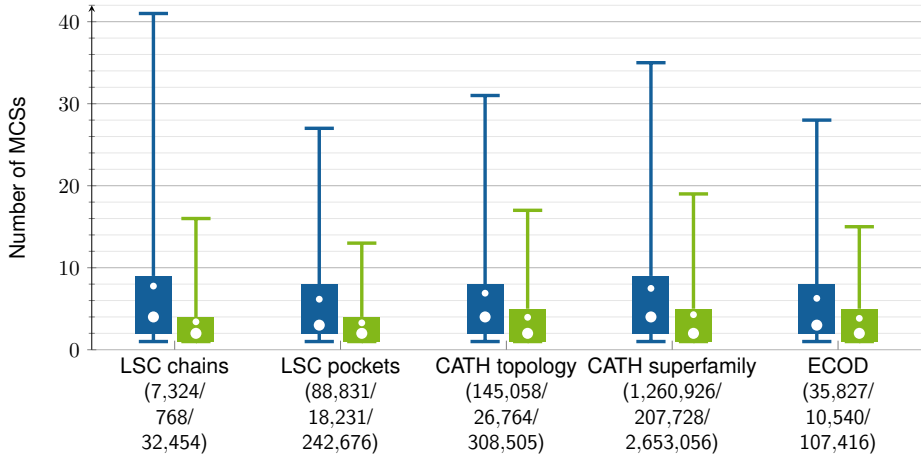
There are several reasons for the MCS not to be unique. First, the high deviations in the parameters required to find interesting matches allow for a high degree of flexibility in the arrangement of the SSEs. Figure 5.20 shows 4 out of 8 matchings based on multiple MCSs for the same pair of proteins and their graphs. This flexibility can especially be seen in Figures 5.20a and 5.20b. Here, the purple colored strand of one protein (bottom) is matched to different strands in the target protein. This usually happens in densely packed strands of β-sheets.

Second, using relative distances in contrast to exact coordinates leads to mirrored matches. In Figure 5.20c, for instance, the magenta colored helices are matched in a mirrored fashion considering the matched strands as the axis of symmetry. The relative distances, the applied thresholds, and the allowed displacement in particular are the reasons for this match in particular and mirrored matchings in general. In this figure, none of the proteins can be rotated and/or translated in such a way, that all identically colored SSE pairs are superimposed. However, it is still a valid match as the relative distances of each graph are compatible to the ones of the other.

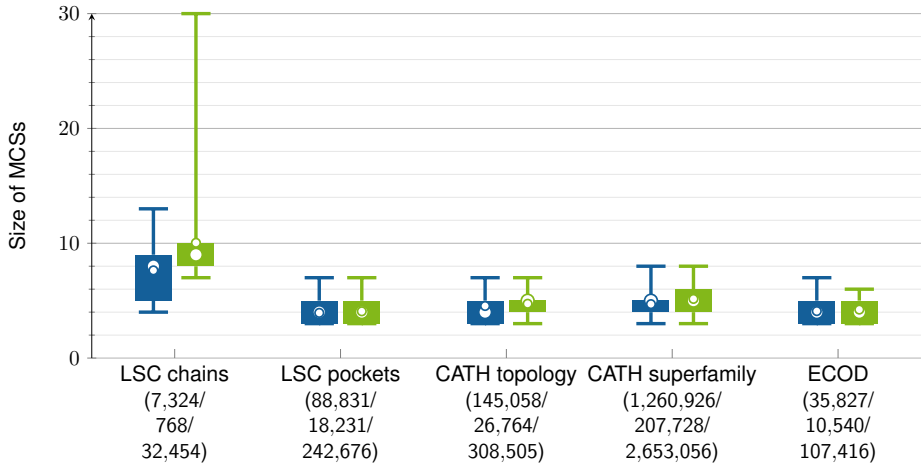
Finally, the matching given in Figure 5.20d also shows the ability of SLOT to match SSEs independent of their sequence direction. In contrast to the other matchings, all SSEs of each matched pair have opposite sequence directions. This can be particularly seen with the help of the arrow-representation of the strands.

5.6.7 Runtime and Memory Consumption

The subgraph isomorphism problem for two given graphs is NP-complete in general [31]. On the one hand, this means that no polynomial time algorithm is known and, on the other hand, that the determination the MCS is a time-demanding procedure.



(a) Number of MCSs



(b) Size of MCSs

Figure 5.19: Boxplots showing the number of MCSs and their sizes for different datasets. Boxplots showing the number of MCSs (a) and their sizes (b) obtained by SLOT for five datasets with SCOT-assigned SSEs which were split at kink positions. The comparison of LSC chains und pockets is query-target based whereas the domains of both CATH and the ECOD subset datasets were compared in an all-against-all fashion. The boxplots in blue are based on the MCSs of pairs for which at least one MCS was found (first numbers in parentheses) whereas additionally for the green boxplots the $\mathrm{score}_{\mathrm{min}}$ (see Equation 5.11) for these pairs had to be at least 0.5 (middle numbers in parentheses). The last numbers in parentheses are the total numbers of comparisons for each dataset. The median is indicated by a big and the mean by a small white dot. Outliers were omitted in favor of a concise visualization.

We performed a query-target-based search for the protein chains of both *ligand-sensing core* datasets *hidden* in the X-ray representatives dataset. We analyzed the runtimes using each of the six different modeling algorithms introduced in Section 5.3.6. In addition, we further investigated two variants of the SegmentedV1DM algorithm with respect to the edge compatibility, i.e., bottom up and top down. Bottom up means from high to low displacement resulting in low to high index range lengths (see Algorithm 1 lines 6 and 8). Top down is defined analogously from low to high

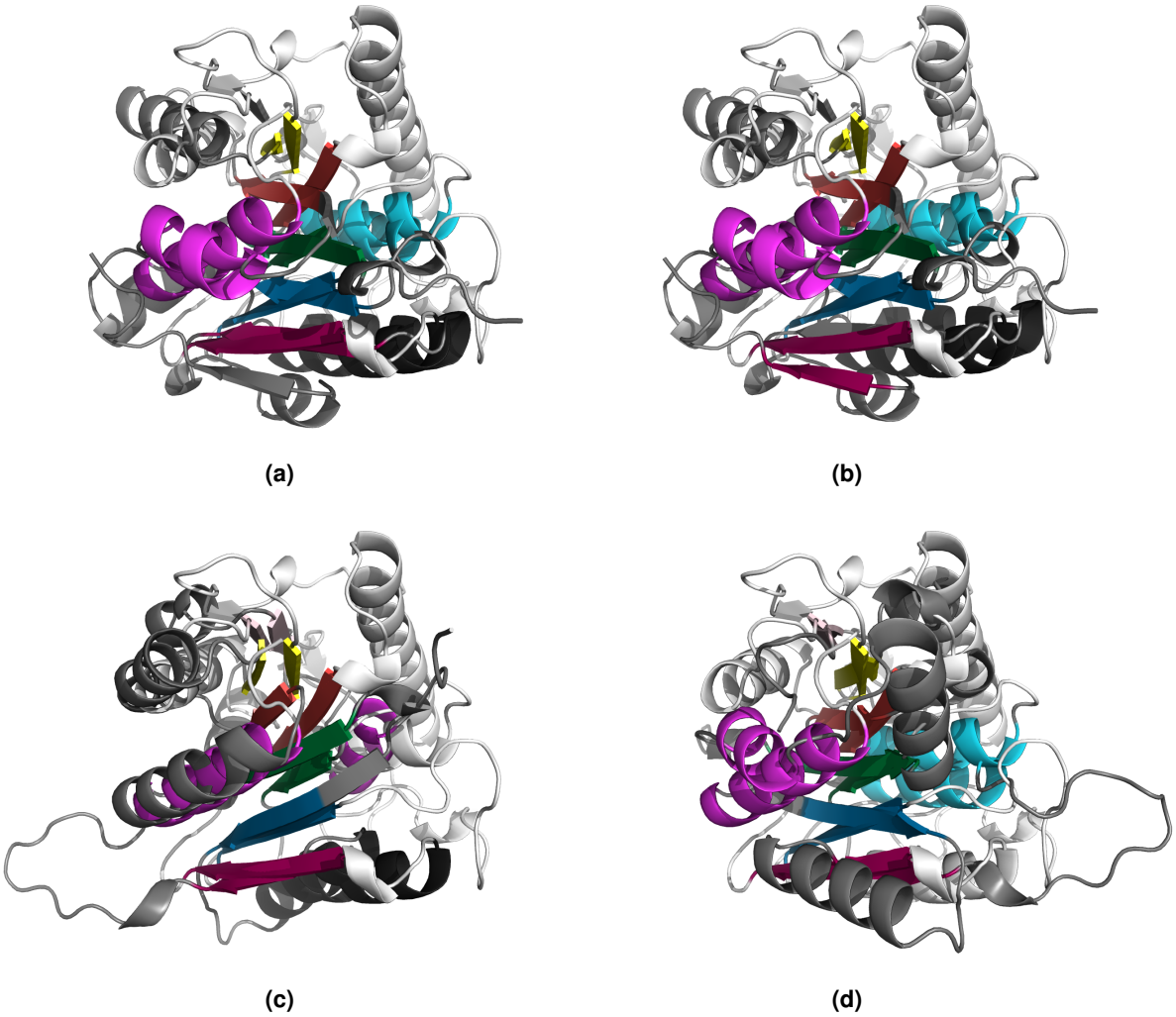


Figure 5.20: Equally sized matches for two chains. Four out of eight equally-sized (8 SSEs) matches for 1fj2A@pdb (white) and 2qipA@pdb (gray). The orientation of the query structure 1fj2A@pdb is fixed in all subfigures. Matched SSEs are highlighted in the same color. The colors are fixed with respect to 1fj2A@pdb.

displacement. As a remainder, the runtimes are based on a parallel execution using 30 threads. We used a maximum edge connectivity distance of 30 Å and $dynamic_{dst} = 0.15$ and $static_{dst} = 1$ (Å) for all Static modeling algorithms. For the SegmentedVSD1 algorithm, we used the same distance parameters. For the SegmentedV1DM algorithm, we used the parameters obtained in the parameter optimization described in Section 5.6.3. The parameters for the Comparator were set to $minimum_{mtc} = 3$, $static_{mtc} = 0$, and $dynamic_{mtc} = 0.3$ for all modeling algorithms.

Table 5.9 shows the runtimes with respect to each modeling algorithm and/or variant. For those algorithms that did not finish within 6 h, the progress at that point of time is given in percentages instead.

For the graph-based modeling algorithms, the SegmentedV1DM algorithm outperforms all other with respect to runtime. The runtimes for the Static algorithms alone demonstrate the exponential

Modeling algorithm	Section	Runtime/Progress
StaticV1D1	5.3.6.1	14 %
StaticV2D1	5.3.6.2	3 %
StaticV3D1	5.3.6.3	0 %
SegmentedV1DM top down	5.3.6.4	23 min 12 s
SegmentedV1DM bottom up	5.3.6.4	22 min 35 s
SegmentedVSD1	5.3.6.5	0 %
Turn histograms	5.3.6.6	59 s

Table 5.9: Runtimes for the comparison of the queries against the targets of the LSC query target chains dataset using different modeling algorithms. If a run had not completed after 6 h, it was aborted and its progress at that point of time is given instead. Our optimized version of the clique detection algorithm by Tomita et al. was used for all graph-based modeling algorithms. All runtimes are based on parallel execution using 30 threads.

runtime behavior with respect to the input size (number of vertices $|V|$). The transition to complete graphs by the SegmentedV1DM algorithm influences the construction of the product graph to a huge extend, too. Before, two pairs of compatible non-adjacent vertices led to an edge in the product graph G_P without any restriction. In a complete graph, these vertices are adjacent and their edges have to be compatible in order to result in the corresponding edge in G_P. Thus, G_P is less dense in most cases leading to lower runtimes for the determination of its maximum clique.

The two variants for the SegmentedV1DM algorithm are very close with respect to their runtimes. The worst case time for both variants is identical as all displacements have to be considered whether it is from high to low or vice versa. Their algorithmic difference tackles the construction of the product graph by the use of the vertex and edge compatibility criteria. This step contributes to only a very small fraction to the overall runtime. Combined with the fact that the results are in each and every way identical, whether to use the bottom up or the top down variant is negligibly. However, as we had to make a choice at some point and one would expect that it is more promising to match fewer numbers of segmentation points than higher. In addition, the number of matched segmentation points has no influence on the final score. Therefore, we preferred to use the bottom up variant.

We also analyzed the benefits of the algorithm by Tomita et al. [28] over the one by Bron and Kerbosch [29]. Our optimizations regarding the reported cliques described in Section 5.3.8.1 were used for both algorithms. Both algorithms were used to compare the graphs created by the SegmentedV1DM algorithm for the CATH topology dataset (see Section 2.3.7.1) and the LSC query target chains dataset (see Section 2.3.8.3). The runtimes are shown in Table 5.10.

Although the differences in runtimes for the LSC query target chains dataset are almost negligible, the benefits of the algorithm by Tomita et al. come into play for the CATH topology dataset. One possible reason may be the high structural similarities within the CATH topology dataset which was already discussed in Section 5.6.4. However, the cases in which the algorithm by Bron and Kerbosch is faster than the one by Tomita et al. are rare. In such a case, both algorithms select the same vertices in the same order for their recursion calls. The only difference is that the algorithm

Clique algorithm	CATH topology	LSC query target chains
Tomita	21 min 35 s	22 min 35 s
Bron Kerbosch	25 min 1 s	23 min 18 s

Table 5.10: Comparison of the runtimes using the algorithm by Bron and Kerbosch and the algorithm by Tomita et al. Direct comparison of the runtimes of our optimized versions of the clique detection algorithms by Bron and Kerbosch and the one by Tomita et al. (see Section 5.3.8.1). The runtimes were achieved based on graphs of the SegmentedV1DM modeling algorithm. All runtimes are based on parallel execution using 30 threads.

by Bron and Kerbosch selects these vertices in this specific order by chance without the additional overhead to determine the optimal one every time as realized in the algorithm by Tomita et al.

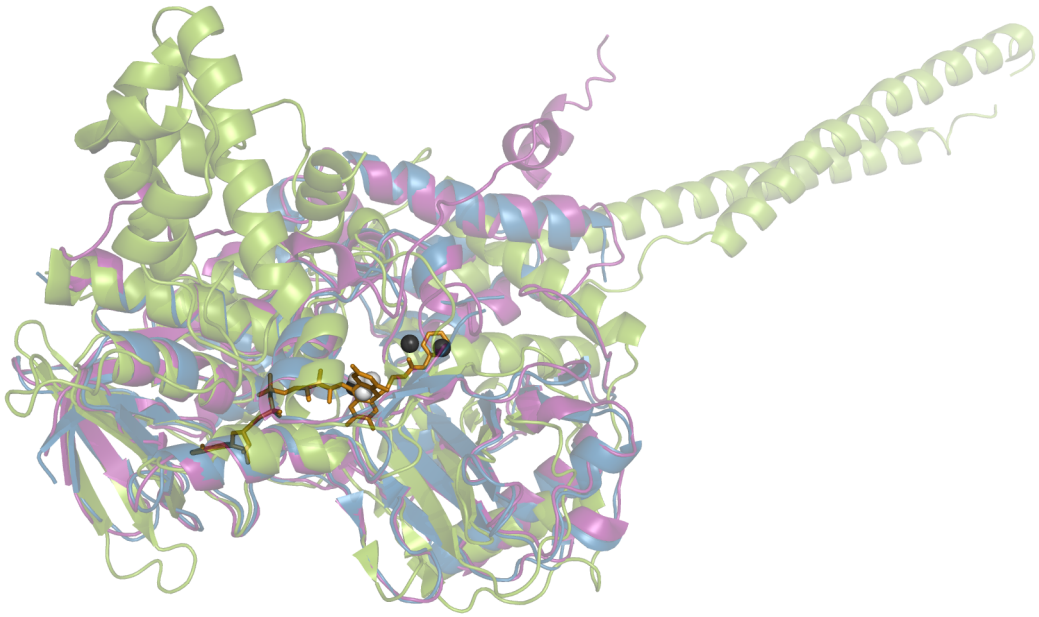
The memory consumption is high compared to the other tools presented herein, i.e., SCOT (see Section 3.5.5) and SNOT (see Section 4.6.10). It took about on average 4 GB to process the LSC query target chains dataset. A comparison to other SSCMs is difficult as some, e.g., LOCK2, only support the input of two proteins for a single pairwise comparison. Thus, the comparison in an all-against-all manner had to be scripted and proteins had to be loaded multiple times. SLOT avoids such redundant I/O operations to reduce the overall runtime, which leads to the comparatively high memory consumption. DaliLite supports the input of multiple proteins and also the parallelization of the comparisons. However, the input has to be pre-processed in a separate step. Nevertheless, its memory footprint is negligible small.

5.6.8 Automated Pocket Detection

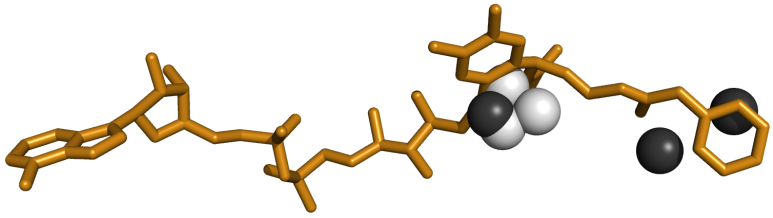
The creation of the LSD query target dataset based on pockets (see Section 2.3.8.3) required an automated pocket detection algorithm. An exhaustive evaluation of different binding site detection approaches can be found in the doctoral thesis by Christiane Ehrt [90]. Based on this evaluation, we selected P2Rank [46] and highlight its superior performance in a comparison to Ligsite^{CS} [188]. Ligsite^{CS} detected pockets were used by the CavBase algorithm integrated in the retired Relibase [92].

We automatically searched for the pockets in the LSD1 dataset (see Section 2.3.8.1) using the default settings for both tools. Figure 5.21 shows the superposition of the three proteins and the detected primary pockets as spheres.

The superposition emphasizes the high structural similarity among all proteins which should contribute to consistent locations of the detected pockets. While the centers of the pockets detected by P2Rank are in close proximity (white spheres) with a maximum pairwise distance of 1.74 Å, the situation massively differs for those detected by Ligsite^{CS} (black spheres). Here, the pairwise distances are 3.5 Å, 7.6 Å, and 10.35 Å. The highest distance was observed between the pockets of the structurally highly similar proteins MAO-A (2bxr@pdb) and MAO-B (1gos@pdb).



(a)



(b)

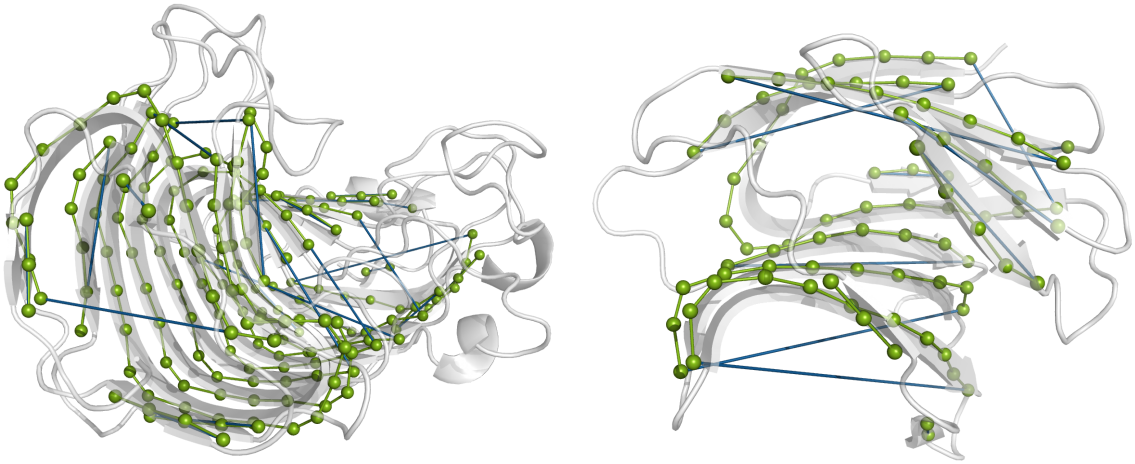
Figure 5.21: Visualization of the detected primary pockets by Ligsite^{CS} and P2Rank. (a) Visualization of the centers of detected primary pockets by Ligsite^{CS} [188] (black spheres) and P2Rank [46] (white spheres) for the proteins of the LSD1 dataset. 2ejrA@pdb is highlighted in green, 1gosA@pdb in purple, and 2bxrA@pdb in blue. The ligand and the co-factor of 1gosA@pdb are exemplarily shown (orange). (b) Enlarged visualization of the ligand, the co-factor, and the detected pockets.

5.7 Discussion

We introduce SLOT as an innovative SSCM for the discovery of similar structural arrangements of SSEs. It is on par with other SSCMs with respect to the performance and the runtime, but is able to discover a yet unrevealed layer of structural similarity. It provides several similarity measures (e.g., $\mathrm{score}_{\mathrm{min}}$ or the number of matched SSEs) for an application-dependent filtering of the results. In addition, the possibilities to export the graph-based models and the matchings for visualization tools, such as PyMOL, allow an in-depth analysis of the results. Its parameters enable a high degree of individualization and are initially set to our optimized values for an exemplary dataset for an immediate usage.

We have developed six different algorithms for the representation of SSEs and their arrangements. The five graph-based algorithms are the result of an intensive optimization with respect to the

match quality and runtime. We focussed on graphs for the representation of the SSEs and their arrangements as they provide the highest flexibility for the representation of objects and their spatial relationships. Furthermore, as the search for common *ligand-sensing cores* has been rarely studied, we wanted to avoid pitfalls from the start, e.g., limitations due to the chosen data structure. The best performing modeling algorithm for SLOT is SegmentedV1DM (see Section 5.3.6.4). In contrast to the majority of the SSCMs, it utilizes segmentation points instead of vectors to mimic the geometry of an SSE. Vectors are not applicable to represent the geometry of bent SSEs. A way to overcome this issue is to additionally determine and store the maximum BDA of an SSE. This, however, does neither take the position of the bend into account nor does it support multiple bends in an SSE. Multiple bends especially occur in strands (see Figure 5.22). The segmentation points used herein mimic the geometry of an SSE including any bend or conformation in general.



(a) 1k5cA@pdb **(b)** 1is3A@pdb

Figure 5.22: Examples of two protein chains with multiply bent strands assigned by SCOT and their geometrical representation by the segmentation points. For comparison purposes, vectors from the first to the last segmentation point of each SSE are visualized by blue lines.

We analyzed several procedures for the placement of segmentation points. The procedures for strands mainly differ in their selections of backbone atoms (see Figure 5.13). Using solely Cα atoms led to the highest deviations from an ideal axis among the presented backbone atom selections. This is particularly problematic for the matching procedure. Given two identical SSEs of which one is rotated by $180°$, their segmentation points would end up at opposite sides of the ideal axis. The matching of these points requires high thresholds for the distance and angle compatibility determinations which also negatively affect the differentiation between positives and negatives. For instance, in right-handed α-helices, the Radius is ≈ 2.3 Å leading to a distance of 4.6 Å between the segmentation points on opposite sides of the ideal axis (see Table 6.2, appendix for the geometric characteristics of right-handed α-helices). Thus, other SSCMs, e.g., MIRAGE-align [171], use multiple atoms to smoothen this effect. For the calculation of the segmentation points in helices, we use the N and the C atoms of a specific number of residues per point. In contrast to LOCK2, we use a class dependent instead of a fixed number (4) of residues. We observed that the class-specific segmentation point placement result in lower deviations from the axis and lower deviations in the segmentation point distances as well.

An additional major benefit of the segmentation points is that they allow a non-sequential matching of SSEs. The corresponding mechanism is implemented in the creation of the distance matrices. Here, the sequential dependency is eliminated by always selecting the sequence direction that leads to the lowest distances. An example of a non-sequential match by SLOT is shown in Figure 5.20d. The last benefit to be discussed is that the segmentation points also allow for a fine-grained sub-matching of SSEs. This feature can be fine-tuned by several parameters.

The calculation of the entire distance matrices positively influences the runtimes as they reduce the required and oftentimes redundant geometric calculations during the compatibility checks. Only the calculation of the relative angle is done on-the-fly as the distance criteria limit the number of such calculations to a huge extend.

In contrast to all available SSCMs except LOCK2, SLOT is able to use user-defined SSE annotations in the standardized and widely supported PDB file format. It enables users to throw off the shackles of preset SSE assignments. In Section 4.6.9 we have shown that in the structural alignment with LOCK2, our classification by SCOT outperforms the state-of-the-art SSAMs, among which DSSP is the most prominent one. Here, we underline the benefits of the SCOT-based SSE classification with respect to the segmentation point distances. Although these distances based on SCOT-assigned SSEs are not the most stable distances for all SSE types and classes, they are among the best. Thus, no other SSAM provides such a good overall performance. This is particularly worth mentioning considering the high number of supported SSE classes by SCOT. PPII and right-handed mixed are the classes for which the highest deviations in the segmentation point distances were observed. In this context be reminded, that SCOT rarely assigns right-handed mixed helices which led to only a handful of segmentation point distances (108) compared to SEGNO (30,273). In addition, in Section 4.6.5 we already discussed that many PPII helices assigned by the other SSAMs correspond to SCOT-assigned strand structures which explains their higher stability with respect to the segmentation point distances here. Our analyses also underline the benefits of splitting kinked SSEs for the structural alignment. The variations of the segmentation point distances improved when this feature of SCOT was used.

The structural similarity of two graphs is represented by their MCS. The calculation of a MCS is based on the determination of the maximum clique in an appropriately defined product graph. We investigated two different algorithms, namely, the one by Bron and Kerbosch [29] and the one by Tomita et al. [28], and also introduced some optimizations. There are other algorithms available including algorithms for the general problem of determining the MCS, e.g., by Suters et al. [189], and also algorithms for special cases, such as for outer planar graphs by Akutsu and Tamura [190]. However, we focussed on the modeling of SSEs and also decided to implement all aspects (parsing, modeling, comparing, and writing) of SLOT by ourselves to have our hands on any detail. Therefore, it was not feasible to implement a wide range of different algorithms for the determination of the MCS and evaluate them with respect to our requirements.

One important aspect to be mentioned is that the maximum clique of the product graph and the corresponding MCS of the input graphs are not necessarily unique. Our current scoring scheme solely considers the size of an MCS and, therefore, it is sufficient to report only one of multiple equally sized MCSs. We evaluated whether the occurrence of multiple MCSs is rare or common

with the help of different datasets. Multiple MCSs were found even for matches for which a score was reported that usually indicates an active match (≥ 0.5). Most interestingly, these matches can contain 30 SSEs and more. One possible consequence is, that either the allowed deviations are not sufficiently strict or the arrangement of SSEs alone does not provide sufficient information for distinct matches. In Section 4.6.8 we have discussed the flexibility of protein structures with respect to the consistency of SSAM assignments with the help of NMR and X-ray ensembles. The allowed deviations defined by the chosen parameter values are required to cope with fluctuations in the SSE assignments. Thus, the level of abstraction of SSE arrangements allow multiple alignments or matches. Nevertheless, multiple MCSs might still prove valuable. It is possible to derive a matching purity from the number of detected MCSs for a given representative matching. But several questions remain. First, how high is the impact of this additional information? Second, how high is the influence on the runtime? And third, is the additional information worthwhile the expected additional runtime?

In 1965, Moon and Moser showed that a graph $G = (V, E)$ can contain up to $3^{|V|/3}$ cliques in general [191]. In other words, solely reporting all cliques of a graph requires exponential time. 25 years later, Garey and Johnson [31] proved that the subgraph isomorphism problem is NP-complete. This fact is also reflected by the measured runtimes for using each of the different modeling algorithms introduced herein. The SegmentedV1DM algorithm was the only graph-based modeling algorithm, for which the execution finished within 6 hours. Most astonishing is the fact, that using the StaticV3D1 and the SegmentedVSD1 algorithm, the progress after this duration of time was below $1\,\%$. Our evaluation also revealed the reduced computation time for the algorithm by Tomita et al. in comparison to the algorithm by Bron and Kerbosch.

One of the main challenges in the development of SLOT was and still is the lack of an appropriate dataset.

Two common *ligand-sensing cores* have been proposed and exploited for new drugs in the structure-based design. They involve a total of 5 proteins. This small amount of proteins is barely sufficient for the optimization of a tool when faced with almost $150{,}000$ publicly available protein structures in the PDB in February of 2019. We addressed this challenge by using the CATH topology and the ECOD subset dataset. Each dataset's individual classification was used to define pairs of similar domains. However, apart from these pairings, SLOT identified several domains sharing structural similarities which remained invisible to the other SSCMs. These domains showed high structural similarities but differed significantly in their topologies. However, the defined domain pairings do not reflect the structural similarities of other domain pairings solely detected by SLOT, which had an influence on the evaluation of SLOT's performance. Such hidden structurally similar domain pairs were regarded as false positives and, therefore, negatively influenced the ROC curve and the corresponding AUC for SLOT. Therefore, it had an impact on the parameter optimization and also discriminates SLOT against the other SSCMs.

The main scope of SLOT is to identify new common *ligand-sensing cores* in the PDB. To answer the question of whether SLOT is able to fulfill this task, we used the LSC query target datasets to analyze the performance of SLOT in comparison to other SSCMs. All SSCMs reported high structural similarities for the targets of the queries LSD1 (2ejrA@pdb) and APT1 (1fj2A@pdb).

Moreover, although the target dataset consisted of more than 3,500 representative protein chains from the PDB, all chains with common *ligand-sensing cores* with LSD1 were ranked among the top 10, if self-matches were excluded. Especially for TM-align, all targets for LSD1 were ranked prior to all chains of that dataset which demonstrates the high overall structural similarity as well as the high similarity with respect to the SSE sequence and direction.

The matches to targets of APT1 were still among the top 1.5 % of all ranks. This structural similarity is also reflected by the CATH identifiers (see Table 5.11). The domains of the proteins with the LSD1 *ligand-sensing core* share the same classification and architecture levels. This means, all of these domains mainly consist of α-helices and share a very similar overall shape. In the current version of the CATH database (February of 2019), this group of domains consists of 60,694 domains in 1,226 superfamilies and 291 folds.

PDB-ID	CATH-ID
2ejr	1.10.10.10
1gos	1.10.405.10
2bxr	1.10.405.10

(a) LSD1

PDB-ID	CATH-ID
1fj2	3.40.50.1820
1k8q	3.40.50.1820

(b) APT1

Table 5.11: PDB and CATH identifiers for the two sets of proteins sharing common *ligand-sensing core*.

However, the situation was completely different for the corresponding domains of the APT1 example. Here, the CATH identifiers are identical. Thus, both domains consist of α-helices and strands, have a highly similar overall shape, share the same topology or fold in their cores, and are homologues. Homologues means that they share a common ancestor which correlates to a high sequence identity (e.g., $\geq$ 35 %) and/or high SSAP score from structure comparison [192]. The group of domains with this CATH identifier contained 3,274 domains in the latest version, but it contained only 1,062 in the most recent version available at the time of publication (version 3.3), i.e., the beginning of 2010.

The by far most striking fact is that these results were based on the entire chains and did not focus on the binding site, a domain, or the core manually defined by the authors. Thus, these proteins share a high similarity on the tertiary fold level that separated them from a set of representative decoy chains. Having said that, be reminded that the manually defined core of 1fj2@pdb includes SSEs that are within 15 Å of the core's center which are then extended to at most 25 Å from this center. Figure 5.1a reveals that almost all SSEs of the chain are entirely included in the core. The authors state that their core contained 5 α-helices (all of the chain) and 4 β-strands. Although they do not explicitly state that other helix classes were excluded, these helices may had been removed due to minimum length requirements. Another possible explanation is that the term α-helices is sometimes used for the group of helices regardless of any classification. In this case, this number of reported helices by the authors differed from the number of helices we observed which is 7 (see Figure 5.1a). In addition, the number of strands also differed between the authors' (4) and our observations (8). Both discrepancies might be the result of different atoms used for the core's center definition or the reduction of SSEs to Cα atoms during the distance calculation. Both details

are not specified in the publication.

The identical Z-score of 13.7 for the overall fold comparison of APT1 with DaliLite obtained by us as well as by the authors suggests that they also compared chains instead of entire proteins. For LSD1, the Z-score reported by Willmann et al. is even lower, although the corresponding chains are larger. Unfortunately, the authors do not provide the details of how the core for LSD1 was defined. Assuming that they used the same criteria as for APT1, the number of SSEs and their sizes are at least comparable. Given the fact that all SSCMs reported the highest structural similarities, the low reported score is questionable in this sense. However, the authors report comparatively low lengths of alignments even for identity matches of the core to its chain. The reported alignment length of the core to its chain for 2ejrA@pdb was 158 residues to 643 residues of the entire chain. Although the number of residues for the alignment is given, the selected residues remain unknown.

For the cores or automatically detected pockets, SLOT reported similar rankings for the targets of LSD1 as well as of APT1. Although the performance of SLOT was improved for APT1, this improvement was not based on a higher similarity. The matching of the entire chains contained 11 SSEs, which reflects good performance in general. The different relative rankings between the pockets and the chains were the result of the selected score of SLOT. $\mathrm{score}_{\mathrm{min}}$ favors structures of the same size. Thus, the differences in sizes are usually higher for chains than for pockets. To overcome this issue $\mathrm{score}_{\mathrm{min}}$ could be used for the determination of similarity on the overall-fold-level and $\mathrm{score}_{\mathrm{max}}$ for the similarity on the core-level.

Nevertheless, the best match of the primary pocket of ATP1 to a target pocket was assigned a score of 0.3571, which led to a rank of $3{,}252$. Only 5 SSEs, i.e., 2 helices and 3 strands, are matched between these pockets. Furthermore, the spatial arrangement of these SSEs can be frequently observed in the results. In conclusion, this match is out of the range of matches one would pick for further examination, such as visual inspection. This is especially and generally true for matches with a score of less than 0.5.

A topic that has not been discussed with respect to *ligand-sensing cores*, but which is important in this context is the occurrence of super-secondary structures. Super-secondary structures are common structural motifs consisting of helices and strands arranged in patters that can be found in many different protein structures. The Rossmann fold [193] is one of the most common super-secondary structure motifs in proteins [194, 195] with a high structural variety [196]. Moreover, the Rossmann fold is present in approximately $22\,\%$ of the available protein structures [197]. All of these proteins share a common *ligand-sensing core* per definition.

Another blind spot of this concept is the fact that similar binding molecules in case of a common *ligand-sensing core* are limited to a common scaffold [15]. There exists a myriad of possible side-chains combinations for the functionalization of the scaffold. This fact in combination with the frequency of super-secondary structures demonstrate the inspecificity of this concept.

From a computational point of view a lot of questions arose that still remain: What is a *good ligand-sensing core*? What is a good spatial arrangement of SSEs? What is the tradeoff between the size and the quality of the spacial arrangement? How many SSEs define a *ligand-sensing*

core? How high is the probability that such a spatial arrangement *can* recognize similar scaffolds?

All in all, we observed high similarities for the queries and targets of APT1 and LSD1 on the chain and the core level, which were reported by several SSCMs. We found additional structural similarities with SLOT due to its independence from the SSE sequence und the SSEs' directions. Since the publication of the PSSC approach in 2004, two *ligand-sensing cores* have been reported in the following 15 years. Both publications lack important details on the definition of their cores. In addition, there is a high number of open questions that hamper the transition from visual inspection preferences to formal criteria. Plus, there is a high degree of inspecificity with respect to the scaffold and no demarcation from super-secondary structures. In consequence, there is no evidence that demonstrates any benefits of the concept of *ligand-sensing cores* for the rational drug design.

One of the last topics to be discussed are the Turn Histograms introduced in Section 5.3.6.6. They provided the fastest runtimes in general, but were not able to separate actives from decoys in our CATH topology domain pair analysis. In spite of that, their performance was yet sufficient to be used in a pre-filtering step on a protein scale. In more detail, although a combination of the Turn Histograms and the SegmentedV1DM algorithm was not implemented, it could have been used to fulfill the two-step procedure for the search for *ligand-sensing cores*, i.e., dissimilarity on the protein and similarity on the binding site level.

Finally, there are several open challenges and possible optimizations for SLOT. First, all scores reflect the number of matched SSEs without considering the matching quality. This hampers a granular differentiation between different matchings. Second, SLOT utilizing the SegmentedV1DM modeling algorithm, is only able to match one SSE to another and does not support the matching of multiple SSEs to different parts of a single one. We introduced the SegmentedVSD1 modeling algorithm to address both challenges, but the observed runtimes are insufficient in any way. We additionally addressed the second challenge by using the option to split SSEs at kink positions provided by SCOT. Third, the parsing procedure and the internal protein data structures could be limited to the use of backbone atoms only. This would reduce the high memory footprint which currently detains the processing of the entire PDB at once. Fourth, a functionality could be added to cut SSEs to the size of the binding site. This however, is very sensitive with respect to the accuracy and consistency of the detected binding sites among the proteins of interest. Last but most important, an appropriate dataset has to be created to optimize and evaluate SLOT. The undiscovered similarities revealed by SLOT show that the current datasets and their definitions of decoys and actives are inadequate for our application domain.

In summary, SLOT is a novel tool for the search for similar spatial arrangements of SSEs. It was shown to find a new layer of similarity by its independence from the SSE sequence and the SSEs' directions. The multiple scores, parameters, and exports of models and matches allow a high degree of application-tailored adjustments and the analyses of their effects. We evaluated its performance against three other SSCMs, namely, LOCK2, DaliLite, and TM-align, with the help of domain pairs in general and the concept of *ligand-sensing cores* in particular. This concept was analyzed and discussed in detail with the consequence that the concept itself and its benefit for the rational drug design proposed by the authors remains questionable and unproven.

"It is a blessed thing that in every age some one has had the individuality enough and courage enough to stand by his own convictions."

Ferdinand Magellan

6 Conclusion

SSEs play important roles in the world of proteins but their importance is often disregarded. From visualization to structure comparison, alignment and SSE prediction, they contribute to many fundamental applications in the field of structure-based analyses for chemical biology. Yet, their classification remains subjective. Although there is a moderate consent among the SSAMs in the classification of common SSE classes, i.e., right-handed α-, and 3_{10}-helices as well as β-sheets, this does not hold true for comparatively rare and underestimated classes. It particularly manifests in the assignment of PPII helices for which no consent exists.

But as long as SSE assignments are considered as given by default by those who utilize them or assigned with little diligence by those who provide them, their influence remains concealed. Furthermore, the adherence to a more than 35 years old standard narrows the biasses our view on the most common SSE classes and hampers a discussion about the rare ones.

Apart from the assignment of SSEs, their influence on a protein's function in general is still an appealing but yet insufficiently answered question. The high number of SSCMs hints toward the idea that there can be a correlation or at least some sort of information still hidden. Although the concept of *ligand-sensing cores* failed to prove its applicability in general, a generally applicable connection between the arrangement of SSEs and a protein's function is not refuted heretofore. Plus, we showed that there are still many similarities to be discovered that have not yet been unveiled by other SSCMs so far.

We hope to nourish the discussion on SSEs with our contributions to their classification (SCOT),

their evaluation (SNOT), and the comparison of spatial SSE arrangements (SLOT).

At the end we can give the answer to the initial question of this thesis.

Have we found *ligand-sensing cores*?

No,

but we have found a new structural level of similarity, somewhere between the secondary and the tertiary level, that awaits its exploration.

Bibliography

[1] C. Ehrt, T. Brinkjost, and O. Koch. “Impact of Binding Site Comparisons on Medicinal Chemistry and Rational Molecular Design”. In: *Journal of Medicinal Chemistry* 59.9 (2016), pp. 4121–4151. DOI: 10.1021/acs.jmedchem.6b00078.

[2] H. Patel, T. Brinkjost, and O. Koch. “PyGOLD: a python based API for docking based virtual screening workflow generation”. In: *Bioinformatics* 33.16 (2017), pp. 2589–2590. DOI: 10.1093/bioinformatics/btx197.

[3] J. Jasper et al. “A novel interaction fingerprint derived from per atom score contributions: exhaustive evaluation of interaction fingerprint performance in docking based virtual screening”. In: *Journal of Cheminformatics* 10.1 (2018), p. 15. DOI: 10.1186/s13321-018-0264-0.

[4] C. Ehrt, T. Brinkjost, and O. Koch. “A benchmark driven guide to binding site comparison: An exhaustive evaluation using tailor-made data sets (ProSPECCTs)”. In: *PLOS Computational Biology* 14.11 (2018), pp. 1–50. DOI: 10.1371/journal.pcbi.1006483.

[5] C. Ehrt, T. Brinkjost, and O. Koch. “Binding Site Comparison – Software and Applications”. In: *Encyclopedia of Bioinformatics and Computational Biology*. Ed. by S. Ranganathan et al. Oxford: Academic Press, 2019, pp. 650–660. ISBN: 978-0-12-811432-2. DOI: 10.1016/B978-0-12-809633-8.20196-9.

[6] C. Ehrt, T. Brinkjost, and O. Koch. “Binding Site Characterization – Similarity, Promiscuity, and Druggability”. In: *Med. Chem. Commun.* (2019). In submission.

[7] T. Brinkjost et al. “SCOT: Rethinking the Classification of Secondary Structure Elements”. In: *Bioinformatics* (2019). DOI: 10.1093/bioinformatics/btz826.

[8] T. Blundell, H. Jhoti, and C. Abell. “High-throughput crystallography for lead discovery in drug design.” In: *Nature Reviews. Drug Discovery* 1 (2002), pp. 45–54. DOI: 10.1038/nrd706.

[9] H. M. Berman et al. “The protein data bank”. In: *Nucleic Acids Research* 28.1 (2000), pp. 235–242. DOI: 10.1093/nar/28.1.235.

[10] B. Adhikari and J. Cheng. “Improved protein structure reconstruction using secondary structures, contacts at higher distance thresholds, and non-contacts”. In: *BMC Bioinformatics* 18.1 (2017), p. 380. DOI: 10.1186/s12859-017-1807-5.

[11] Y. Zhang and J. Skolnick. “TM-align: a protein structure alignment algorithm based on the TM-score”. In: *Nucleic Acids Research* 33.7 (2005), pp. 2302–2309. DOI: 10.1093/nar/gki524.

[12] Q. Jiang et al. “Protein secondary structure prediction: a survey of the state of the art”. In: *Journal of Molecular Graphics and Modelling* 76 (2017), pp. 379–402. DOI: 10.1016/j.jmgm.2017.07.015.

[13] A. C. Pfluck et al. “Stability of lipases in miniemulsion systems: correlation between secondary structure and activity”. In: *Enzyme and Microbial Technology* 114 (2018), pp. 7–14. DOI: 10.1016/j.enzmictec.2018.03.003.

[14] Z. Khattari. “A correlation between secondary structure and rheological properties of low-density lipoproteins at air/water interfaces”. In: *Journal of Biological Physics* 43.3 (2017), pp. 381–395. DOI: 10.1007/s10867-017-9458-3.

[15] M. A. Koch and H. Waldmann. “Protein structure similarity clustering and natural product structure as guiding principles in drug discovery”. In: *Drug Discovery Today* 10.7 (2005), pp. 471–483. DOI: 10.1016/S1359-6446(05)03419-7.

[16] M. Tyagi et al. “Analysis of loop boundaries using different local structure assignment methods”. In: *Protein Science* 18.9 (2009), pp. 1869–1881. DOI: 10.1002/pro.198.

[17] E. F. Pettersen et al. “UCSF Chimera - a visualization system for exploratory research and analysis”. In: *Journal of Computational Chemistry* 25.13 (2004), pp. 1605–1612. DOI: 10.1002/jcc.20084.

[18] O. Koch and J. Cole. “An automated method for consistent helix assignment using turn information”. In: *Proteins: Structure, Function, and Bioinformatics* 79.5 (2011), pp. 1416–1426. DOI: 10.1002/prot.22968.

[19] H. Berman, K. Henrick, and H. Nakamura. “Announcing the worldwide protein data bank”. In: *Nature Structural and Molecular Biology* 10 (2003), p. 980. DOI: 10.1038/nsb1203-980.

[20] I. K. McDonald and J. M. Thornton. “Satisfying hydrogen bonding potential in proteins”. In: *Journal of Molecular Biology* 238.5 (1994), pp. 777–793. DOI: 10.1006/jmbi.1994.1334.

[21] S. L. Mayo, B. D. Olafson, and W. A. Goddard. “DREIDING: a generic force field for molecular simulations”. In: *The Journal of Physical Chemistry* 94.26 (1990), pp. 8897–8909. DOI: 10.1021/j100389a010.

[22] W. Kabsch and C. Sander. “Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features”. In: *Biopolymers* 22.12 (1983), pp. 2577–2637. DOI: 10.1002/bip.360221211.

[23] G. Wang and R. L. Dunbrack Jr. “PISCES: a protein sequence culling server”. In: *Bioinformatics* 19.12 (2003), pp. 1589–1591. DOI: 10.1093/bioinformatics/btg224.

[24] A. Ultsch and F. Mörchen. *ESOM-Maps: tools for clustering, visualization, and classification with Emergent SOM*. Tech. rep. 46. University of Marburg, 2005.

[25] P. Y. Chou and G. D. Fasman. “Conformational parameters for amino acids in helical, beta-sheet, and random coil regions calculated from proteins”. In: *Biochemistry* 13.2 (1974), pp. 211–222. DOI: 10.1021/bi00699a001.

[26] C. Wilmot and J. Thornton. “Analysis and prediction of the different types of beta-turn in proteins”. In: *Journal of Molecular Biology* 203.1 (1988), pp. 221–232. DOI: 10.1016/0022-2836(88)90103-9.

[27] E. Krissinel and K. Henrick. “Secondary-structure matching (SSM), a new tool for fast protein structure alignment in three dimensions”. In: *Acta Crystallographica Section D* 60.12 Part 1 (2004), pp. 2256–2268. DOI: 10.1107/S0907444904026460.

[28] E. Tomita, A. Tanaka, and H. Takahashi. “The worst-case time complexity for generating all maximal cliques and computational experiments”. In: *Theoretical Computer Science* 363.1 (2006). Computing and Combinatorics, pp. 28–42. DOI: 10.1016/j.tcs.2006.06.015.

[29] C. Bron and J. Kerbosch. “Algorithm 457: finding all cliques of an undirected graph”. In: *Communications of the ACM* 16.9 (1973), pp. 575–577. DOI: 10.1145/362342.362367.

[30] G. Levi. “A note on the derivation of maximal common subgraphs of two directed or undirected graphs”. In: *CALCOLO* 9.4 (1973), p. 341. DOI: 10.1007/BF02575586.

[31] M. R. Garey and D. S. Johnson. *Computers and intractability; a guide to the theory of NP-completeness*. New York, NY, USA: W. H. Freeman & Co., 1990. ISBN: 0716710455.

[32] Schrödinger, LLC. “The PyMOL molecular graphics system, version 1.8”. 2015.

[33] D. Willmann et al. “Impairment of prostate cancer cell growth by a selective and reversible lysine-specific demethylase 1 inhibitor”. In: *International Journal of Cancer* 131.11 (2012), pp. 2704–2709. DOI: 10.1002/ijc.27555.

[34] F. J. Dekker et al. “Small-molecule inhibition of APT1 affects Ras localization and signaling”. In: *Nature Chemical Biology* 6.449-456 (2010). DOI: 10.1038/nchembio.362.

[35] L. Pauling, R. B. Corey, and H. Branson. “The structure of proteins: two hydrogen-bonded helical configurations of the polypeptide chain”. In: *Proceedings of the National Academy of Sciences of the USA* 37.4 (1951), pp. 205–211. DOI: 10.1073/pnas.37.4.205.

[36] L. Pauling and R. B. Corey. “The Pleated Sheet, A New Layer Configuration of Polypeptide Chains”. In: *Proceedings of the National Academy of Sciences* 37.5 (1951), pp. 251–256. DOI: 10.1073/pnas.37.5.251.

[37] J. W. Raymond and P. Willett. “Maximum common subgraph isomorphism algorithms for the matching of chemical structures”. In: *Journal of Computer-Aided Molecular Design* 16.7 (2002), pp. 521–533. DOI: 10.1023/A:1021271615909.

[38] N. L. Dawson et al. “CATH: an expanded resource to predict protein function through structure and sequence”. In: *Nucleic Acids Research* 45.D1 (2017), pp. D289–D295. DOI: 10.1093/nar/gkw1098.

[39] S. F. Altschul et al. “Basic local alignment search tool”. In: *Journal of Molecular Biology* 215.3 (1990), pp. 403–410. DOI: https://doi.org/10.1016/S0022-2836(05)80360-2.

[40] A. S. Konagurthu, A. M. Lesk, and L. Allison. “Minimum message length inference of secondary structure from protein coordinate data”. In: *Bioinformatics* 28.12 (2012), pp. i97–i105. DOI: 10.1093/bioinformatics/bts223.

[41] C. C. G. ULC. “Molecular Operating Environment (MOE)”. 1010 Sherbooke St. West, Suite #910, Montreal, QC, Canada, H3A 2R7. 2015.

[42] Y. Zhang and J. Skolnick. “Scoring function for automated assessment of protein structure template quality”. In: *Proteins: Structure, Function, and Bioinformatics* 57.4 (2004), pp. 702–710. DOI: 10.1002/prot.20264.

[43] R. C. Edgar. “Search and clustering orders of magnitude faster than BLAST”. In: *Bioinformatics* 26.19 (2010), pp. 2460–2461. DOI: 10.1093/bioinformatics/btq461.

[44] H. Cheng et al. “ECOD: an evolutionary classification of protein domains”. In: *PLOS Computational Biology* 10.12 (2014), pp. 1–18. DOI: 10.1371/journal.pcbi.1003926.

[45] R. Krivák and D. Hoksza. “P2RANK: knowledge-based ligand binding site prediction using aggregated local features”. In: *Algorithms for Computational Biology*. Ed. by A.-H. Dediu et al. Springer International Publishing, 2015, pp. 41–52. DOI: 10.1007/978-3-319-21233-3_4.

[46] R. Krivák and D. Hoksza. “P2Rank: machine learning based tool for rapid and accurate prediction of ligand binding sites from protein structure”. In: *Journal of Cheminformatics* 10.1 (2018), p. 39. DOI: 10.1186/s13321-018-0285-8.

[47] M. Levitt and J. Greer. “Automatic identification of secondary structure in globular proteins”. In: *Journal of Molecular Biology* 114.2 (1977), pp. 181–239. DOI: 10.1016/0022-2836(77)90207-8.

[48] J. Zacharias and E.-W. Knapp. “Protein secondary structure classification revisited: processing DSSP information with PSSC”. In: *Journal of Chemical Information and Modeling* 54.7 (2014), pp. 2166–2179. DOI: 10.1021/ci5000856.

[49] Y. Mansiaux et al. “Assignment of polyproline II conformation and analysis of sequence – structure relationship”. In: *PLOS ONE* 6.3 (Mar. 2011), pp. 1–15. DOI: 10.1371/journal.pone.0018401.

[50] R. Chebrek et al. “PolyprOnline: polyproline helix II and secondary structure assignment database”. In: *Database : the journal of biological databases and curation* 2014 (2014), bau102. DOI: 10.1093/database/bau102.

[51] D. Frishman and P. Argos. “Knowledge-based protein secondary structure assignment”. In: *Proteins: Structure, Function, and Bioinformatics* 23.4 (1995), pp. 566–579. DOI: 10.1002/prot.340230412.

[52] J. Martin et al. “Protein secondary structure assignment revisited: a detailed analysis of different assignment methods”. In: *BMC Structural Biology* 5.17 (2005). DOI: 10.1186/1472-6807-5-17.

[53] M. V. Cubellis, F. Cailliez, and S. C. Lovell. “Secondary structure assignment that accurately reflects physical and evolutionary characteristics”. In: *BMC Bioinformatics* 6.Suppl 4 (2005). DOI: 10.1186/1471-2105-6-S4-S8.

[54] G. Nagy and C. Oostenbrink. “Dihedral-based segment identification and classification of biopolymers I: proteins”. In: *Journal of Chemical Information and Modeling* 54.1 (2014), pp. 266–277. DOI: 10.1021/ci400541d.

[55] R. Srinivasan and G. D. Rose. “A physical basis for protein secondary structure”. In: *Proc Natl Acad Sci USA* 96.25 (1999), pp. 14258–14263.

[56] K.-H. Chen et al. “A multidimensional divide-and-conquer algorithm for assigning secondary structures in proteins”. In: *The 26th Workshop on Combinatorial Mathematics and Computation Theory* (2009).

[57] M. Parisien and F. Major. “A new catalog of protein β-sheets”. In: *Proteins: Structure, Function, and Bioinformatics* 61.3 (2005), pp. 545–558. DOI: 10.1002/prot.20677.

[58] P. Carter, C. A. F. Andersen, and B. Rost. “DSSPcont: continuous secondary structure assignments for proteins”. In: *Nucleic Acids Research* 31.13 (2003), pp. 3293–3295. DOI: 10.1093/nar/gkg626.

[59] M. N. Fodje and S. Al-Karadaghi. “Occurrence, conformational features and amino acid propensities for the pi-helix”. In: *Protein Engineering* 15.5 (2002), pp. 353–358. DOI: 10.1093/protein/15.5.353.

[60] E. G. Hutchinson and J. M. Thornton. “PROMOTIF–A program to identify and analyze structural motifs in proteins”. In: *Protein Science* 5.2 (1996), pp. 212–220. DOI: 10.1002/pro.5560050204.

[61] K. Mizuguchi et al. “JOY: Protein sequence-structure representation and analysis”. In: *Bioinformatics* 14.7 (1998), pp. 617–623. DOI: 10.1093/bioinformatics/14.7.617.

[62] J. Overington et al. “Tertiary structural constraints on protein evolutionary diversity: templates, key residues and structure prediction”. In: *Proceedings of the Royal Society of London B: Biological Sciences* 241.1301 (1990), pp. 132–145. DOI: 10.1098/rspb.1990.0077.

[63] E. O. Salawu. “RaFoSA: Random forests secondary structure assignment for coarse-grained and all-atom protein systems”. In: *Cogent Biology* 2.1 (2016), p. 1214061. DOI: 10.1080/23312025.2016.1214061.

[64] C. Cao et al. “A new secondary structure assignment algorithm using Cα backbone fragments”. In: *International Journal of Molecular Sciences* 17.3 (2016), p. 333. DOI: 10.3390/ijms17030333.

[65] P. Kumar and M. Bansal. “Identification of local variations within secondary structures of proteins”. In: *Acta Crystallographica Section D* 71.5 (2015), pp. 1077–1086. DOI: 10.1107/S1399004715003144.

[66] G. Kneller and K. Hinsen. “Protein secondary structure description with a coarse-grained model”. In: *Acta Crystallographica* 71 (July 2015), pp. 1411–1422. DOI: 10.1107/S1399004715007191.

[67] S. M. Law, A. T. Frank, and C. L. Brooks III. “PCASSO: A fast and efficient Cα-based method for accurately assigning protein secondary structure elements”. In: *Journal of Computational Chemistry* 35.24 (2014), pp. 1757–1761. DOI: 10.1002/jcc.23683.

[68] S. Y. Park et al. “SABA (secondary structure assignment program based on only alpha carbons): a novel pseudo center geometrical criterion for accurate assignment of protein secondary structures”. In: *BMB Reports* 44.2 (2011), pp. 118–122. DOI: 10.5483/BMBRep.2011.44.2.118.

[69] A. S. Konagurthu et al. “Piecewise linear approximation of protein structures using the principle of minimum message length”. In: *Bioinformatics* 27.13 (2011), pp. i43–i51. DOI: 10.1093/bioinformatics/btr240.

[70] S.-R. Hosseini et al. “PROSIGN: A method for protein secondary structure assignment based on three-dimensional coordinates of consecutive Cα atoms”. In: *Computational Biology and Chemistry* 32.6 (2008), pp. 406–411. DOI: 10.1016/j.compbiolchem.2008.07.027.

[71] I. Majumdar, S. Krishna, and N. V. Grishin. “PALSSE: A program to delineate linear secondary structural elements from protein structures”. In: *BMC Bioinformatics* 6.202 (2005), pp. 1471–2105. DOI: 10.5483/BMBRep.2011.44.2.118.

[72] T. Taylor et al. “New method for protein secondary structure assignment based on a simple topological descriptor”. In: *Proteins: Structure, Function, and Bioinformatics* 60.3 (2005), pp. 513–524. DOI: 10.1002/prot.20471.

[73] F. Dupuis, J.-F. Sadoc, and J.-P. Mornon. “Protein secondary structure assignment through Voronoi tessellation”. In: *Proteins: Structure, Function, and Bioinformatics* 55.3 (2004), pp. 519–528. DOI: 10.1002/prot.10566.

[74] W. R. Taylor. “Defining linear segments in protein structure”. In: *Journal of Molecular Biology* 310.5 (2001), pp. 1135–1150. ISSN: 0022-2836. DOI: 10.1006/jmbi.2001.4817.

[75] S. M. King and J. W. Curtis. “Assigning secondary structure from protein coordinate data”. In: *Proteins: Structure, Function, and Bioinformatics* 35.3 (1999), pp. 313–320. DOI: 10.1002/(SICI)1097-0134(19990515)35:3<313::AID-PROT5>3.0.CO;2-1.

[76] G. Labesse et al. “P-SEA: a new efficient assignment of secondary structure from C alpha trace of proteins”. In: *Computer Applications in the iosciences : CABIOS* 13.5 (1997), pp. 291–295. DOI: 10.1093/bioinformatics/13.3.291.

[77] G. J. Kleywegt and T. A. Jones. “Detecting Folding Motifs and Similarities in Protein Structures”. In: *Methods in Enzymology* 277.27 (1997), pp. 525–545. DOI: 10.1016/S0076-6879(97)77029-0.

[78] H. Sklenar, C. Etchebest, and R. Lavery. “Describing protein structure: a general algorithm yielding complete helicoidal parameters and a unique overall axis”. In: *Proteins: Structure, Function, and Bioinformatics* 6.1 (1989), pp. 46–60. DOI: 10.1002/prot.340060105.

[79] F. M. Richards and C. E. Kundrot. “Identification of structural motifs from protein coordinate data: Secondary structure and first-level supersecondary structure”. In: *Proteins: Structure, Function, and Bioinformatics* 3.2 (1988), pp. 71–84. DOI: 10.1002/prot.340030202.

[80] W. Zhang, A. K. Dunker, and Y. Zhou. “Assessing secondary structure assignment of protein structures by using pairwise sequence-alignment benchmarks”. In: *Proteins: Structure, Function, and Bioinformatics* 71.1 (2007), pp. 61–67. DOI: 10.1002/prot.21654.

[81] N. Colloc’h et al. “Comparison of three algorithms for the assignment of secondary structure in proteins: the advantages of a consensus assignment”. In: *Protein Engineering, Design and Selection* 6.4 (1993), pp. 377–382. DOI: 10.1093/protein/6.4.377.

[82] R. van der Kant and G. Vriend. “α-bulges in G protein-coupled receptors”. In: *International Journal of Molecular Sciences* 15.5 (2014), pp. 7841–7864. DOI: 10.3390/ijms15057841.

[83] W. Humphrey, A. Dalke, and K. Schulten. “VMD – visual molecular dynamics”. In: *Journal of Molecular Graphics* 14 (1996), pp. 33–38. DOI: 10.1016/0263-7855(96)00018-5.

[84] J. Maupetit, R. Gautier, and P. Tufféry. “SABBAC: online structural alphabet-based protein backbone reconstruction from α-carbon trace”. In: *Nucleic Acids Research* 34.suppl_2 (2006), W147–W151. DOI: 10.1093/nar/gkl289.

[85] P. Kumar and M. Bansal. “HELANAL-Plus: a web server for analysis of helix geometry in protein structures”. In: *Journal of Biomolecular Structure and Dynamics* 30.6 (2012), pp. 773–783. DOI: 10.1080/07391102.2012.689705.

[86] S. C. Lovell et al. “Structure validation by Cα geometry: ϕ, ψ and Cβ deviation”. In: *Proteins: Structure, Function, and Bioinformatics* 50.3 (2003), pp. 437–450. DOI: 10.1002/prot.10286.

[87] J. M. Word et al. “Asparagine and glutamine: using hydrogen atom contacts in the choice of side-chain amide orientation”. In: *Journal of Molecular Biology* 285.4 (1999), pp. 1735–1747. DOI: 10.1006/jmbi.1998.2401.

[88] O. Koch and G. Klebe. “Turns revisited: A uniform and comprehensive classification of normal, open, and reverse turn families minimizing unassigned random chain portions”. In: *Proteins: Structure, Function, and Bioinformatics* 74.2 (2009), pp. 353–367. DOI: 10.1002/prot.22185.

[89] B. I. Dahiyat, D. Benjamin Gordon, and S. L. Mayo. “Automated design of the surface positions of protein helices”. In: *Protein Science* 6.6 (1997), pp. 1333–1337. DOI: 10.1002/pro.5560060622.

[90] C. Ehrt. “PhD thesis”. Unpublished.

[91] T. Holder. *https://pymolwiki.org/index.php/AngleBetweenHelices*. 2018. (Visited on 03/25/2018).

[92] M. Hendlich et al. “Relibase: design and development of a database for comprehensive analysis of protein–ligand interactions”. In: *Journal of Molecular Biology* 326.2 (2003), pp. 607–620. DOI: 10.1016/S0022-2836(02)01408-0.

[93] R. Aurora, R. Srinivasan, and G. Rose. “Rules for alpha-helix termination by glycine”. In: *Science* 264.5162 (1994), pp. 1126–1130. DOI: 10.1126/science.8178170.

[94] T. O. Street et al. “Physical-chemical determinants of turn conformations in globular proteins”. In: *Protein Science* 16.8 (2007), pp. 1720–1727. DOI: 10.1110/ps.072898507.

[95] G. Hughes. “Multivariate and time series models for circular data with applications to protein conformational angles”. PhD thesis. The University of Leeds, Department of Statistics, 2007.

[96] E. Batschelet. *Circular statistics in biology*. Mathematics in Biology. Academic Press, 1981. ISBN: 9780120810505.

[97] H. Sugeta and T. Miyazawa. “General method for calculating helical parameters of polymer chains from bond lengths, bond angles, and internal-rotation angles”. In: *Biopolymers* 5.7 (1967), pp. 673–679. DOI: 10.1002/bip.1967.360050708.

[98] M. Bansal, S. Kumart, and R. Velavan. “HELANAL: a Program to Characterize Helix Geometry in Proteins”. In: *Journal of Biomolecular Structure and Dynamics* 17.5 (2000), pp. 811–819. DOI: 10.1080/07391102.2000.10506570.

[99] O. Carugo and P. Argos. “Accessibility to internal cavities and ligand binding sites monitored by protein crystallographic thermal factors”. In: *Proteins: Structure, Function, and Bioinformatics* 31.2 (1998), pp. 201–213. DOI: 10.1002/(SICI)1097-0134(19980501)31:2<201::AID-PROT9>3.0.CO;2-O.

[100] E. C. Meng et al. “Tools for integrated sequence-structure analysis with UCSF Chimera”. In: *BMC Bioinformatics* 7 (2006), p. 339. DOI: 10.1186/1471-2105-7-339.

[101] J. Shapiro and D. Brutlag. “FoldMiner and LOCK 2: protein structure comparison and motif discovery on the web”. In: *Nucleic Acids Research* 32 (2004), W536–W541. DOI: 10.1093/nar/gkh389.

[102] Web of Science Core Collections. *https://apps.webofknowledge.com*. 2018. (Visited on 2018).

[103] H. R. Wilman, J. Shi, and C. M. Deane. “Helix kinks are equally prevalent in soluble and membrane proteins”. In: *Proteins: Structure, Function, and Bioinformatics* 82.9 (2014), pp. 1960–1970. DOI: 10.1002/prot.24550.

[104] D. N. Langelaan et al. “Improved helix and kink characterization in membrane proteins allows evaluation of kink sequence predictors”. In: *Journal of Chemical Information and Modeling* 50.12 (2010), pp. 2213–2220. DOI: 10.1021/ci100324n.

[105] P. Enkhbayar et al. “310-helices in proteins are parahelices”. In: *Proteins: Structure, Function, and Bioinformatics* 64.3 (2006), pp. 691–699. DOI: 10.1002/prot.21026.

[106] A. D. Meruelo, I. Samish, and J. U. Bowie. “TMKink: a method to predict transmembrane helix kinks”. In: *Protein Science* 20.7 (2011), pp. 1256–1264. DOI: 10.1002/pro.653.

[107] Z. Guo, E. Kraka, and D. Cremer. “Description of local and global shape properties of protein helices”. In: *Journal of Molecular Modeling* 19.7 (2013), pp. 2901–2911. DOI: 10.1007/s00894-013-1819-7.

[108] R. B. Cooley, D. J. Arp, and P. A. Karplus. “Evolutionary origin of a secondary structure: π-helices as cryptic but widespread insertional variations of α-helices that enhance protein f unctionality”. In: *Journal of Molecular Biology* 404.2 (2010), pp. 232–246. DOI: 10.1016/j.jmb.2010.09.034.

[109] S. A. Hollingsworth, D. S. Berkholz, and P. A. Karplus. “On the occurrence of linear groups in proteins”. In: *Protein Science* 18.6 (2009), pp. 1321–1325. DOI: 10.1002/pro.133.

[110] P. Kumar and M. Bansal. “Dissecting π-helices: sequence, structure and function”. In: *The FEBS Journal* 282.22 (2015), pp. 4415–4432. DOI: 10.1111/febs.13507.

[111] M. Novotny and G. J. Kleywegt. “A survey of left-handed helices in protein structures”. In: *Journal of Molecular Biology* 347.2 (2005), pp. 231–241. DOI: 10.1016/j.jmb.2005.01.037.

[112] S. A. Hollingsworth and P. A. Karplus. “A fresh look at the Ramachandran plot and the occurrence of standard structures in proteins”. In: *Biomolecular Concepts* 1.3-4 (2010), pp. 271–283. DOI: 10.1515/BMC.2010.022.

[113] P. M. Cowan and S. McGavin. “Structure of poly-L-proline”. In: *Nature* 176 (1955), pp. 501–503. DOI: 10.1038/176501a0.

[114] A. A. Adzhubei, M. J. Sternberg, and A. A. Makarov. “Polyproline-II helix in proteins: structure and function”. In: *Journal of Molecular Biology* 425.12 (2013), pp. 2100–2132. DOI: 10.1016/j.jmb.2013.03.018.

[115] T. J. Narwani et al. “Recent advances on polyproline II”. In: *Amino Acids* 49.4 (2017), pp. 705–713. DOI: 10.1007/s00726-017-2385-6.

[116] N. Sreerama and R. W. Woody. “Molecular dynamics simulations of polypeptide conformations in water: a comparison of α, β, and poly(pro)II conformations”. In: *Proteins: Structure, Function, and Bioinformatics* 36.4 (1999), pp. 400–406. DOI: 10.1002/(SICI)1097-0134(19990901)36:4<400::AID-PROT3>3.0.CO;2-B.

[117] M. Martino et al. “On the occurrence of polyproline II structure in elastin”. In: *Journal of Molecular Structure* 519 (2000), pp. 173–189. DOI: 10.1016/S0022-2860(99)00299-9.

[118] I. J. Bruno et al. “IsoStar: A library of information about nonbonded interactions”. In: *Journal of Computer-Aided Molecular Design* 11.6 (1997), pp. 525–537. DOI: 10.1023/A:1007934413448.

[119] Z. Liu et al. “Geometrical preferences of the hydrogen bonds on protein–ligand binding interface derived from statistical surveys and quantum mechanics calculations”. In: *Journal of Chemical Theory and Computation* 4.11 (2008), pp. 1959–1973. DOI: 10.1021/ct800267x.

[120] G. Ramachandran and V. Sasisekharan. “Conformation of polypeptides and proteins”. In: ed. by C. Anfinsen et al. Vol. 23. Advances in Protein Chemistry. Academic Press, 1968, pp. 283–437. DOI: 10.1016/S0065-3233(08)60402-7.

[121] J. Donohue. “Hydrogen bonded helical configurations of the polypeptide chain”. In: *Proceedings of the National Academy of Sciences of the USA* 39.6 (1953), pp. 470–478. DOI: 10.1073/pnas.39.6.470.

[122] A. I. Jiménez, G. Ballano, and C. Cativiela. “First observation of two consecutive γ turns in a crystalline linear dipeptide”. In: *Angewandte Chemie International Edition* 44.3 (2005), pp. 396–399. DOI: 10.1002/anie.200461230.

[123] M. Tsunemi et al. “Crystal structure of an elastase-specific inhibitor elafin complexed with porcine pancreatic elastase determined at 1.9 Å resolution”. In: *Biochemistry* 35.36 (1996), pp. 11570–11576. DOI: 10.1021/bi9609001.

[124] M. Yang et al. “Structural basis of histone demethylation by LSD1 revealed by suicide inactivation”. In: *Nature Structural and Molecular Biology* 14.6 (2007), pp. 535–539. DOI: 10.1038/nsmb1255.

[125] G. Ramachandran and R. Chandrasekaran. “Conformation of peptide chains containing both L- and D-residues. I. Helical structures with alternating L- and D-residues with special reference to the LD-ribbon and the LD-helices”. In: *Indian Journal of Biochemistry and Biophysics* 9.1 (1972), pp. 1–11. DOI: 10.1016/S0065-3233(08)60402-7.

[126] E. Bradbury et al. “The structure of the omega-form of poly-beta-benzyl-L-aspartate”. In: *Journal of Molecular Biology* (1962), pp. 230–247. DOI: 10.1016/S0022-2836(62)80086-2.

[127] R. Fraser, T. Macrae, and I. Stapleton. “Omega-helix in synthetic polypeptides”. In: *Nature* (1962). DOI: 10.1038/193573a0.

[128] P. Enkhbayar, B. Boldgiv, and N. Matsushima. “ω-helices in proteins”. In: *The Protein Journal* 29.4 (2010), pp. 242–249. DOI: 10.1007/s10930-010-9245-5.

[129] J. Martin et al. “Protein secondary structure assignment revisited: a detailed analysis of different assignment methods”. In: *BMC Structural Biology* 5.1 (2005), p. 17. DOI: 10.1186/1472-6807-5-17.

[130] C. A. Andersen et al. “Continuum secondary structure captures protein flexibility”. In: *Structure* 10.2 (2002), pp. 175–184. DOI: 10.1016/S0969-2126(02)00700-1.

[131] B. Offmann, M. Tyagi, and A. G. de Brevern. “Local protein structures”. In: *Current Bioinformatics* 2.3 (2007), pp. 165–202. DOI: 10.2174/157489307781662105.

[132] Y. Zhang and C. Sagui. “Secondary structure assignment for conformationally irregular peptides: comparison between DSSP, STRIDE and KAKSI”. In: *Journal of Molecular Graphics and Modelling* 55 (2015), pp. 72–84. DOI: 10.1016/j.jmgm.2014.10.005.

[133] W. G. Hol. “The role of the α-helix dipole in protein function and structure”. In: *Progress in Biophysics and Molecular Biology* 45.3 (1985), pp. 149–195. DOI: 10.1016/0079-6107(85)90001-X.

[134] M. A. Koch et al. “Charting biologically relevant chemical space: a structural classification of natural products (SCONP)”. In: *Proceedings of the National Academy of Sciences of the USA* 102.48 (2005), pp. 17272–17277. DOI: 10.1073/pnas.0503647102.

[135] M. A. Koch et al. “Compound library development guided by protein structure similarity clustering and natural product structure”. In: *Proceedings of the National Academy of Sciences of the USA* 101.48 (2004), pp. 16721–16726. DOI: 10.1073/pnas.0404719101.

[136] F. J. Dekker, M. A. Koch, and H. Waldmann. “Protein structure similarity clustering (PSSC) and natural product structure as inspiration sources for drug development and chemical genomics”. In: *Current Opinion in Chemical Biology* 9.3 (2005), pp. 232–239. DOI: 10.1016/j.cbpa.2005.03.003.

[137] L. Holm and C. Sander. “Protein structure comparison by alignment of distance matrices”. In: *Journal of Molecular Biology* 233.1 (1993), pp. 123–138. DOI: 10.1006/jmbi.1993.1489.

[138] P. Willett. “Matching of chemical and biological structures using subgraph and maximal common subgraph isomorphism algorithms”. In: *Rational Drug Design*. Ed. by D. G. Truhlar et al. New York, NY: Springer New York, 1999, pp. 11–38. ISBN: 978-1-4612-1480-9. DOI: 10.1007/978-1-4612-1480-9_3.

[139] S. Minami, K. Sawada, and G. Chikenji. “MICAN : a protein structure alignment algorithm that can handle Multiple-chains, Inverse alignments, Calpha only models, Alternative alignments, and Non-sequential alignments”. In: *BMC Bioinformatics* 14.1 (2013), p. 24. DOI: 10.1186/1471-2105-14-24.

[140] K. P. Tan, M. N. Nguyen, and M. S. Madhusudhan. “CLICK—topology-independent comparison of biomolecular 3D structures”. In: *Nucleic Acids Research* 39.suppl_2 (2011), W24–W28. DOI: 10.1093/nar/gkr393.

[141] A. Guerler and E.-W. Knapp. “Novel protein folds and their nonsequential structural analogs”. In: *Protein Science* 17.8 (2008), pp. 1374–1382. DOI: 10.1110/ps.035469.108.

[142] S. Shi et al. “Searching for three-dimensional secondary structural patterns in proteins with ProSMoS”. In: *Bioinformatics* 23.11 (2007), pp. 1331–1338. DOI: 10.1093/bioinformatics/btm121.

[143] E. Krissinel and K. Henrick. “Multiple alignment of protein structures in three dimensions”. In: *Computational Life Sciences*. Ed. by M. R. Berthold et al. Springer Berlin Heidelberg, 2005, pp. 67–78. DOI: 10.1007/11560500_7.

[144] O. Dror et al. “MASS: multiple structural alignment by secondary structures”. In: *Bioinformatics* 19.suppl_1 (2003), pp. i95–i104. DOI: 10.1093/bioinformatics/btg1012.

[145] G. Lu. “TOP: a new method for protein structure comparisons and similarity searches”. In: *Journal of Applied Crystallography* 33.1 (2000), pp. 176–183. DOI: 10.1107/S0021889899012339.

[146] M. Petitjean. “Interactive maximal common 3D substructure searching with the combined SDM/RMS algorithm”. In: *Computers and Chemistry* 22.6 (1998), pp. 463–465. DOI: 10.1016/S0097-8485(98)00017-5.

[147] B. Kolbeck et al. “Connectivity independent protein-structure alignment: a hierarchical approach”. In: *BMC Bioinformatics* 7.1 (2006), p. 510. DOI: 10.1186/1471-2105-7-510.

[148] J. D. Szustakowski and Z. Weng. “Protein structure alignment using a genetic algorithm”. In: *Proteins: Structure, Function, and Bioinformatics* 38.4 (2000), pp. 428–440. DOI: 10.1002/(SICI)1097-0134(20000301)38:4<428::AID-PROT8>3.0.CO;2-N.

[149] J.-F. Gibrat, T. Madej, and S. H. Bryant. “Surprising similarities in structure comparison”. In: *Current Opinion in Structural Biology* 6.3 (1996), pp. 377–385. DOI: 10.1016/S0959-440X(96)80058-3.

[150] H. Sun et al. “Smolign: a spatial motifs-based protein multiple structural alignment method”. In: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 9.1 (2012), pp. 249–261. DOI: 10.1109/TCBB.2011.67.

[151] A. D. Stivala, P. J. Stuckey, and A. I. Wirth. “Fast and accurate protein substructure searching with simulated annealing and GPUs”. In: *BMC Bioinformatics* 11.1 (2010), p. 446. DOI: 10.1186/1471-2105-11-446.

[152] A. Stivala, A. Wirth, and P. J. Stuckey. “Tableau-based protein substructure search using quadratic programming”. In: *BMC Bioinformatics* 10.1 (2009), p. 153. DOI: 10.1186/1471-2105-10-153.

[153] A. S. Konagurthu, P. J. Stuckey, and A. M. Lesk. “Structural search and retrieval using a tableau representation of protein folding patterns”. In: *Bioinformatics* 24.5 (2008), pp. 645–651. DOI: 10.1093/bioinformatics/btm641.

[154] J. Vesterstrøm and W. R. Taylor. “Flexible secondary structure based protein structure comparison applied to the detection of circular permutation”. In: *Journal of Computational Biology* 13.1 (2006), pp. 43–63. DOI: 10.1089/cmb.2006.13.43.

[155] E. S C Shih and M.-J. Hwang. “Protein structure comparison by probability-based matching of secondary structure elements”. In: *Bioinformatics* 19 (2003), pp. 735–41. DOI: 10.1093/bioinformatics/btg058.

[156] A. Harrison et al. “Recognizing the fold of a protein structure”. In: *Bioinformatics* 19.14 (2003), pp. 1748–1759. DOI: 10.1093/bioinformatics/btg240.

[157] V. Alesker, R. Nussinov, and H. J. Wolfson. “Detection of non-topological motifs in protein structures”. In: *Protein Engineering, Design and Selection* 9.12 (1996), pp. 1103–1119. DOI: 10.1093/protein/9.12.1103.

[158] I. Koch, T. Lengauer, and E. Wanke. “An qlgorithm for finding maximal common subtopologies in a set of protein structures”. In: *Journal of Computational Biology* 3 (1996), pp. 289–306. DOI: 10.1089/cmb.1996.3.289.

[159] N. N. Alexandrov and D. Fischer. “Analysis of topological and nontopological structural similarities in the PDB: New examples with old structures”. In: *Proteins: Structure, Function, and Bioinformatics* 25.3 (1996), pp. 354–365. DOI: 10.1002/(SICI)1097-0134(199607)25:3<354::AID-PROT7>3.0.CO;2-F.

[160] K. Mizuguchi and N. Go. “Comparison of spatial arrangements of secondary structure elements in proteins”. In: *Protein engineering* 8 (1995), pp. 353–362. DOI: 10.1093/protein/8.4.353.

[161] H. M. Grindley et al. “Identification of tertiary structure resemblance in proteins using a maximal common subgraph isomorphism algorithm”. In: *Journal of Molecular Biology* 229.3 (1993), pp. 707–721. DOI: 10.1006/jmbi.1993.1074.

[162] Z. H. Zhang et al. “deconSTRUCT: general purpose protein database search on the substructure level”. In: *Nucleic Acids Research* 38.suppl_2 (2010), W590–W594. DOI: 10.1093/nar/gkq489.

[163] J. Ebert and D. Brutlag. “Development and validation of a consistency based multiple structure alignment algorithm”. In: *Bioinformatics* 22.9 (2006), pp. 1080–1087. DOI: 10.1093/bioinformatics/btl046.

[164] T. Kawabata. “MATRAS: A program for protein 3D structure comparison”. In: *Nucleic Acids Research* 31 (2003), pp. 3367–3369. DOI: 10.1093/nar/gkg581.

[165] A.-S. Yang and B. Honig. “An integrated approach to the analysis and modeling of protein sequences and structures. I. Protein structural alignment and a quantitative measure for protein structural distance”. In: *Journal of Molecular Biology* 301.3 (2000), pp. 665–678. DOI: 10.1006/jmbi.2000.3973.

[166] G. J. Kleywegt and T. A. Jones. “Detecting folding motifs and similarities in protein structures”. In: *Macromolecular Crystallography Part B*. Vol. 277. Methods in Enzymology. Academic Press, 1997, pp. 525–545. DOI: 10.1016/S0076-6879(97)77029-0.

[167] C. A. Orengo, N. P. Brown, and W. R. Taylor. “Fast structure alignment for protein databank searching”. In: *Proteins: Structure, Function, and Bioinformatics* 14.2 (1992), pp. 139–167. DOI: 10.1002/prot.340140203.

[168] C. A. Orengo and W. R. Taylor. “SSAP: sequential structure alignment program for protein structure comparison”. In: vol. 266. Methods in Enzymology. Academic Press, 1996, pp. 617–635. DOI: 10.1016/S0076-6879(96)66038-8.

[169] S. Fotoohifiroozabadi, M. S. Mohamad, and S. Deris. “Samira-VP: a simple protein alignment method with rechecking the alphabet vector positions”. In: *Journal of Bioinformatics and Computational Biology* 15.2 (2017), p. 1750004. DOI: 10.1142/S0219720017500044.

[170] J. Razmara. “Flexible protein structure alignment based on topology string alignment of secondary structure”. In: *International Journal of e-Education, e-Business, e-Management and e-Learning* (2014). DOI: 10.7763/IJEEEE.2014.V4.294.

[171] K. Hung et al. “Enhancement of initial equivalency for protein structure alignment based on encoded local structures”. In: *IEEE Transactions on Information Technology in Biomedicine* 16.6 (2012), pp. 1185–1192. DOI: 10.1109/TITB.2012.2204892.

[172] J. Razmara, S. Deris, and S. Parvizpour. “TS-AMIR: a topology string alignment method for intensive rapid protein structure comparison”. In: *Algorithms for Molecular Biology* 7.1 (2012), p. 4. DOI: 10.1186/1748-7188-7-4.

[173] J. Roach et al. “Structure alignment via Delaunay tetrahedralization”. In: *Proteins: Structure, Function, and Bioinformatics* 60.1 (2005), pp. 66–81. DOI: 10.1002/prot.20479.

[174] A. Williams, D. Gilbert, and D. Westhead. “Multiple structural alignment for distantly related all structures using TOPS pattern discovery and simulated annealing”. In: *Protein Engineering* 16 (2004), pp. 913–23. DOI: 10.1093/protein/gzg116.

[175] T. Przytycka, R. Aurora, and G. D. Rose. “A protein taxonomy based on secondary structure”. In: *Nature Structural and Molecular Biology* 6.7 (1999), pp. 672–682. DOI: 10.1038/10728.

[176] A. P. Singh and D. L. Brutlag. “Hierarchical protein structure superposition using both secondary structure and atomic representations”. In: *Proceedings of the 5th International Conference on Intelligent Systems for Molecular Biology*. AAAI Press, 1997, pp. 284–293. ISBN: 1-57735-022-7.

[177] E. M. Mitchell et al. “Use of techniques derived from graph theory to compare secondary structure motifs in proteins”. In: *Journal of Molecular Biology* 212.1 (1990), pp. 151–166. DOI: 10.1016/0022-2836(90)90312-A.

[178] L. Holm and C. Sander. “Dali/FSSP classification of three-dimensional protein folds”. In: *Nucleic Acids Research* 25.1 (1997), pp. 231–234. DOI: 10.1093/nar/25.1.231.

[179] I. N. Shindyalov and P. E. Bourne. “Protein structure alignment by incremental combinatorial extension (CE) of the optimal path.” In: *Protein Engineering, Design and Selection* 11.9 (1998), pp. 739–747. DOI: 10.1093/protein/11.9.739.

[180] S. J. Emery-Corbin et al. “Annotation of the Giardia proteome through structure-based homology and machine learning”. In: *GigaScience* 8.1 (2018), giy150. DOI: 10.1093/gigascience/giy150.

[181] C. H. Bamford, A. Elliott, and W. E. Handby. *Synthetic polypeptides*. Academic Press, 1956.

[182] B. W. Low and H. J. Grenville-Wells. “Generalized mathematical relations for polypeptide chain helices. The coordinates for the pi helix”. In: *PNAS* 39 (1953), pp. 785–801. DOI: 10.1073/pnas.39.8.785.

[183] F. H. C. Crick and A. Rich. “The structure of collagen”. In: *Nature* 176 (1955), pp. 915–916. DOI: doi.org/10.1038/176915a0.

[184] M. R. Berthold et al. “KNIME: the konstanz information miner”. In: *Data Analysis, Machine Learning and Applications*. Ed. by C. Preisach et al. Springer Berlin Heidelberg, 2008, pp. 319–326. DOI: 10.1007/978-3-540-78246-9_38.

[185] A. Stivala et al. “Automatic generation of protein structure cartoons with Pro-origami”. In: *Bioinformatics* 27.23 (2011), pp. 3315–3316. DOI: 10.1093/bioinformatics/btr575.

[186] T. UniProt Consortium. “UniProt: the universal protein knowledgebase”. In: *Nucleic Acids Research* 46.5 (2018), pp. 2699–2699. DOI: 10.1093/nar/gky092.

[187] A. Gaulton et al. “The ChEMBL database in 2017”. In: *Nucleic Acids Research* 45.D1 (2016), pp. D945–D954. DOI: 10.1093/nar/gkw1074.

[188] B. Huang and M. Schroeder. “LIGSITEcsc: predicting ligand binding sites using the Connolly surface and degree of conservation”. In: *BMC Structural Biology* 6.1 (2006), p. 19. DOI: 10.1186/1472-6807-6-19.

[189] W. H. Suters et al. “A new approach and faster exact methods for the maximum common subgraph problem”. In: *Computing and Combinatorics*. Ed. by L. Wang. Springer Berlin Heidelberg, 2005, pp. 717–727. DOI: 10.1007/11533719_7.

[190] T. Akutsu and T. Tamura. “A polynomial-time algorithm for computing the maximum common connected edge subgraph of outerplanar graphs of bounded degree”. In: *Algorithms* 6.1 (2013), pp. 119–135. DOI: 10.3390/a6010119.

[191] J. W. Moon and L. Moser. “On cliques in graphs”. In: *Israel Journal of Mathematics* 3.1 (1965), pp. 23–28. DOI: 10.1007/BF02760024.

[192] C. Orengo et al. “CATH – a hierarchic classification of protein domain structures”. In: *Structure* 5.8 (1997), pp. 1093–1109. ISSN: 0969-2126. DOI: 10.1016/S0969-2126(97)00260-8.

[193] S. Rao and M. G. Rossmann. “Comparison of super-secondary structures in proteins”. In: *Journal of Molecular Biology* 76.2 (1973), pp. 241–256. DOI: 10.1016/0022-2836(73)90388-4.

[194] B.-G. Ma et al. “Characters of very ancient proteins”. In: *Biochemical and Biophysical Research Communications* 366.3 (2008), pp. 607–611. DOI: 10.1016/j.bbrc.2007.12.014.

[195] G. Caetano-Anollés, H. S. Kim, and J. E. Mittenthal. “The origin of modern metabolic networks inferred from phylogenomic analysis of protein architecture”. In: *Proceedings of the National Academy of Sciences of the USA* 104.22 (2007), pp. 9358–9363. DOI: 10.1073/pnas.0701214104.

[196] I. Hanukoglu. “Proteopedia: Rossmann fold: A beta-alpha-beta fold at dinucleotide binding sites”. In: *Biochemistry and Molecular Biology Education* 43.3 (2015), pp. 206–209. DOI: 10.1002/bmb.20849.

[197] K. E. Medvedev, L. N. Kinch, and N. V. Grishin. “Functional and evolutionary analysis of viral proteins containing a Rossmann-like fold”. In: *Protein Science* 27.8 (2018), pp. 1450–1463. DOI: 10.1002/pro.3438.

[198] S. Annavarapu and V. Nanda. “Mirrors in the PDB: left-handed alpha-turns guide design with D-amino acids”. In: *BMC Structural Biology* 9.1 (2009), p. 61. DOI: 10.1186/1472-6807-9-61.

Appendix

6.1 SCOT

6.1.1 PDB Files

05N, 0EA, 0FL, 0LF, 0Y8, 0YG, 11Q, 2MT, 3PX, 4FB, AET, AME, AYA, BMT, CXM, EME, FC0, FME, FP9, GNC, HY3, HYP, HZP, I4G, IML, IPG, MAA, MEA, MGG, MH6, MLE, MME, MMO, MP8, MPQ, MVA, NCB, NLY, NMC, NZC, OTH, PCA, PH6, PRJ, PRO, PRS, PXU, SAC, SAR, SC2, SL5, THC, TYJ, UMA, WLU, XPR, YNM, ZYJ, ZYK

List 6.1: Residues to which no artificial hydrogen is added during parsing by SCOT.

00E, 00I, 00S, 010, 011, 01B, 02G, 02J, 02N, 03O, 04D, 05W, 0A9, 0AD, 0DQ, 0HQ, 0JT, 0JU, 0MG, 0OB, 0QE, 0QZ, 0R8, 0TJ, 0UH, 0W5, 0W6, 0XM, 0XN, 0YA, 10C, 11Q, 125, 126, 127, 12A, 175, 18M, 18Q, 1AP, 1BO, 1C3, 1CC, 1FC, 1HB, 1HD, 1KC, 1LU, 1MA, 1MG, 1OL, 1QQ, 1RN, 1SC, 1U8, 1VR, 1ZN, 22Q, 22W, 23G, 24M, 24O, 28H, 28K, 2A1, 2AR, 2AT, 2AU, 2BD, 2BT, 2BU, 2DA, 2DT, 2EG, 2GT, 2JF, 2JG, 2JV, 2KT, 2MA, 2MG, 2MU, 2N2, 2NT, 2OP, 2OT, 2PP, 2PR, 2SG, 2ST, 2UC, 2UE, 2X0, 32L, 34H, 39Y, 3A5, 3AZ, 3BY, 3CN, 3DA, 3FB, 3ME, 3PA, 3V2, 3V3, 3V7, 40A, 40C, 40G, 40T, 47C, 48V, 4AR, 4BA, 4CG, 4F3, 4FU, 4G6, 4H0, 4KY, 4L0, 4L8, 4LT, 4M9, 4MM, 4N7, 4N8, 4N9, 4NT, 4NU, 4OC, 4PC, 4SC, 4SU, 54L, 56J, 5AA, 5AT, 5BU, 5CG, 5CM, 5FC, 5FU, 5HC, 5HT, 5HU, 5IC, 5IU, 5MC, 5MU, 5NC, 5PC, 5PY, 5R0, 5R5, 5SE, 5SQ, 5UA, 5VV, 5XU, 5ZA, 60H, 62H, 63G, 63H, 64P, 64T, 66N, 68Z, 69P, 6E4, 6F5, 6FH, 6HA, 6HB, 6HC, 6HG, 6HT, 6IA, 6J9, 6L3, 6L9, 6MA, 6MZ, 6NA, 6OG, 6PO, 6RK, 6V9, 6VA, 6VF, 6VO, 707, 70U, 7AT, 7BB, 7DA, 7GA, 7GU, 7MG, 8AG, 8AN, 8BA, 8BB, 8FG, 8LR, 8MC, 8MG, 8OG, 92B, 9AT, 9BB, 9DK, 9DZ, 9GE, 9PR, 9TS, 9V7, A23, A2L, A2M, A38, A3A, A3P, A40, A43, A44, A47, A5L, A5M, A5O, A5R, A6A, A6C, A6G, A6U, A7E, A9Z, AA1, AAR, ABN, ABR, ABS, ABU, ACA, ACE, ACT, ACY, AD2, AEA, AF2, AFC, AG2, AH0, AHH, AJE, AKK, AKR, AKZ, ALQ, ALT, ALV, AMP, AMU, AMV, ANZ, AP7, APK, APN, AR0, AR7, ARF, AS , ASA, ASJ, ATD, ATL, ATM, AVC, AY0, AYE, AZ1, AZI, AZK, B27, B2A, B2F, B2I, B2N, B2V, B3A, B3D, B3E, B3K, B3L, B3M, B3Q, B3S, B3T, B3X, B3Y, B7C, BAL, BBC, BBS, BCX, BE2, BEZ, BGC, BGM, BIL, BLE, BMN, BNO, BOC, BOE, BOR, BP4, BPE, BRU, BTN, BUA, BZG, C12, C2L, C2S, C31, C34, C36, C37, C38, C42, C43, C45, C46, C49, C4S, C5L, C6G, C99, CA1, CAR, CBR, CBV, CCC, CCY, CDE, CDW, CEV, CF0, CFD, CFL, CFT, CFY, CFZ, CG1, CGN, CH , CH6, CH7, CHS, CJO, CKC, CLR, CLV, CM0, CMR, CMT, COI, CP1, CPN, CQ1, CQ2, CR0, CR2, CR5, CR7, CR8, CRF, CRG, CRK, CRO, CRQ, CRU, CRX, CSH, CSL, CTG, CWR, CX2, CXP, CY3, CYF, CZO, D00, DA , DAO, DBH, DC , DCL, DDG, DFI, DFO, DFT, DG , DG8, DGP, DHL, DIP, DIX, DIY, DKA, DM0, DMG, DNR, DOA, DOC, DT , DUZ, DYG, DYJ, DZM, E , E1H, E1X, ECC, ECQ, EDA, EFG, EHG, EIT, END, ESD, ETA, EXB, EXC, EYG, F3H, F4H, FA2, FAX, FBE, FBP, FDG, FDL, FE3, FHU, FMG, FMT, FMU, FOR, FOX, FPA, FPR, FRD, FUC, FUL, FUM, G2L, G2S, G31, G36, G38, G3A, G46, G47, G48, G49, G7M, GAL, GAO, GAU, GCK, GDO, GDP, GF2, GFL, GG7, GIC, GL3, GLC, GLK, GLZ, GM8, GMA, GMP, GMS, GMU, GN7, GNE, GNG, GOA, GPN, GRB, GS , GSR, GSS, GTP, GVE, GX1, GYC, GYS, H2U, HAO, HAQ, HCI, HEU, HF2, HG7, HIA, HM7, HM8, HM9, HMB, HMR, HN0, HN1, HOA, HPH, HR7, HS9, HSL, HSO, HSV, HT7, HY1, IBU, IC , IEY, IG , IGU, IIC, IL0, ILM, IMC, IP8, IPI, IU , IVA, JDT, JG3, KAC, KAG, KCQ, KI2, KPN, KWS, L2O, L3O, LAC, LAL, LCA, LCC, LCG, LEN, LGP, LHO, LHV, LIG, LKC, LML, LOV, LPD, LPL, LTA, LYJ, LYK, LYN, LYT, M1G, M2G, M5M, M9P, MA6, MA7, MAN, MAZ, MBN, MCM, MCR, MCY, MDF, MDO, MDP, ME6, MFC, MFD, MG1, MH6, MH9, MHE, MHW, MIA, MKE, MLI, MLL, MMT, MN1, MN2, MN7, MN8, MNU, MPR, MPT, MRG, MSU, MTR, MTU, MUB, MX3, MX4, MX5, MY1, MY2, MY3, MY5, MYR, N2G, N5M, N6G, N79, N7P, N80, NAG, NAK, NDG, NEH, NFA, NH2, NHH, NIT, NLO, NLW, NME, NMI, NMS, NMT, NOR, NRP, NRQ, NTA, NYG, NZH, O2G, OAR, OCE, ODA, ODR, OFM, OGX, OHU, OIL, OIM, OMC, OMG, OMU, ONE, OPR, OSL, OTT, OUD, OUE, OUH, OUI, OUK, OUR, P , P2T, P2U, P5P, PBE, PCS, PDU, PDW, PEA, PFX, PGA, PGN, PGP, PHA, PHL, PHQ, PIA, PIP, PIV, PJE, PJJ, PLF, PLJ, PLW, PN2, POL, PPI, PPU, PPW, PR3, PR4, PR7, PR9, PRN, PRQ, PRW, PSA, PST, PSU, PTL, PU , PVA, PVO, PVX, PXZ, PYO, PYR, QAC, QBT, QFG, QLG, QSC, QUA, QUO, R , RBD, RC7, RDG, RGL, RIA, RMP, RNG, RPC, RSQ, RTY, RUS, S2M, S4A, S4C, S4G, S6G, SC , SDE, SDG, SDH, SEL, SET, SIC, SIN, SMP, SMT, SNN, SPT, SRA, SSU, STA, SUI, SUJ, SUR, SWG, T23, T2S, T32, T38, T39, T3P, T41, T48, T49, T4S, T5O, T5S, T6A, TA2, TA3, TA4, TAF, TC1, TCP, TCY, TDY, TED, TEE, TFE, TFO, TFT, TGP, THO, TKL, TLB, TLC, TLN, TLX, TM2, TP1, TPC, TPL, TPN, TRW, TSP, TT , TTD, TTI, TTM, TYB, TYC, TYE, TYF, TYW, TYZ, U2L, U31, U33, U34, U36, U37, U8U, UAR, UBB, UBI, UCL, UD5, UDP, UF2, UFR, UFT, UMP, UMS, UMX, UPE, UPS, UPV, UR3, URE, URX, US1, US2, US3, US5, USC, USM, UVX, UZR, V9C, VAD, VAI, VLM, VLT, VME, VOL, W6Q, WCR, X , X9Q, XAD, XAL, XCL, XCP, XCR, XCT, XCY, XGL, XGR, XGU, XPB, XPC, XSN, XTF, XTH, XTL, XTR, XUA, XUG, XXY, XY1, XYG, XZA, Y , YAC, YCO, YG , YYG, Z , ZAD, ZBC, ZBU, ZCY, ZDU, ZGL, ZGU, ZHP, ZSC, ZSN, ZTH, ZZJ

List 6.2: Residues that are set as missing during parsing by SCOT.

6.1.2 Turn Dihedral Angles

Normal-3

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
1	1,852			−175.8	−82.3	61.1	178.3												−1.0
σ				4.8	5.7	9.7	4.7												0.4
2	334			175.6	74.3	−52.0	−178.1												−1.4
σ				5.7	6.5	10.9	5.4												0.7

Normal-4

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
1	70,498			−178.7	−61.6	−28.4	179.7	−78.2	−11.9	−179.7									−1.7
σ				4.4	6.8	11.4	3.7	15.9	15.2	4.9									0.7
2	6,970			179.4	−56.5	131.9	179.3	81.8	−0.3	179.4									−3.0
σ				5.1	6.5	7.4	3.5	13.7	14.8	4.9									1.2
3	3,227			179.3	53.9	37.1	178.2	75.3	8.6	−179.9									−1.9
σ				5.0	6.6	10.8	3.7	12.9	14.9	4.6									0.7
4	2,166			−179.1	56.8	−126.3	−179.3	−85.1	−0.1	−179.6									−3.1
σ				6.4	7.9	9.3	3.5	17.4	17.1	5.4									1.2
5	212			176.9	−54.2	141.0	5.6	−91.0	16.4	179.7									−2.8
σ				6.3	8.0	5.9	5.2	7.6	10.9	6.1									1.0
6	17			−179.7	−87.8	130.3	−4.0	−116.3	68.4	−173.0									−3.7
σ				6.1	9.7	10.9	7.6	13.8	13.0	7.2									1.6
7	9			−177.5	−72.5	76.8	−175.1	179.3	−32.5	176.0									−2.9
σ				6.5	9.9	9.6	7.6	17.0	10.4	4.3									1.4

Normal-5

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
1	221,356			179.3	−62.6	−41.5	178.9	−64.1	−41.8	179.2	−65.5	−40.7	179.5						−4.2
σ				3.5	5.4	6.7	3.0	6.6	6.6	3.4	10.4	7.7	3.6						1.4
2	283			−178.5	−57.5	130.1	−179.6	80.1	−3.2	−179.9	−118.7	−50.6	179.6						−3.8
σ				4.7	7.4	6.5	3.1	12.5	14.8	4.8	13.7	13.0	4.7						1.4
3	219			176.6	−83.8	143.5	2.8	−90.8	−2.6	−175.4	−67.1	−34.3	179.7						−2.9
σ				6.6	21.0	11.3	4.7	8.3	11.4	5.5	14.9	13.4	5.3						1.5
4	190			175.0	58.5	−150.5	−178.2	−63.0	−36.6	179.8	−63.0	−41.6	178.3						−1.3
σ				6.3	9.6	16.4	3.9	6.6	7.7	3.7	6.0	6.8	3.2						0.7
5	57			178.8	55.0	26.9	−178.6	63.2	16.0	−175.0	−126.4	−35.7	−177.6						−3.5
σ				4.7	6.3	7.3	3.4	7.3	9.9	5.3	14.9	14.4	5.6						1.3
6	52			−179.5	−57.5	150.2	−178.7	72.3	28.0	177.5	64.2	40.6	179.5						−1.8
σ				5.7	9.0	13.3	4.6	15.4	14.8	3.9	12.1	11.4	3.8						0.9
7	43			178.2	−59.4	142.5	−179.6	78.0	−56.5	−179.9	−70.5	−34.0	−179.3						−4.0
σ				5.1	7.1	7.3	4.7	9.7	8.5	5.0	14.4	12.2	5.0						1.6
8	28			−177.6	−64.3	−17.3	176.9	−82.4	−1.2	176.6	112.5	29.4	−179.3						−2.5
σ				4.6	8.8	11.2	4.1	11.6	8.3	4.7	13.5	13.7	4.0						1.1
9	31			179.4	55.4	44.7	−179.6	62.3	43.9	177.9	69.0	35.1	179.0						−3.3
σ				6.1	6.6	9.3	4.0	8.2	11.7	4.9	10.5	13.9	4.2						1.4
10	20			−178.9	−66.6	−22.8	−179.9	−98.5	72.7	176.6	84.2	15.5	178.6						−2.4
σ				4.2	5.2	8.9	3.1	13.3	15.6	3.6	25.3	20.3	3.0						1.3
11	21			−177.6	52.9	−137.8	175.7	−105.8	10.6	−176.9	−97.8	−36.6	−175.8						−3.0
σ				4.0	6.4	5.7	4.0	11.0	9.3	5.3	20.0	12.1	5.7						1.3
12	17			−178.4	60.9	−151.2	−177.1	−80.3	−28.3	−170.0	−135.2	1.3	−178.9						−3.4
σ				4.0	6.2	7.7	2.8	9.8	11.4	5.7	13.1	11.9	3.5						1.4
13	17			177.3	63.2	−134.3	179.2	−97.2	74.1	−178.6	56.8	36.5	177.3						−4.2
σ				3.7	8.9	10.5	2.5	10.4	14.7	3.5	7.6	7.0	4.1						1.2
14	9			178.4	54.3	35.7	174.7	83.6	−38.3	−172.4	−141.9	10.5	179.6						−4.3
σ				5.2	3.6	8.5	4.8	8.0	8.9	4.8	6.8	4.7	3.3						1.4

Normal-6

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
1	8,430			179.1	−63.8	−41.7	180.0	−64.0	−31.1	179.8	−91.3	1.5	178.8	74.7	26.9	179.3			−4.0
σ				3.5	5.6	7.7	3.2	7.2	9.8	3.7	12.5	11.9	4.6	19.2	16.8	4.2			1.3
2	2,157			−178.7	−66.5	−40.9	−176.8	−81.7	−35.7	−174.9	−99.9	−44.8	−175.9	−91.7	−46.5	−177.1			−4.1
σ				4.3	9.2	10.8	4.4	13.6	17.6	5.1	21.1	23.7	5.0	25.5	14.3	5.1			1.6
3	80			178.9	54.8	49.7	179.1	59.7	36.9	178.8	83.7	4.7	−179.4	−82.4	−21.6	−179.9			−3.2
σ				4.8	6.9	10.8	3.3	7.8	12.4	3.5	16.0	17.7	5.2	17.8	13.9	4.8			1.1

Table 6.1: Continued on next page.

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
4	67			−177.1	−65.6	126.7	−179.0	79.1	0.3	177.9	−119.0	−70.6	177.9	−91.5	−26.6	−180.0			−2.9
σ				5.3	8.4	6.9	3.9	12.9	13.9	4.9	11.9	18.7	3.6	20.0	14.8	6.0			1.3
5	22			179.3	−97.1	−28.6	−179.7	−135.2	101.5	−4.2	−74.3	166.0	−178.6	−71.5	114.5	180.0			−3.2
σ				4.7	14.1	20.3	3.3	8.5	11.7	5.2	6.1	8.4	4.0	7.9	12.1	4.9			1.3
6	22			175.5	77.0	−178.2	−177.5	−63.2	−29.0	179.5	−91.4	0.9	178.8	53.7	56.3	178.8			−2.9
σ				7.3	9.4	11.5	4.9	7.3	9.8	4.2	12.9	13.0	4.7	9.8	7.9	4.4			1.2
7	16			172.6	101.9	173.6	175.1	−130.2	107.8	−2.0	−72.3	161.4	−176.1	−65.0	134.1	175.3			−3.5
σ				4.6	12.1	11.3	4.7	6.0	7.2	4.2	4.9	5.9	3.0	6.7	5.7	5.4			1.1
8	12			−178.6	−71.3	−28.1	−176.4	−79.2	−25.8	−177.0	−67.1	−54.3	−7.8	−91.2	156.2	177.5			−3.2
σ				4.2	9.6	12.0	4.5	11.1	15.2	5.0	11.5	2.2	5.8	4.4	8.9	6.4			1.2
9	8			180.0	−74.2	123.1	−177.6	96.2	−57.9	−177.8	−72.6	−37.1	−180.0	−108.8	−45.3	178.0			−3.9
σ				3.4	13.2	21.7	2.8	28.1	10.2	4.0	13.6	11.7	6.3	21.5	9.3	6.2			2.1
10	8			179.1	−61.2	145.3	−178.3	96.1	−135.8	−172.0	−59.7	−38.2	−178.9	−62.4	−51.2	179.7			−2.7
σ				7.6	4.7	16.0	5.9	17.3	11.6	3.7	5.5	8.8	4.8	5.5	7.6	3.2			1.1
11	9			−177.8	−79.5	−10.5	−179.5	−107.7	42.3	173.8	125.2	−29.6	−178.1	−65.9	−39.4	−179.9			−2.8
σ				4.2	9.2	10.5	4.2	13.4	10.9	7.8	21.8	15.2	4.6	11.6	12.7	2.7			1.1
12	8			179.6	−60.7	129.4	−178.5	74.5	3.0	−179.3	−131.0	29.2	178.5	115.7	−16.6	177.9			−3.1
σ				5.0	4.5	2.7	1.9	9.4	9.6	2.0	11.2	16.5	4.3	17.3	16.2	6.0			1.0
13	9			−179.2	−92.2	−70.7	−176.5	−66.1	125.2	−179.6	74.1	7.0	179.0	−127.5	−91.4	178.5			−2.7
σ				6.7	8.5	3.9	4.2	6.5	7.2	2.4	11.0	10.0	3.9	10.7	12.9	3.4			1.7
14	8			−178.7	−61.8	−40.4	176.5	−89.3	−38.0	−173.7	−122.3	89.9	2.7	−133.9	28.9	−175.2			−4.1
σ				2.9	4.9	6.3	2.8	8.4	13.7	3.8	11.5	9.0	6.4	13.8	14.2	7.9			1.5
15	10			−178.7	−118.3	128.5	3.7	−91.5	−4.9	−171.7	−58.7	−46.7	−178.4	−94.8	−42.2	−180.0			−3.8
σ				8.2	13.5	8.3	4.8	3.7	11.8	6.2	8.5	7.0	3.9	16.3	9.5	5.6			1.4
16	7			176.9	−122.4	163.9	176.6	52.1	63.6	2.2	−90.0	164.6	178.3	−64.6	124.8	−179.7			−2.6
σ				6.7	10.5	6.3	3.4	5.1	7.5	4.8	3.8	5.2	3.5	7.5	7.2	1.8			0.7
17	7			179.2	−68.3	−30.2	177.0	−84.3	−45.1	177.2	−145.1	122.0	7.9	−84.0	−5.7	−176.2			−3.8
σ				2.9	6.9	13.3	3.4	14.4	7.9	3.0	10.1	8.3	5.4	8.3	10.1	6.6			1.6
18	6			179.1	−75.0	158.2	−178.5	83.2	−55.0	174.5	−78.8	−19.2	177.7	65.0	41.6	−178.9			−3.0
σ				2.1	6.7	6.6	5.1	10.6	14.8	6.2	13.8	7.7	4.3	11.7	18.8	0.9			1.0
19	6			176.4	−65.0	−24.9	177.7	−74.0	99.2	−178.6	88.6	7.2	−176.1	43.7	58.4	−173.4			−4.0
σ				5.2	5.0	12.1	3.4	5.2	10.2	2.2	15.2	8.1	6.3	5.9	5.1	6.1			1.3

Open-4																			
C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	d
1	39,622			180.0	−86.1	−21.0	180.0	−123.8	135.1	179.4									7.1
σ				5.2	17.0	18.3	5.2	28.3	32.7	6.0									0.7
2	24,781			−178.6	−87.2	−15.3	−176.4	−82.1	−17.8	−179.0									6.4
σ				5.1	19.5	21.4	5.2	25.8	27.6	5.0									0.9
3	6,989			179.0	−106.1	141.4	177.5	61.6	34.5	179.4									6.4
σ				5.6	30.6	19.9	5.5	15.6	20.7	5.0									0.7
4	6,971			175.5	−94.2	177.0	−179.2	−60.2	−30.0	179.3									7.7
σ				6.2	31.2	15.4	4.9	8.2	12.8	4.1									0.3
5	3,398			179.2	80.2	5.9	−179.3	−93.5	−11.0	179.8									7.5
σ				4.6	15.5	18.7	4.8	24.7	28.0	5.4									0.5
6	3,014			−179.2	−97.5	2.2	179.7	78.2	14.7	179.5									7.6
σ				5.3	19.9	16.2	5.0	18.4	21.1	4.8									0.4
7	1,442			176.2	−115.0	122.6	−177.4	62.7	−130.0	−179.7									7.2
σ				5.8	26.7	22.0	6.0	12.1	14.0	4.2									0.5
8	1,333			179.1	84.7	−171.7	−175.9	−67.9	−26.6	−179.8									7.1
σ				6.0	18.6	22.4	5.2	15.2	15.3	4.6									0.8
9	1,429			173.4	−74.0	168.4	178.8	−58.8	123.2	−178.1									7.8
σ				6.4	15.8	22.5	5.4	12.3	20.5	5.1									0.2
10	1,146			−178.9	81.4	−153.7	−179.0	−78.3	121.5	−179.0									7.3
σ				5.8	19.5	25.2	5.0	17.9	24.6	5.7									0.6
11	1,051			178.4	−119.1	138.7	−1.7	−76.0	155.1	178.9									6.3
σ				6.0	29.7	19.3	5.1	13.5	13.2	6.5									0.9
12	742			−179.6	79.1	7.5	178.7	−150.3	158.0	178.3									7.7
σ				5.3	15.8	17.1	5.1	14.8	21.7	5.7									0.3
13	337			177.5	−98.6	105.0	−177.2	91.6	148.5	−178.6									7.3
σ				5.3	14.6	20.1	4.7	16.5	34.5	5.6									0.6
14	339			−179.9	−87.3	49.9	179.2	−158.2	160.3	178.1									7.7
σ				5.3	8.7	13.8	5.7	13.8	18.1	5.9									0.2
15	284			178.0	76.2	17.7	176.4	68.4	30.2	−179.5									6.5
σ				8.0	16.1	19.1	5.4	19.5	22.3	5.4									0.8

Table 6.1: Continued on next page.

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	d
16	206			177.1	78.5	−55.9	179.3	−146.5	152.5	177.7									7.2
σ				6.0	9.0	15.9	5.0	14.6	16.2	5.8									0.5
17	208			−1.8	−67.7	155.0	−179.2	−69.8	133.1	178.7									7.7
σ				4.8	6.4	13.4	4.7	12.1	23.2	5.5									0.2
18	189			5.6	−87.0	0.4	178.7	−90.9	147.9	178.0									6.7
σ				5.8	9.5	13.9	5.8	30.0	15.7	6.1									0.6
19	149			178.4	−92.7	4.1	−179.9	120.5	151.1	−180.0									7.7
σ				4.8	15.8	15.0	6.2	20.7	18.5	5.1									0.3
20	166			−179.3	88.7	−9.4	177.9	−101.2	91.4	−175.2									7.7
σ				4.6	10.4	10.8	4.8	13.7	18.6	5.3									0.2
21	84			−178.9	72.4	23.2	−179.6	145.5	−170.6	−179.1									7.2
σ				5.1	16.6	21.2	4.3	24.8	15.3	4.8									0.5
22	124			−2.1	−73.2	150.6	−178.0	−64.5	−32.2	−179.9									7.7
σ				5.5	7.9	12.9	4.9	9.9	9.4	4.4									0.2
23	87			178.0	−162.7	−177.6	177.7	−69.6	97.6	−175.0									7.8
σ				6.2	18.4	10.7	6.2	14.6	22.1	8.3									0.2
24	67			178.2	93.6	146.6	177.5	63.7	27.2	−179.4									7.2
σ				5.7	17.2	28.0	4.7	13.7	16.4	4.6									0.6
25	71			179.2	−83.9	−21.1	179.7	−127.7	134.0	−1.6									7.5
σ				5.0	17.0	19.3	5.1	19.2	13.4	6.0									0.4
26	68			174.7	−74.0	153.2	177.0	−82.4	128.9	−2.5									7.7
σ				5.7	12.3	18.4	5.9	26.0	17.2	6.2									0.3
27	54			−177.9	−85.6	64.4	−175.4	−126.6	16.5	−178.5									7.8
σ				5.9	5.3	5.6	5.0	13.2	10.8	4.8									0.2
28	62			179.7	74.9	11.3	−177.7	−120.3	−49.5	−178.2									6.9
σ				4.6	11.7	14.3	4.7	7.9	8.7	6.1									0.5
29	45			179.1	−96.6	127.9	176.4	75.4	−60.7	−179.3									6.3
σ				5.2	19.6	8.9	7.3	7.5	13.4	6.8									0.6
30	48			179.9	−135.4	137.0	2.4	−79.8	−11.1	−179.8									5.5
σ				4.2	14.2	10.2	4.3	7.1	9.3	4.7									0.7
31	52			3.3	−91.0	10.9	−177.0	−70.8	−22.9	178.4									5.4
σ				5.1	5.4	11.6	5.5	16.1	21.1	5.2									0.5
32	40			179.9	88.6	1.7	179.7	101.1	150.0	−179.1									7.5
σ				3.7	10.2	10.4	4.0	14.1	15.9	4.7									0.5
33	37			177.8	−60.0	146.9	2.3	−86.7	−4.1	179.9									5.9
σ				4.0	6.3	6.5	3.4	6.5	10.7	4.5									0.4

Open-5

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	d
1	41,710			−179.6	−65.7	−36.6	−179.1	−67.1	−27.8	−179.3	−80.5	−14.8	−179.1						6.9
σ				4.4	13.3	12.0	4.0	12.8	13.4	4.3	19.2	19.3	5.1						0.7
2	28,500			177.7	−90.5	155.2	−179.1	−59.1	−33.3	−180.0	−72.7	−23.7	179.3						6.6
σ				5.8	31.4	19.5	5.0	7.1	11.3	3.6	15.9	20.3	4.2						0.7
3	5,945			−179.5	−67.5	−25.8	−177.8	−98.9	−6.8	−178.6	−128.9	132.2	179.2						6.9
σ				4.7	11.9	12.4	4.7	17.8	17.5	5.3	22.4	38.1	6.8						0.9
4	4,660			178.5	−91.5	151.8	178.1	−56.4	134.8	179.0	81.1	2.0	179.3						6.7
σ				5.7	30.3	17.5	5.1	7.4	9.0	3.8	14.4	18.3	4.9						0.7
5	3,296			178.8	−76.1	−29.7	−177.5	−91.4	−33.4	−177.7	−126.3	96.7	−177.9						5.8
σ				4.6	13.7	14.6	4.9	18.7	12.9	5.3	19.3	33.5	5.9						1.1
6	3,659			−177.9	−73.3	−27.7	−178.4	−93.0	−7.4	179.5	71.4	20.9	179.5						6.9
σ				4.5	16.5	16.0	5.0	18.8	16.2	5.4	15.6	21.1	4.8						0.8
7	2,631			179.1	−81.7	−16.2	177.9	−120.7	163.4	178.8	−64.6	132.4	−179.9						7.4
σ				5.4	16.2	23.4	5.4	28.6	25.3	5.5	13.6	20.5	5.5						0.5
8	2,027			178.3	77.3	−165.3	−176.8	−64.9	−26.2	180.0	−71.9	−23.9	179.2						5.7
σ				6.0	21.9	33.9	4.8	11.6	16.2	4.4	16.1	20.7	4.6						0.8
9	2,226			178.0	−123.2	127.2	180.0	53.9	37.8	177.9	74.9	8.2	179.9						5.5
σ				5.2	30.4	16.8	5.2	7.8	11.8	3.8	12.1	15.7	4.4						0.5
10	2,054			−179.8	52.9	39.7	177.6	78.5	4.2	179.7	−110.4	139.1	177.9						6.7
σ				4.7	6.2	9.0	3.4	10.9	13.3	4.1	18.1	18.2	5.3						0.5
11	1,422			177.3	−119.4	123.9	−178.5	61.4	−128.2	−179.5	−87.5	−0.2	−179.3						5.8
σ				6.3	24.4	26.0	6.8	12.2	13.6	4.3	17.8	19.0	5.2						0.8
12	1,283			−179.1	−66.4	−29.7	−178.7	−93.9	−5.8	−179.5	106.2	179.5	−178.2						6.7
σ				4.6	11.9	12.5	4.9	15.4	18.1	5.1	27.7	23.2	4.9						0.8
13	1,218			179.8	−61.3	129.9	179.0	76.1	7.1	179.9	−125.2	153.1	177.9						7.3
σ				4.8	15.8	10.2	3.8	13.4	17.6	4.9	20.7	26.2	6.3						0.6

Table 6.1: Continued on next page.

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	d
14	1,075			179.3	−88.1	−11.9	178.2	−105.3	173.8	−179.0	−66.3	−24.1	179.8						7.5
σ				5.4	18.8	18.9	6.0	28.3	19.7	5.2	13.6	19.1	5.2						0.5
15	960			177.7	−94.4	163.5	179.1	−63.6	−33.2	176.7	−116.0	107.2	−177.5						7.2
σ				5.9	31.9	19.6	4.9	9.5	13.7	5.6	19.6	27.8	6.0						0.7
16	1,058			−176.0	60.4	−123.4	−179.8	−93.3	5.4	−178.3	−102.9	138.9	176.8						6.5
σ				5.8	9.1	11.1	3.5	12.6	14.7	5.1	19.7	22.0	5.6						0.6
17	851			−178.8	−88.8	−24.0	−178.5	−84.3	132.5	179.0	68.9	19.0	−179.5						6.9
σ				5.4	18.7	19.6	4.9	32.7	17.4	4.3	15.4	20.9	5.0						0.9
18	688			−179.8	−56.6	130.3	179.8	82.5	−3.6	−178.4	−103.5	−1.9	−178.7						7.0
σ				4.9	7.6	8.0	3.5	13.3	14.2	5.5	24.7	29.4	5.1						0.8
19	453			179.4	−81.5	−30.8	−177.6	−87.5	−33.4	−178.3	−157.9	161.9	177.1						6.1
σ				5.2	19.2	19.6	5.2	15.6	13.2	5.2	12.2	14.1	6.0						0.9
20	414			−179.8	−91.4	−0.3	178.9	87.5	7.1	−179.9	−100.7	1.6	−179.9						7.6
σ				4.3	17.1	14.1	4.7	13.4	13.4	4.9	23.3	31.7	5.3						0.5
21	381			177.8	57.1	32.7	178.3	76.6	10.4	−177.8	−77.4	−22.8	179.8						6.6
σ				5.0	8.1	11.3	4.0	14.2	13.4	4.9	18.6	20.2	5.2						0.7
22	319			177.9	−96.5	144.7	179.2	82.5	−156.8	−178.0	−82.1	122.3	−179.3						7.1
σ				5.4	27.4	19.2	5.1	17.1	16.1	5.2	16.7	24.7	6.1						0.7
23	310			177.4	−89.1	150.3	176.6	−108.8	132.6	−1.5	−73.9	151.7	−179.3						6.9
σ				5.8	26.0	15.6	6.3	26.3	16.0	4.6	8.9	13.6	7.4						0.8
24	287			180.0	−143.4	166.2	175.3	64.9	−144.1	−176.5	−61.1	−33.1	179.7						6.4
σ				4.8	13.7	12.6	5.2	13.9	14.7	4.0	7.0	9.5	3.4						0.8
25	194			179.7	−91.4	153.1	174.6	−61.4	144.4	4.9	−88.4	4.9	179.9						6.4
σ				5.2	23.0	17.6	5.7	14.6	7.1	4.8	7.2	14.3	7.5						0.7
26	148			−178.9	89.9	−14.5	179.1	−112.8	−39.0	−179.9	−134.1	145.1	177.5						7.1
σ				4.2	14.0	14.6	5.6	17.9	15.7	4.8	18.8	20.9	6.2						0.6
27	158			−179.7	−82.1	−27.0	179.1	−143.1	151.3	177.0	74.2	−155.9	−177.3						7.2
σ				5.4	16.4	15.3	5.3	15.6	17.6	6.0	15.4	17.8	4.3						0.6
28	139			−179.1	75.9	9.6	178.8	−157.2	166.9	177.0	−69.0	126.7	−179.6						7.5
σ				5.6	16.7	23.0	5.5	14.5	11.2	5.9	12.0	20.3	6.1						0.4
29	144			177.3	−132.9	120.0	−2.1	−69.1	161.1	179.8	−64.2	144.7	177.9						7.0
σ				6.3	12.2	11.1	4.6	7.9	10.8	4.5	7.7	11.5	5.4						0.5
30	128			178.7	96.8	175.9	−178.8	−67.9	137.7	178.4	69.0	18.6	−179.9						5.9
σ				5.6	18.4	21.9	4.6	18.8	12.2	4.6	14.2	20.9	4.1						0.8
31	123			−179.8	−87.9	59.5	−179.1	−149.5	173.5	−179.2	−65.1	−25.9	179.6						7.5
σ				5.1	9.2	18.5	6.8	16.5	13.1	5.6	12.1	18.3	3.7						0.5
32	99			179.5	−114.5	−79.6	178.5	−91.6	−18.5	−179.9	−109.6	124.2	178.9						6.1
σ				5.6	16.0	22.4	8.4	19.0	14.9	6.2	22.7	18.8	7.2						1.1
33	109			178.3	75.9	15.2	−178.6	−111.5	5.4	179.4	82.0	15.9	−179.4						7.4
σ				5.0	13.3	14.2	4.0	16.0	11.2	4.3	15.6	15.5	3.8						0.4
34	100			−178.6	−106.9	5.8	−178.2	93.5	−14.5	−179.5	−120.4	−37.5	−178.6						6.3
σ				6.2	14.0	10.2	3.8	15.5	16.1	3.4	12.8	17.3	4.3						0.9
35	70			179.2	90.9	−173.4	−174.9	−79.4	−20.6	−179.2	−122.7	102.1	−178.5						6.5
σ				5.7	13.0	15.8	4.9	12.9	14.2	5.0	15.6	30.3	6.6						0.9
36	105			2.7	−90.6	1.3	−175.6	−58.3	−36.6	−179.6	−68.1	−25.2	−179.3						6.6
σ				4.5	8.5	13.2	5.5	6.9	11.3	3.9	13.7	17.8	4.5						0.6
37	72			−178.4	−90.1	3.4	179.4	−108.6	−131.1	−176.4	−65.6	−34.3	−178.6						5.9
σ				5.9	13.0	21.6	4.4	13.1	15.3	5.6	13.9	10.7	5.7						0.9
38	81			179.6	77.3	10.5	178.2	−127.1	172.7	180.0	−64.3	−21.7	179.7						7.5
σ				3.4	12.9	18.0	4.4	22.0	10.9	4.4	11.2	13.8	3.9						0.3
39	63			177.1	−143.9	−126.0	174.9	−63.0	−34.5	174.3	−123.5	168.5	176.8						4.9
σ				4.9	19.3	13.6	5.6	9.3	8.4	5.7	13.5	11.5	5.9						0.8
40	65			−179.8	87.5	−7.6	−179.7	−113.9	−60.7	179.9	−88.9	−21.2	178.9						5.9
σ				3.6	11.4	12.5	3.9	12.7	13.2	3.9	18.4	14.5	5.4						0.7
41	64			179.3	−51.8	139.5	6.9	−88.3	10.5	177.6	−75.5	148.0	178.1						7.2
σ				6.1	7.0	5.9	5.0	6.7	7.1	5.4	12.2	13.6	6.5						0.6
42	49			1.8	−89.7	0.9	−177.6	−66.6	−36.4	177.8	−122.7	134.1	179.2						6.1
σ				3.7	5.7	11.3	5.0	10.3	9.6	6.4	19.8	21.4	5.3						0.8
43	57			−1.9	−73.9	156.9	−176.9	−58.7	−32.2	−179.0	−66.9	−22.7	179.1						6.9
σ				3.9	8.0	16.2	3.4	7.4	9.4	3.6	11.0	14.2	3.8						0.6
44	48			176.0	−55.9	142.1	3.5	−93.5	13.6	−177.2	−73.0	−19.2	178.2						6.0
σ				4.2	7.8	6.8	4.1	5.3	12.8	4.3	17.9	22.2	4.9						0.8
45	42			−179.3	−124.2	138.5	2.4	−81.7	−10.6	178.3	−136.2	146.4	178.6						4.7
σ				4.6	17.2	11.5	3.9	7.7	10.4	5.2	16.3	22.1	6.4						0.6
46	56			178.1	53.1	37.9	179.6	57.3	27.5	179.1	78.6	10.5	−179.2						7.5
σ				5.9	5.1	7.8	3.1	5.8	9.1	3.7	11.4	10.9	4.7						0.4

Table 6.1: Continued on next page.

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	d
47	33			179.1	75.2	−167.1	−180.0	57.2	−137.5	−178.3	−74.3	−16.5	−179.1						6.4
σ				5.0	11.1	14.3	3.3	6.5	15.8	3.1	14.0	18.9	4.6						0.7
48	41			178.2	−60.7	128.5	−178.5	84.4	1.0	177.9	81.8	179.2	−178.6						6.5
σ				4.4	5.6	5.9	2.7	7.9	8.4	5.9	16.9	28.5	4.7						1.0
49	32			179.0	−64.4	−23.6	−178.9	−104.5	5.7	179.3	−77.2	147.3	2.3						7.8
σ				3.7	6.2	6.2	3.4	10.2	13.6	4.1	22.2	7.9	5.5						0.2
50	24			−179.9	−107.7	13.8	179.9	103.5	−4.5	174.3	−139.7	164.5	179.4						7.3
σ				5.6	21.0	13.3	8.8	13.1	11.5	5.7	12.9	11.2	5.1						1.1
51	31			180.0	57.3	−126.4	−180.0	−80.6	−2.7	176.1	91.5	8.6	−178.6						7.5
σ				4.6	4.1	9.1	1.8	11.4	10.1	5.5	9.2	9.2	4.5						0.3
52	25			179.6	−103.2	12.3	179.9	96.7	−79.7	178.8	−81.0	−12.4	−179.9						6.4
σ				2.5	10.1	9.9	4.4	17.8	17.2	3.3	8.9	17.1	4.5						0.5
53	22			−175.5	84.0	−59.7	178.0	−62.1	−41.3	−179.7	−63.6	−35.3	178.9						7.0
σ				6.3	11.7	7.0	5.1	6.8	8.7	3.5	7.5	15.0	3.5						0.7
54	17			175.7	−128.7	99.5	−176.3	−52.8	138.8	4.3	−90.5	12.6	−177.3						6.6
σ				7.5	8.5	16.5	5.5	9.1	3.9	4.5	3.9	15.9	6.8						0.8
55	16			−180.0	−97.2	−28.5	178.2	−141.9	125.8	−6.4	−66.5	153.0	−179.1						6.7
σ				5.3	11.1	18.3	4.8	9.3	10.0	5.0	7.5	13.5	2.4						1.0
56	13			177.9	−134.4	105.2	0.6	−74.0	166.0	−178.0	−57.3	−35.3	−178.5						7.2
σ				7.0	5.2	19.4	4.4	11.7	5.4	4.0	7.4	5.4	3.1						0.5

Open-6

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	d
1	19,044			177.4	−86.1	147.2	−178.8	−58.5	−37.5	−179.9	−65.9	−36.0	179.2	−77.3	−30.8	179.9			6.8
σ				5.8	27.7	23.4	5.1	7.2	9.4	3.3	9.1	12.7	3.7	20.3	22.2	4.4			0.7
2	7,143			−179.4	−90.6	−12.1	179.9	−101.3	143.7	−178.6	−60.4	−32.4	−179.9	−71.8	−26.2	179.4			6.3
σ				5.5	19.1	19.6	5.4	33.4	24.2	5.1	8.0	12.1	3.7	16.3	20.1	4.3			0.9
3	7,070			−179.8	−63.6	−27.0	178.9	−90.1	2.3	178.2	82.3	16.4	178.2	−92.6	141.0	178.5			6.6
σ				3.6	7.5	10.6	3.8	12.2	11.5	4.6	17.2	16.2	4.1	20.9	23.3	6.0			0.7
4	6,539			179.9	−66.4	−40.6	−179.0	−73.0	−42.8	−176.6	−80.4	−38.9	−174.5	−97.5	−18.7	−178.5			7.3
σ				4.2	11.9	12.1	4.2	17.9	14.0	4.2	16.2	12.0	5.4	21.7	20.1	5.3			0.6
5	2,742			179.4	−73.1	−26.2	−177.5	−98.5	−18.4	−178.8	−136.7	146.2	178.4	−75.5	132.4	179.7			7.0
σ				5.0	15.1	14.9	5.0	18.3	21.3	5.4	35.5	34.4	6.2	18.1	24.7	5.5			0.8
6	2,988			−179.4	−67.2	−38.9	−178.1	−76.7	−33.3	−177.0	−93.2	−14.1	−179.9	71.0	20.9	179.4			5.8
σ				4.5	10.5	11.1	3.9	15.0	13.7	5.1	18.4	14.5	5.4	17.0	24.4	4.8			1.1
7	2,291			−178.6	−64.4	−30.8	−179.8	−88.4	−1.8	178.9	72.4	23.6	−179.0	−89.1	−12.6	179.3			5.7
σ				4.0	9.7	10.5	3.9	12.8	11.8	4.7	13.7	15.7	4.4	21.1	24.4	5.5			0.9
8	2,136			177.6	−86.6	172.1	−178.2	−62.3	−23.2	178.1	−88.9	2.8	178.4	83.3	8.7	178.1			5.7
σ				5.7	17.3	11.6	4.3	7.6	11.0	3.8	11.4	10.6	4.5	13.6	14.8	4.6			0.7
9	1,877			177.6	−126.1	126.3	−179.7	54.2	38.2	177.5	77.6	5.7	179.6	−109.8	141.8	177.8			4.8
σ				5.2	29.0	14.5	4.5	9.4	12.1	3.6	10.7	13.6	4.4	19.9	16.6	5.1			0.8
10	1,566			−179.2	−64.2	−30.1	178.2	−72.8	−34.3	−177.7	−90.2	−33.9	−178.3	−140.0	85.9	−176.5			7.3
σ				4.1	7.5	10.7	3.9	11.1	11.9	4.8	16.5	12.4	4.8	15.2	31.7	5.6			0.6
11	1,438			178.5	−67.6	−32.9	−178.6	−90.7	−25.4	−178.0	−128.6	105.5	−177.0	−63.9	−24.8	179.8			7.5
σ				4.5	9.7	10.3	4.1	16.6	17.5	4.7	24.5	31.4	5.8	9.8	13.2	4.4			0.4
12	1,114			−179.6	−93.3	91.4	179.2	−140.4	165.9	−178.9	−60.0	−33.9	−179.4	−76.2	−26.5	−179.7			6.9
σ				5.9	23.1	29.6	5.9	27.0	20.1	4.9	7.6	12.2	3.6	18.8	20.7	5.6			0.9
13	1,361			−179.1	52.2	41.3	177.4	78.4	3.8	179.6	−113.4	141.0	177.4	−77.5	131.0	179.2			6.9
σ				4.3	6.1	8.8	3.3	10.2	12.1	4.0	16.8	13.7	4.8	15.9	12.5	5.0			0.6
14	1,015			−179.3	−123.1	158.1	176.3	73.5	−157.1	−177.4	−64.2	−27.6	179.8	−71.5	−25.4	178.6			5.6
σ				5.8	26.6	22.9	5.9	22.4	32.7	5.0	12.5	16.2	4.3	16.8	21.2	4.1			1.0
15	1,079			175.7	−135.7	178.7	−179.0	−63.2	−28.5	−176.7	−106.9	−9.2	−177.3	−143.5	141.9	177.3			5.3
σ				6.2	21.7	15.2	5.4	10.4	13.2	4.4	18.0	21.2	5.5	20.8	30.8	7.0			1.1
16	1,111			176.9	−90.8	155.1	178.8	−54.7	131.9	179.3	82.1	−0.1	179.6	−108.7	150.9	178.4			6.8
σ				5.6	32.5	16.4	4.9	6.3	7.2	3.4	12.6	14.4	4.6	23.8	23.0	6.4			0.9
17	896			179.7	−65.2	−35.3	−178.3	−81.6	−39.9	−174.4	−101.9	−15.3	−177.9	86.6	177.3	−177.9			6.1
σ				4.0	8.7	11.4	4.0	14.0	11.1	5.2	18.6	15.0	5.7	17.4	27.2	5.2			1.0
18	864			−179.6	−89.4	−10.8	179.9	−99.0	144.3	178.9	−59.5	135.1	178.3	78.0	7.4	179.4			6.3
σ				5.5	17.8	18.9	5.4	32.0	19.9	4.9	9.9	11.5	4.0	16.1	19.9	5.3			0.9
19	813			−179.4	−91.2	−7.6	179.7	−91.5	162.9	179.8	−70.2	−26.2	178.5	−117.2	123.7	−179.8			6.9
σ				5.2	16.6	17.8	5.1	26.5	19.8	5.3	13.2	19.4	5.5	23.7	32.2	5.6			0.9
20	969			178.1	82.9	−179.8	−177.0	−61.4	−38.3	−179.7	−64.4	−40.0	178.4	−70.0	−36.6	179.3			7.1
σ				7.0	21.9	32.6	5.1	8.2	9.3	3.6	8.0	10.9	3.6	13.6	14.3	4.0			0.7
21	743			−178.6	−70.6	−22.1	−178.6	−97.7	−2.8	−178.8	−113.1	169.8	179.1	−80.4	−12.6	−179.5			7.0
σ				4.4	12.9	13.8	5.1	16.2	18.7	5.2	25.7	15.6	5.3	16.9	22.1	5.0			0.8

Table 6.1: Continued on next page.

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	d
22	766			175.1	−133.1	107.8	−174.8	60.4	−123.8	179.9	−94.1	6.3	−178.0	−106.3	139.9	176.0			4.6
σ				5.6	14.3	15.2	5.0	8.1	8.2	3.4	11.8	13.7	5.1	16.9	17.6	5.3			0.4
23	571			179.0	76.4	8.3	179.2	−107.5	153.3	−179.2	−60.9	−32.8	−179.6	−71.7	−25.8	179.0			6.5
σ				5.3	16.0	23.5	5.4	29.4	23.4	5.2	7.8	12.2	4.0	15.1	20.6	4.1			0.9
24	478			−179.7	−65.0	130.2	178.7	74.1	9.6	179.9	−126.7	159.6	175.7	−85.4	132.6	179.6			7.2
σ				4.8	17.0	14.4	4.1	14.7	20.0	5.2	21.6	17.8	6.1	21.7	22.2	5.9			0.7
25	526			−174.9	61.8	−124.3	179.9	−94.0	5.1	−177.7	−105.4	143.3	175.1	−93.4	132.6	179.6			7.1
σ				5.7	9.8	8.9	3.6	12.5	13.5	5.1	19.4	19.4	5.5	21.5	14.4	5.3			0.6
26	475			177.7	−77.3	145.7	179.6	−55.5	133.1	179.4	83.0	−8.9	−178.1	−96.3	−8.5	−179.0			6.6
σ				5.4	21.6	17.6	5.7	7.0	9.3	3.8	13.4	19.8	4.7	25.7	30.8	5.0			1.0
27	433			177.3	−114.1	171.7	−179.9	−61.0	−26.7	−179.0	−94.0	−0.1	179.9	124.9	−172.3	−178.7			5.1
σ				5.6	28.7	18.3	4.4	6.8	10.5	4.1	14.7	16.4	4.7	29.6	23.9	4.3			1.1
28	312			175.5	−72.9	160.6	−176.4	−59.6	−30.1	−179.1	−91.9	2.7	−178.6	−105.0	155.9	177.7			7.3
σ				5.8	13.6	14.4	5.6	10.9	11.9	4.0	12.9	14.3	5.7	20.1	18.5	6.7			0.6
29	256			−179.4	−66.9	−33.8	−177.9	−83.7	−36.9	−172.1	−117.6	−7.6	−177.7	−106.7	172.8	178.6			7.4
σ				3.6	10.2	11.9	3.8	12.4	11.8	5.5	12.9	15.3	4.8	30.4	24.0	6.8			0.7
30	258			179.0	−58.2	134.6	178.7	78.5	5.8	178.7	−106.5	164.8	179.3	−72.8	−22.5	−179.1			7.1
σ				5.0	9.4	9.8	3.9	12.9	16.9	4.4	22.8	12.7	5.5	17.2	16.3	4.7			0.8
31	158			−177.6	−99.9	6.2	177.7	70.8	−147.3	−177.2	−62.8	−24.8	179.3	−80.9	−13.2	−179.5			7.2
σ				6.1	19.4	20.3	5.3	22.0	22.1	4.5	8.9	14.5	3.8	17.9	21.2	5.6			0.7
32	131			−179.9	−92.0	148.5	174.9	−81.8	145.1	3.2	−89.6	−2.3	−175.7	−65.6	−32.8	179.7			5.5
σ				4.4	21.7	18.7	6.2	23.5	9.7	4.4	7.5	12.4	4.8	12.9	13.1	4.3			0.7
33	130			178.4	−101.3	173.4	177.9	−59.7	147.9	179.0	69.4	22.4	176.6	67.7	26.5	179.0			6.3
σ				6.7	20.4	18.5	5.6	11.1	13.8	5.7	14.9	16.7	5.5	13.6	16.1	5.0			0.8
34	101			178.6	−83.7	−11.2	−179.4	−115.6	35.0	176.0	65.0	31.6	176.8	82.9	11.3	178.6			7.4
σ				4.5	19.7	15.2	4.3	15.1	11.9	3.8	13.1	11.4	4.2	16.8	17.9	5.0			0.5
35	115			177.1	−98.9	118.8	179.2	−129.3	118.3	−178.0	51.7	41.1	177.2	81.6	1.5	−179.2			7.8
σ				5.8	20.6	12.4	6.1	14.4	12.3	5.2	4.7	7.9	3.3	9.6	10.7	4.4			0.2
36	88			−178.2	−100.2	−7.4	179.7	−110.5	154.3	176.6	−118.6	136.7	−2.4	−73.4	155.5	178.8			6.7
σ				4.7	18.0	15.9	5.4	32.6	17.3	5.2	26.2	14.0	4.4	8.0	14.2	6.3			0.9
37	113			180.0	−110.5	118.5	−179.0	−56.3	132.1	−179.8	82.3	−4.9	−179.5	−119.9	−58.8	−179.2			5.8
σ				6.1	19.3	14.0	4.3	5.6	5.5	3.1	12.2	12.4	4.8	11.3	13.6	4.9			0.5
38	105			179.9	−97.3	99.9	−179.4	−161.8	163.0	179.0	−57.3	131.4	178.6	85.9	−3.3	−179.8			6.6
σ				5.8	20.0	29.4	4.8	32.9	17.6	4.8	8.4	7.7	4.4	13.4	16.9	4.3			1.0
39	94			−179.1	−64.1	138.8	177.2	60.5	26.0	179.9	59.4	29.0	−176.9	−70.7	−24.1	179.0			5.9
σ				4.5	13.1	19.8	5.7	9.8	14.6	5.2	11.1	13.5	4.5	15.2	14.6	4.8			0.7
40	87			−179.7	84.2	−4.3	179.1	−117.8	−61.7	178.4	−91.0	−27.9	179.6	−138.5	153.8	177.2			6.1
σ				3.4	12.1	11.1	4.8	13.5	13.4	4.1	16.1	15.9	4.9	20.3	13.9	7.4			0.7
41	81			−179.0	−104.6	9.8	−177.9	98.1	−32.7	179.4	−110.1	−31.2	−178.8	−138.6	156.5	174.7			4.7
σ				5.4	12.9	10.3	4.2	12.7	25.8	3.1	23.4	17.4	4.9	17.0	15.0	6.8			0.9
42	85			179.1	76.6	−166.5	−176.8	−63.1	−25.0	−178.8	−98.2	9.3	179.4	−75.3	147.5	179.4			7.2
σ				6.8	13.3	22.2	4.4	7.3	11.1	4.4	10.7	12.7	4.9	32.1	22.4	5.4			0.7
43	85			−180.0	−69.1	142.5	178.5	72.9	15.4	−178.8	−105.4	−0.4	−179.6	74.4	17.8	−179.7			7.2
σ				4.8	16.4	13.4	3.7	12.2	14.3	3.8	18.8	13.9	4.6	13.5	15.0	4.1			0.6
44	73			−177.8	−94.0	−8.9	−178.8	−60.6	143.3	178.4	59.8	30.5	179.0	65.4	27.0	−179.8			6.4
σ				5.7	19.6	16.8	4.0	9.7	14.0	5.3	8.6	13.7	4.5	14.6	16.9	3.9			1.0
45	70			179.7	120.8	−165.2	179.0	−75.8	174.3	−178.5	−56.4	−37.3	179.7	−65.7	−34.8	179.1			7.2
σ				5.7	33.6	14.6	4.4	11.9	16.3	5.4	5.4	8.1	3.0	10.0	14.5	4.2			0.5
46	54			−179.5	−72.0	−34.3	−177.6	−91.6	−19.3	−177.8	−118.6	−86.9	−179.8	−177.0	169.6	179.5			5.7
σ				3.7	10.6	11.6	4.9	14.8	16.0	3.8	12.3	31.1	4.3	28.1	19.0	4.0			1.1
47	61			178.5	55.5	27.5	−179.3	66.0	13.9	−175.3	−121.1	−27.0	−177.3	−131.5	144.6	177.0			6.4
σ				4.4	7.2	9.4	4.0	10.0	12.4	5.2	21.0	15.8	6.3	22.9	21.9	5.8			0.7
48	60			177.8	101.4	3.6	176.0	74.1	−156.3	−177.8	−61.9	−32.7	179.2	−63.0	−39.4	179.0			6.8
σ				5.6	19.2	14.3	4.7	23.1	21.9	4.1	8.2	13.0	3.9	8.5	13.8	3.9			0.8
49	69			179.1	−105.1	146.5	178.2	55.6	−128.7	−178.8	−95.4	10.4	−179.6	−69.4	145.4	178.2			6.5
σ				4.5	20.3	21.7	4.5	7.9	7.9	3.2	9.9	12.7	4.6	10.2	14.7	4.6			0.8
50	61			178.0	−98.1	123.6	178.7	55.8	26.8	−179.2	63.1	16.1	−174.0	−128.2	−32.8	−176.7			5.5
σ				5.4	21.4	11.4	5.3	9.6	10.7	4.2	9.8	13.2	5.1	15.4	24.1	5.7			0.4
51	56			−177.7	−90.3	−17.5	−178.6	−144.7	171.7	174.9	74.2	−160.6	−176.7	−60.1	−34.2	−179.1			5.6
σ				5.4	13.5	13.8	5.0	13.8	11.2	6.1	14.7	20.4	4.2	8.0	10.2	3.0			1.1
52	53			176.0	−71.9	138.0	−179.3	−75.5	−17.9	−176.5	−108.9	−15.7	180.0	−141.7	147.3	−179.7			7.3
σ				5.7	13.4	11.7	5.2	12.0	11.7	5.3	17.6	18.5	5.4	12.8	19.6	5.8			0.6
53	59			178.3	−151.7	148.7	179.6	−114.4	134.4	176.1	59.2	−119.8	175.7	−54.4	−35.4	177.7			7.6
σ				4.4	23.7	17.9	4.0	10.6	10.2	4.3	6.9	7.2	3.9	5.2	5.6	3.0			0.3
54	54			176.5	−92.7	−172.9	−179.8	−73.2	−19.3	176.0	−99.4	134.7	−179.4	−57.6	138.4	178.2			7.4
σ				5.8	14.3	19.1	4.6	10.2	16.6	5.5	18.4	17.5	4.8	7.8	10.6	4.2			0.6

Table 6.1: Continued on next page.

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	d
55	50			179.5	81.6	3.2	177.4	−89.3	148.2	179.7	−57.4	131.7	179.9	81.3	1.4	179.5			6.8
σ				4.0	15.4	16.5	5.3	16.6	18.6	5.3	6.4	10.4	4.8	16.9	16.5	5.8			0.8
56	51			−177.0	−53.8	127.4	−179.9	76.2	1.5	−177.9	−121.1	−49.8	−178.7	−151.9	148.6	175.6			6.9
σ				3.9	11.2	5.7	2.2	12.7	14.1	4.6	12.8	13.1	5.0	12.7	14.1	5.9			0.6
57	47			−179.7	−85.7	68.9	−177.3	−149.5	162.2	−179.0	−69.7	−30.8	179.5	−140.0	136.5	178.8			6.7
σ				5.7	8.0	10.7	5.3	15.3	12.0	4.5	10.0	11.4	3.6	14.7	26.8	5.1			0.9
58	36			179.3	−69.9	−22.1	179.4	−94.5	6.3	178.0	111.3	0.0	178.5	94.4	174.6	−178.9			7.2
σ				3.8	11.6	13.0	4.1	14.2	11.8	3.7	13.0	11.2	3.5	16.1	22.4	4.5			0.6
59	40			178.7	84.2	8.1	177.3	−97.7	168.0	−179.3	−65.3	−31.7	175.8	−127.1	147.6	178.5			7.0
σ				5.1	20.2	16.8	4.8	22.4	12.0	4.0	9.9	13.2	5.1	21.2	19.5	7.2			0.8
60	37			−178.5	−103.5	8.1	−177.7	−94.4	151.4	176.1	53.8	−129.0	−178.4	−85.6	3.2	179.7			5.9
σ				5.8	14.6	16.4	5.3	19.4	18.1	5.8	11.2	9.5	4.7	13.1	16.1	5.5			1.0
61	44			−179.9	−58.9	131.4	−179.1	87.8	−10.1	179.6	−115.4	−59.5	−179.4	−98.8	−11.8	178.2			6.7
σ				3.3	6.5	6.1	3.0	11.9	12.1	3.6	11.0	9.1	3.5	20.4	17.7	5.2			0.5
62	34			175.3	−116.5	120.7	178.4	−138.4	110.4	−175.1	57.4	−124.2	179.3	−89.9	2.7	−178.4			7.8
σ				6.3	17.2	8.6	4.8	11.3	13.0	4.3	6.1	6.0	2.8	10.0	13.7	5.5			0.2
63	42			179.8	−76.4	148.7	−179.9	−69.3	−31.7	179.2	−132.1	152.9	177.9	56.6	37.7	−179.4			6.6
σ				5.9	26.1	10.2	3.7	11.3	12.8	4.2	17.5	16.1	4.0	11.3	17.7	5.4			0.7
64	34			175.9	−137.0	−126.2	173.5	−65.5	−33.5	174.3	−122.2	171.7	176.9	−111.6	131.4	−180.0			6.4
σ				5.3	12.4	12.4	4.1	9.5	9.7	5.4	10.7	11.5	6.8	21.7	13.8	5.2			0.6
65	45			168.8	−70.0	151.5	179.9	−83.0	−55.4	−176.9	−68.3	120.6	179.9	79.8	2.2	179.7			7.1
σ				9.5	11.1	13.6	4.5	16.7	18.4	5.6	8.9	10.7	2.9	9.9	11.3	5.0			0.6
66	34			179.8	−93.0	−8.1	−179.8	−97.7	151.5	174.5	−70.4	145.4	2.2	−86.9	−0.6	−178.3			5.8
σ				5.1	13.5	12.0	4.7	22.3	12.7	6.8	21.0	10.3	3.4	7.5	14.0	6.5			1.1
67	43			179.6	48.0	56.7	−174.1	−57.9	−34.0	179.6	−66.4	−35.9	178.8	−83.4	−31.0	−178.8			7.6
σ				5.2	6.7	8.2	5.2	4.7	7.3	2.8	7.7	12.7	3.1	22.9	16.4	4.7			0.4
68	27			176.6	−70.6	148.8	1.9	−88.0	−2.0	−177.9	−62.1	−41.1	178.9	−128.0	145.6	177.9			7.0
σ				7.0	13.5	6.2	2.9	5.5	9.7	3.6	6.7	7.4	6.0	14.0	20.0	5.3			0.8
69	23			178.2	57.6	30.4	178.3	76.5	8.2	179.4	−89.0	155.4	179.7	−77.3	−16.7	179.8			7.4
σ				4.9	6.6	10.7	2.9	10.7	10.5	3.7	19.5	12.7	5.3	18.0	20.5	4.8			0.6
70	30			179.9	83.7	172.8	−175.5	−64.2	−26.9	−179.5	−112.5	9.3	178.6	66.7	21.3	−176.5			6.2
σ				4.9	8.1	12.4	4.9	7.8	9.2	5.0	14.1	12.0	4.6	11.9	13.2	4.6			0.5
71	33			−1.7	−77.0	160.6	−174.5	−57.6	−37.7	−178.0	−62.2	−32.9	178.8	−74.9	−34.7	−179.5			6.7
σ				6.3	12.2	29.4	6.1	5.7	8.2	3.2	5.2	11.3	2.6	11.2	13.9	5.7			0.7
72	28			−177.4	97.0	−16.0	179.6	−120.5	−34.9	179.7	−146.8	167.2	176.4	−82.8	136.5	177.7			7.0
σ				4.8	14.0	11.6	2.9	15.6	13.4	3.7	12.5	14.3	6.2	20.7	9.5	5.3			0.5
73	27			−178.4	−77.7	157.3	178.5	−65.5	159.6	179.0	73.4	−136.7	−177.8	−68.6	−20.9	179.4			6.8
σ				4.4	11.0	17.5	5.4	12.3	17.7	4.5	15.0	6.3	5.1	11.9	15.6	4.5			0.8
74	30			178.5	−83.0	133.1	176.2	−54.9	143.0	8.4	−85.1	6.3	177.5	−70.1	151.6	−180.0			6.6
σ				5.5	15.0	29.1	3.7	5.4	5.4	5.8	6.2	6.0	4.8	7.7	9.7	11.7			0.8
75	28			−177.6	−109.0	12.7	−175.5	−59.8	−29.1	179.5	−101.2	11.1	179.4	88.6	2.9	−179.9			7.3
σ				6.6	16.7	15.5	3.8	6.5	7.2	4.0	15.8	12.3	4.5	16.1	16.9	6.3			0.6
76	19			178.7	−105.7	112.1	177.2	−124.5	−136.5	174.4	−61.6	−32.0	174.4	−126.4	169.0	177.0			6.1
σ				5.2	16.3	17.2	3.7	14.4	7.6	4.5	5.3	8.7	5.4	11.5	8.7	5.7			0.8
77	30			−177.9	−125.4	40.2	176.3	62.7	31.3	175.4	90.1	3.8	177.9	−71.6	−43.0	177.7			7.6
σ				4.1	5.8	5.5	3.7	8.8	10.4	4.5	7.2	6.7	3.2	7.3	7.6	2.3			0.1
78	17			177.2	51.8	−132.0	−178.2	−67.6	−15.6	−178.1	−82.8	−12.3	179.8	82.5	−162.8	−177.1			6.0
σ				4.7	15.1	10.1	3.7	8.7	9.8	8.5	7.3	13.5	5.6	16.2	15.4	4.8			0.9
79	31			−179.4	56.0	−127.5	−179.6	−83.2	2.9	176.1	87.2	11.0	−177.7	−57.4	−44.1	178.6			7.2
σ				4.7	5.5	6.2	2.2	8.7	8.7	5.0	7.6	9.0	3.1	5.1	10.1	4.3			0.3
80	13			−178.2	−79.0	−9.1	177.2	−92.9	−38.8	−178.9	−87.0	−30.1	−178.5	−159.6	−175.1	−179.9			7.0
σ				2.9	13.6	11.4	6.7	14.6	14.9	2.5	11.2	13.1	2.9	12.2	16.0	3.9			1.3
81	13			−178.7	−94.1	−3.1	−179.0	93.1	158.9	177.4	56.7	31.5	178.7	79.1	8.1	179.5			6.5
σ				5.0	10.2	8.6	3.4	7.6	7.0	3.1	5.9	8.5	3.2	8.0	6.3	3.2			0.5

Reverse-2

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
1	810	−147.4	−77.3	162.2	−82.8	175.7													−2.1
σ		16.5	14.4	9.6	14.7	12.3													1.0
2	79	−155.9	56.9	−165.2	82.5	−178.8													−1.5
σ		16.6	18.3	7.3	15.4	13.8													0.8
3	27	−111.6	−16.0	7.2	−81.8	−41.7													−1.4
σ		27.9	28.1	9.2	16.7	22.7													0.7
4	23	−129.9	−61.5	3.8	76.8	95.7													−2.2
σ		28.8	17.4	15.1	24.4	27.9													1.6

Table 6.1: Continued on next page.

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
5	8	122.8	92.5	1.5	−69.1	166.1													−1.1
σ		10.6	15.9	6.1	18.8	29.9													0.5
6	5	−159.4	43.3	−1.7	−82.0	−177.0													−0.9
σ		14.2	21.6	7.4	7.3	11.0													0.3
7	3	104.3	76.0	−0.6	−96.0	−59.7													−1.1
σ		23.2	6.1	9.6	14.7	12.8													0.8
8	2	−98.6	−75.1	6.3	21.1	−122.1													−1.3
σ		8.3	4.0	9.0	19.8	6.5													0.5
9	2	−165.5	−34.6	14.8	70.0	23.7													−0.9
σ		27.4	0.7	2.6	5.3	3.8													0.3
10	2	−135.5	−61.0	128.3	−119.6	−86.6													−1.6
σ		9.4	3.2	9.3	9.8	2.6													0.8
11	2	−163.4	−70.8	1.4	76.9	177.2													−1.2
σ		10.8	0.9	0.6	0.5	17.1													0.5
12	2	154.3	−30.2	−21.8	120.6	178.5													−0.9
σ		6.4	6.8	3.9	0.9	3.2													0.3

Reverse-3

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
1	88	−144.0	−116.7	173.8	−66.6	−31.8	173.9	−126.2	166.6										−4.5
σ		29.1	24.0	4.0	12.4	11.2	6.0	15.2	12.0										1.6
2	7	−127.8	31.7	−169.2	111.3	−24.9	172.4	−139.0	162.0										−3.4
σ		9.7	8.5	4.0	7.8	10.9	8.1	15.2	8.8										1.1
3	8	−145.9	87.5	−172.5	73.8	−57.8	173.6	−114.8	165.4										−4.1
σ		13.0	9.8	2.2	4.7	12.5	3.4	23.8	6.1										1.7
4	8	−160.0	97.5	−173.1	49.8	51.2	−176.1	120.0	168.9										−4.7
σ		4.0	10.9	5.4	3.9	5.6	2.5	11.7	16.9										1.1
5	7	−120.1	23.0	6.6	−86.0	−8.6	176.5	−77.6	153.4										−2.4
σ		22.8	25.6	7.5	8.7	8.0	3.1	19.8	20.9										0.9
6	4	81.1	109.2	10.7	−78.9	−19.7	179.9	−85.1	151.0										−4.0
σ		12.2	16.8	10.3	9.6	10.0	4.8	16.0	17.8										1.8

Reverse-4

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
1	1,654	−135.5	123.4	−179.3	51.9	41.6	177.4	78.4	3.5	179.6	−114.5	141.0							−4.6
σ		16.3	12.7	4.3	5.7	7.8	3.3	9.9	12.5	4.1	16.3	15.0							1.1
2	1,208	−130.1	−178.1	−179.5	−61.5	−29.1	−177.3	−105.7	−6.9	−178.0	−165.4	157.7							−4.2
σ		21.5	11.6	4.4	8.2	12.2	4.1	17.7	21.6	5.3	39.7	36.1							1.9
3	828	−132.6	107.7	−174.8	60.6	−122.9	179.9	−94.6	6.1	−178.0	−107.9	140.0							−5.1
σ		14.9	16.1	5.1	7.6	10.2	3.5	12.7	14.5	5.1	18.1	20.3							1.3
4	137	−141.4	174.0	178.2	−58.5	120.0	179.1	74.9	9.1	−178.6	−150.0	133.2							−3.3
σ		23.7	12.2	4.7	8.9	9.6	3.2	14.4	16.3	4.2	20.4	37.7							1.8
5	94	85.5	−6.8	178.7	−115.0	−60.9	178.4	−96.7	−24.7	179.0	−138.6	147.2							−4.6
σ		13.7	15.2	5.1	13.0	13.5	3.7	17.8	17.0	5.5	20.0	20.6							1.5
6	89	−104.2	10.6	−178.1	98.6	−36.2	179.1	−110.1	−30.1	−179.3	−139.6	155.8							−4.5
σ		11.2	10.1	4.4	14.5	26.1	2.9	23.2	17.3	3.8	17.0	15.8							1.4
7	12	−102.7	−86.5	176.3	−105.4	−87.7	171.3	−100.5	1.3	−178.5	−105.4	139.2							−4.0
σ		18.5	22.4	7.4	13.6	16.8	8.1	19.6	17.8	2.3	19.7	17.2							1.4
8	12	−136.1	88.5	−172.2	−44.5	134.9	4.3	−89.3	16.9	−176.8	−104.3	149.0							−4.3
σ		14.1	14.6	7.1	8.6	5.8	6.4	4.8	14.2	5.6	18.2	12.3							1.5
9	8	−107.8	−177.3	176.6	−61.0	−39.7	173.8	−131.2	120.2	179.0	77.3	−171.5							−3.9
σ		13.2	6.8	6.1	5.3	10.5	5.3	16.2	12.6	3.0	12.9	19.4							1.3
10	4	114.9	167.3	175.8	62.1	−122.2	−179.7	−71.5	−13.2	178.8	−114.0	94.0							−3.6
σ		12.3	15.9	2.3	7.7	4.0	3.6	7.0	8.4	2.2	6.7	18.6							2.2
11	4	−105.2	13.4	−179.6	78.3	94.8	−177.6	92.5	−6.2	176.3	−154.1	140.0							−5.9
σ		2.6	9.7	4.3	11.3	23.8	2.2	28.8	18.6	3.5	16.8	7.3							1.5

Reverse-5

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
1	1,902	−87.0	173.4	−178.3	−62.6	−22.4	177.9	−89.6	3.8	178.3	85.5	7.6	177.6	−86.1	145.2				−4.8
σ		14.8	11.2	4.3	7.5	10.8	3.8	11.6	10.6	4.5	13.1	14.5	4.4	17.0	14.4				1.4
2	170	−96.0	−33.9	176.0	−156.2	−178.0	−178.4	−60.6	−24.5	−177.3	−118.0	5.3	−175.8	−134.8	128.6				−3.7
σ		14.7	17.3	5.4	12.5	8.4	4.9	8.1	12.1	3.9	12.0	18.6	5.6	13.9	16.4				1.5
3	138	−111.6	−177.1	179.1	−59.8	147.6	178.4	68.4	24.7	176.4	70.8	22.9	178.7	−90.0	146.0				−4.5
σ		20.0	21.9	5.5	11.6	13.4	5.1	15.1	15.9	5.1	13.8	16.9	4.2	20.9	22.6				1.6

Table 6.1: Continued on next page.

C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
4	82	−117.2	146.1	−178.0	−64.0	−30.8	−178.2	−93.9	−36.5	−172.5	−108.3	−3.0	−178.3	77.3	−143.2				−3.1
σ		27.0	16.0	4.3	8.6	13.0	4.3	19.2	16.0	4.7	15.2	12.6	5.1	13.3	23.6				1.3
5	69	−119.4	116.3	−177.3	−51.8	129.2	179.4	75.1	3.1	−177.8	−127.0	−47.4	−178.0	165.9	171.5				−4.4
σ		18.8	14.3	5.2	7.0	7.0	2.6	12.2	13.2	5.3	10.2	18.8	4.6	49.0	36.3				1.8
6	67	−94.9	−29.1	176.6	−162.7	171.0	179.7	−55.0	123.9	178.9	73.2	11.0	−179.4	−129.7	118.4				−4.0
σ		16.3	17.9	5.2	17.9	8.7	4.8	6.1	6.8	3.6	11.7	12.5	4.8	11.7	18.4				1.4
7	61	−101.2	123.7	178.9	54.2	28.3	−179.5	63.8	14.4	−174.2	−129.9	−32.0	−176.9	−120.9	137.2				−5.0
σ		20.1	10.9	4.2	6.1	9.1	3.6	10.6	16.0	4.9	14.7	24.4	5.5	24.6	20.5				1.4
8	56	−124.5	111.0	−175.5	60.5	−148.9	−179.5	−92.3	−11.0	−172.3	−117.4	−12.0	−176.8	−124.1	143.1				−3.9
σ		14.7	22.0	5.9	11.3	18.8	5.0	18.6	21.9	6.0	24.6	23.3	5.9	25.1	24.3				1.6
9	29	−102.6	1.9	−179.2	95.5	156.7	178.2	56.5	32.2	179.3	78.9	7.4	−179.9	−96.0	141.0				−4.5
σ		11.9	13.0	5.9	14.1	13.4	4.4	8.2	12.2	2.6	11.2	14.0	4.9	15.9	14.5				1.3
10	26	−93.8	142.3	−178.6	−59.6	−31.9	−179.6	−88.2	4.6	179.4	110.7	150.9	−178.8	82.8	−177.5				−4.2
σ		16.6	13.2	6.3	5.6	9.8	3.0	15.0	17.6	4.2	18.1	10.8	3.1	15.9	20.2				1.5
11	17	−111.6	49.1	177.5	56.2	−129.7	−178.9	−70.2	−10.0	177.0	−84.8	−4.8	178.0	91.3	−158.1				−4.5
σ		16.5	17.3	5.7	7.4	6.2	3.7	10.1	11.1	3.3	12.2	15.9	5.9	23.2	22.3				1.7
12	15	−91.1	−5.6	178.7	83.2	3.3	−178.2	−123.1	20.4	179.2	94.3	4.2	179.3	−121.1	157.4				−3.6
σ		10.1	7.3	4.4	11.9	8.8	3.1	12.6	10.3	4.5	13.0	12.2	6.4	16.6	20.6				1.2
13	16	−94.3	−55.4	176.3	−164.8	141.6	179.2	51.2	−122.8	−179.3	−96.2	13.0	−177.5	−98.5	129.7				−4.3
σ		12.5	13.3	3.9	7.0	13.1	3.2	4.6	6.6	2.3	8.7	12.5	4.5	10.8	8.7				1.2
14	12	−110.3	138.3	−179.6	−59.1	127.1	−176.9	66.3	19.0	179.3	−136.9	−153.8	−178.1	−80.5	158.2				−4.4
σ		42.9	21.7	2.5	6.6	7.5	3.9	10.9	15.2	4.9	14.3	10.0	3.1	10.7	17.6				1.8
15	11	−139.3	89.0	−180.0	−70.6	141.2	2.6	−97.9	14.2	−173.8	−84.9	−38.1	178.6	−144.5	129.3				−4.0
σ		16.7	18.0	5.1	4.7	6.5	5.9	5.2	15.0	5.8	15.8	15.6	5.8	12.5	17.7				1.8
16	9	−138.1	137.6	−173.4	−62.0	−28.5	−178.0	−95.1	−30.3	−170.8	−115.5	−32.4	−177.5	−154.3	91.2				−2.2
σ		24.6	20.7	5.0	9.3	8.3	3.8	13.8	21.9	7.9	19.1	17.4	4.4	12.1	20.1				1.3
17	9	−106.8	1.8	179.3	72.9	6.7	−177.8	−130.9	−64.8	−176.4	−117.3	−16.7	177.9	−128.7	123.8				−3.2
σ		12.7	6.1	1.5	6.6	9.6	6.2	12.8	23.5	5.6	15.7	22.2	8.5	23.1	16.9				1.1
18	9	−132.3	146.3	178.9	53.0	41.9	174.8	81.9	−52.3	−171.0	−129.7	4.3	179.8	−96.0	138.1				−3.5
σ		13.7	17.1	5.5	5.5	9.4	4.5	9.0	9.9	7.1	14.1	17.1	5.8	10.2	15.9				1.5
19	8	100.0	−12.9	180.0	−88.1	134.2	−179.3	56.7	27.1	177.3	82.6	8.4	−179.3	−95.6	130.2				−4.2
σ		13.6	10.3	2.7	26.7	15.6	5.3	5.5	6.3	6.2	13.1	9.3	2.1	22.5	14.8				1.8
20	9	−155.4	151.1	−176.9	−60.6	−30.8	179.1	−103.5	12.8	−180.0	−135.2	−134.2	−179.1	−87.1	139.2				−2.8
σ		31.9	20.8	4.7	7.9	11.4	4.2	10.3	10.3	4.3	15.5	10.2	4.2	14.9	9.8				1.4
21	9	84.4	4.4	179.4	−128.5	149.1	176.0	51.4	−129.1	−178.9	−74.7	−8.3	179.9	−102.9	51.0				−2.3
σ		9.7	11.5	6.5	13.4	10.6	2.7	5.3	4.6	3.8	14.0	12.1	4.2	10.9	40.2				1.3
22	6	−94.1	−60.1	178.4	−176.7	155.7	172.1	55.9	30.3	−179.7	83.1	−2.3	177.5	−80.4	129.5				−5.3
σ		15.2	25.1	3.5	25.0	12.9	4.5	6.8	10.1	3.2	12.3	7.2	3.2	4.7	17.3				1.1
23	7	−143.5	169.3	176.7	59.7	−132.4	177.4	−94.2	69.7	−180.0	63.0	33.3	178.5	−95.5	130.4				−4.3
σ		11.4	3.1	3.8	10.6	9.0	3.6	12.0	16.5	2.4	9.9	15.4	2.8	16.9	18.4				1.2
24	6	−99.9	118.1	−179.6	−55.6	140.2	−0.1	−88.0	−31.8	−167.4	−144.7	30.1	179.2	−112.1	136.5				−4.2
σ		18.8	15.5	5.0	5.7	3.9	2.0	10.5	13.2	8.2	12.1	19.2	4.7	12.0	18.7				1.3
25	7	−131.9	118.0	−170.7	−58.8	−30.2	178.7	−85.6	−7.2	−178.0	−126.2	−71.7	−176.3	−170.9	177.9				−4.2
σ		14.9	15.1	6.8	4.8	16.8	3.4	18.3	14.9	4.1	17.5	13.4	5.0	22.9	14.8				0.9
26	6	−108.9	10.2	175.8	67.5	−34.6	−175.6	−101.9	−7.9	−179.3	−132.2	−53.9	−173.4	−140.1	162.6				−2.2
σ		1.3	3.2	2.4	4.6	7.6	1.1	6.5	8.4	2.6	3.2	4.0	2.2	5.3	2.9				0.2
27	7	−105.6	−165.3	175.5	−66.5	135.8	−178.4	71.3	−140.6	−178.0	−90.8	4.4	−176.1	−76.3	147.3				−3.3
σ		13.1	16.7	4.2	15.8	17.7	6.5	16.0	15.7	11.5	4.7	12.5	5.4	11.8	10.2				1.1
28	6	101.5	−5.1	176.6	−86.3	170.0	178.6	−59.1	−32.1	179.7	−85.8	−10.0	178.5	−143.5	70.0				−4.4
σ		13.0	16.3	2.7	19.4	11.6	1.5	9.9	11.9	3.2	10.4	16.6	1.9	12.4	10.1				1.5
29	5	−126.7	154.0	−178.3	−78.1	139.4	−4.8	−105.4	126.0	175.4	62.3	35.4	−176.6	−117.8	143.9				−4.2
σ		18.9	25.7	4.5	19.4	13.0	5.4	13.2	14.2	7.2	12.7	19.3	1.5	17.2	10.1				1.7
30	5	−164.0	−168.7	−175.6	−55.6	140.6	−176.3	147.1	−60.5	−5.8	−71.6	140.2	175.8	−124.1	144.5				−4.1
σ		25.4	22.0	5.7	5.6	9.6	5.3	11.1	4.8	5.3	9.9	12.9	2.7	13.2	8.3				1.1

Reverse-6

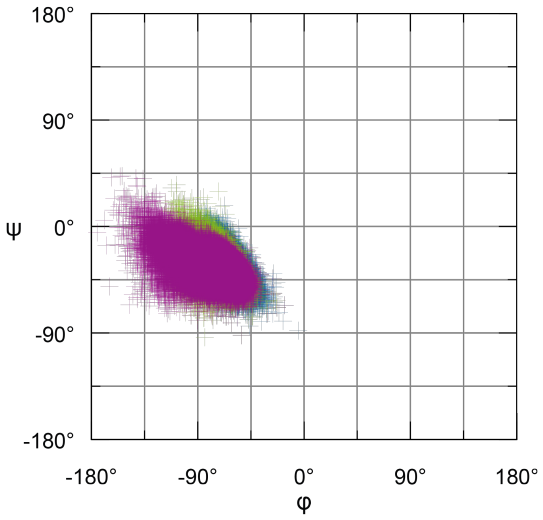
C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
1	1,141	−92.8	114.0	−176.3	−66.0	−29.3	−178.9	−74.6	−40.5	−177.9	−102.0	−9.3	−177.9	65.6	27.7	179.7	−105.2	146.0	−5.5
σ		22.6	15.1	4.5	8.0	12.8	4.1	13.2	12.3	4.2	16.3	11.6	4.8	12.4	20.5	4.5	30.4	15.1	1.3
2	168	−70.7	161.5	−177.3	−59.1	−30.2	178.9	−84.7	−2.3	178.7	70.3	14.1	−177.2	−96.2	−21.6	−178.9	−131.3	146.6	−5.0
σ		8.6	11.9	4.2	6.5	10.5	3.3	11.1	10.1	4.1	9.7	14.4	4.9	18.3	14.2	5.0	22.3	16.6	1.4
3	49	−75.7	129.5	−177.3	−55.3	134.9	178.0	70.4	16.2	179.2	−139.9	168.8	175.9	−97.4	15.8	−178.8	−92.7	136.9	−5.1
σ		16.3	14.9	4.2	8.0	8.3	4.1	12.1	14.6	4.5	13.2	10.8	5.7	9.9	21.7	4.1	22.3	16.4	1.1
4	37	−127.3	166.6	176.3	53.9	51.3	178.2	59.0	33.4	179.0	86.7	7.5	−179.8	−82.4	−17.8	178.5	−135.3	140.1	−4.8
σ		13.1	11.6	3.4	6.2	9.9	3.0	7.5	11.2	3.6	17.3	13.2	5.0	14.6	10.6	5.1	12.4	17.6	1.3

Table 6.1: Continued on next page.

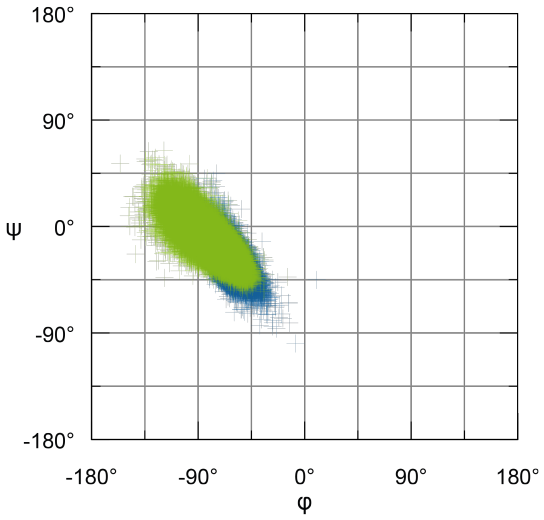
C	|T|	φ_1	ψ_1	ω_1	φ_2	ψ_2	ω_2	φ_3	ψ_3	ω_3	φ_4	ψ_4	ω_4	φ_5	ψ_5	ω_5	φ_6	ψ_6	e
5	33	−68.7	142.6	−179.4	−76.3	−24.4	−177.0	−113.1	−25.0	−179.6	−120.0	155.9	179.9	−81.7	70.6	−177.8	−139.9	148.1	−4.9
σ		7.8	9.7	4.7	12.8	14.7	3.9	18.7	18.0	5.3	27.0	7.8	3.6	5.0	13.5	3.6	17.0	16.6	1.6
6	29	−87.0	−177.8	179.0	−74.2	−15.8	176.9	−98.8	128.8	−178.9	−55.5	136.1	178.2	80.6	1.8	178.7	−113.0	133.9	−3.9
σ		8.2	9.2	4.7	7.8	15.8	4.5	13.6	12.4	4.0	4.0	5.2	2.6	8.9	10.3	3.7	11.2	14.5	1.5
7	24	−83.2	134.9	172.3	−99.8	135.1	9.4	−96.5	−4.7	−172.8	−77.7	−21.8	−179.8	77.3	17.0	177.9	−78.7	146.7	−4.9
σ		17.0	13.7	5.4	17.0	13.2	12.2	8.3	8.1	4.6	11.6	10.5	5.6	14.6	20.2	5.4	12.6	15.6	1.4
8	19	−133.8	145.4	173.4	79.5	178.4	−175.3	−64.4	−27.3	179.2	−95.6	4.5	177.7	54.2	53.2	177.3	−78.7	135.9	−2.8
σ		11.7	10.5	6.3	7.6	11.2	5.6	7.8	9.7	3.4	11.6	10.0	4.8	7.6	9.9	3.5	15.3	10.6	0.9
9	15	−122.7	144.2	−179.9	−92.2	−36.0	−178.9	−135.2	103.9	−4.5	−74.1	164.4	−178.1	−72.0	113.6	179.4	−144.0	146.4	−3.6
σ		31.2	10.5	5.0	11.3	16.9	2.6	7.3	10.2	4.9	5.8	8.5	2.8	7.5	11.3	5.2	18.8	17.0	2.1
10	14	−76.4	134.9	177.3	−64.6	145.6	177.8	79.3	−59.0	179.9	−80.2	−24.0	179.2	67.3	22.8	179.2	−76.2	133.6	−5.0
σ		14.6	14.6	4.2	9.0	10.5	5.3	9.3	9.0	5.1	9.6	7.1	6.4	10.4	13.2	2.4	14.3	9.0	1.1
11	14	81.1	1.1	−178.7	−125.4	154.4	177.0	52.3	−125.4	−178.5	−94.5	3.8	176.6	−81.4	155.6	178.8	−68.5	156.3	−3.5
σ		12.2	9.3	4.2	9.8	11.1	3.6	4.8	7.0	2.7	10.2	8.8	4.3	7.9	11.0	2.3	13.8	18.6	1.4
12	13	−142.4	151.1	171.2	103.4	175.6	175.3	−129.6	107.9	−2.7	−72.9	160.9	−175.4	−67.2	135.6	174.4	−123.7	142.9	−3.8
σ		11.7	11.4	3.7	11.4	3.4	4.2	5.6	7.6	4.2	5.2	5.8	2.8	5.2	4.7	5.6	12.0	13.2	0.8
13	11	−99.1	130.8	176.2	−65.3	−24.4	175.8	−78.6	104.4	−178.8	85.5	−4.4	−178.5	55.4	42.4	−176.6	−94.1	137.1	−5.5
σ		21.6	13.1	4.5	6.6	14.9	3.3	13.6	13.2	1.7	12.7	17.0	5.5	14.4	21.2	5.9	24.5	12.8	0.8
14	10	−104.3	−12.6	178.5	−153.4	173.5	−179.1	−74.1	−5.1	176.4	−89.6	−0.9	176.9	83.7	14.3	176.6	−72.1	141.3	−5.6
σ		16.6	18.3	5.0	9.0	11.9	3.8	8.9	9.1	2.5	12.1	13.0	4.0	18.4	14.3	5.9	9.1	11.0	1.5
15	8	−92.2	109.2	−172.5	−58.7	125.5	179.8	76.5	3.8	−179.9	−110.5	−8.6	177.9	64.0	26.9	−174.6	−128.8	145.0	−5.3
σ		19.4	11.1	6.1	6.0	9.4	3.1	15.8	12.3	7.0	13.3	9.4	5.8	11.0	18.9	3.3	24.3	13.1	1.1
16	8	−98.9	−129.9	−177.4	−74.3	−21.9	177.8	−142.8	166.2	−177.3	−65.5	−12.3	179.4	−96.4	1.4	−178.3	−119.7	118.9	−3.6
σ		8.0	20.8	3.9	10.2	15.5	5.2	20.1	11.7	5.1	8.8	12.5	5.1	9.9	8.9	4.0	14.2	20.7	1.9
17	7	−88.5	165.6	178.0	−54.7	139.6	178.6	95.9	−8.2	179.2	61.8	16.1	−177.7	−94.3	−28.2	−179.4	−128.9	153.6	−4.0
σ		10.8	5.9	1.4	6.5	6.2	4.7	17.8	12.2	3.3	6.4	17.5	5.6	26.0	12.2	3.8	18.7	15.3	1.5
18	6	68.7	23.1	179.3	−88.3	−12.7	177.3	−161.8	172.1	−179.5	−61.9	−33.1	179.3	−71.7	−16.2	−179.8	−119.0	23.6	−2.3
σ		8.1	19.3	3.4	17.0	15.8	3.0	18.7	4.3	3.1	8.5	13.4	3.7	9.7	9.0	3.6	9.8	22.4	0.9
19	6	−65.2	146.1	177.8	−74.6	−26.1	−173.5	−108.4	−8.4	179.2	−143.8	178.8	−179.6	−85.1	2.8	−176.6	−72.2	147.4	−5.6
σ		5.3	9.9	2.3	20.3	12.2	4.8	19.6	10.8	4.5	6.3	7.2	6.1	9.2	9.6	1.9	7.4	13.0	0.8

Table 6.1: Parameters of all turn classes. Turn classes (C), the number of turns in each class (|T|), the mean dihedral angles (in degrees) and the energies (e / kcal/mol) for hydrogen-bonded turns and the Cα–Cα distances (d / Å) for open turns, and the corresponding standard deviations (σ) for all *normal*, *open*, and *reverse* turns. The standard deviations were calculated according to [95, 96]. This Table is extracted from [7].

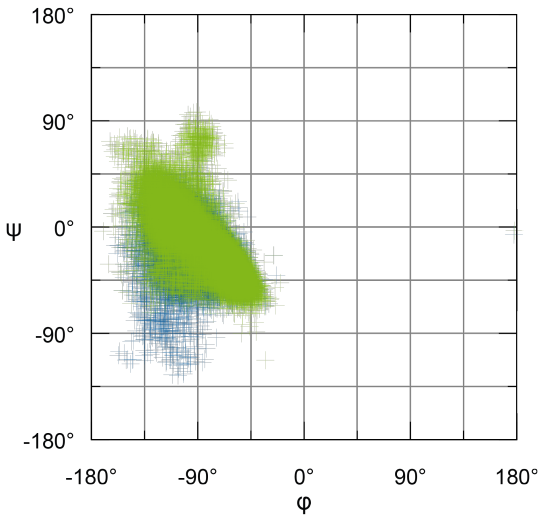
6.1.3 Turn Ramachandran Plots



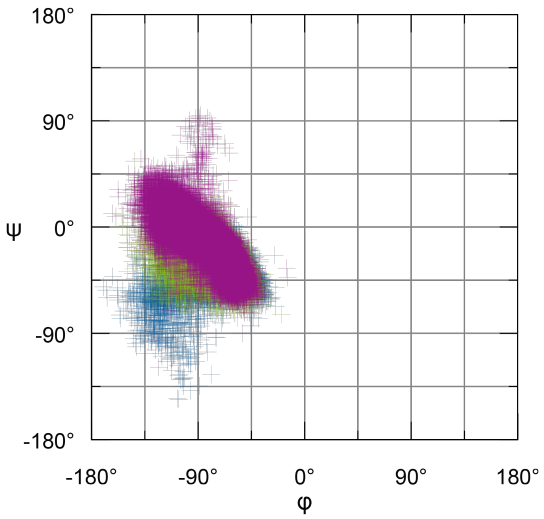
(a) Ramachandran plot for the dihedral angles ϕ_2 and ψ_2 (green), ϕ_3 and ψ_3 (blue), and ϕ_4 and ψ_4 (purple) of the *normal-5 1* turns (X-ray representatives dataset) used for the assignment of right-handed α-helices.



(b) Ramachandran plot for the dihedral angles ϕ_2 and ψ_2 (green) and ϕ_3 and ψ_3 (blue) of the *normal-4 1* turns (X-ray representatives dataset) used for the assignment of right-handed 3_{10}-helices.

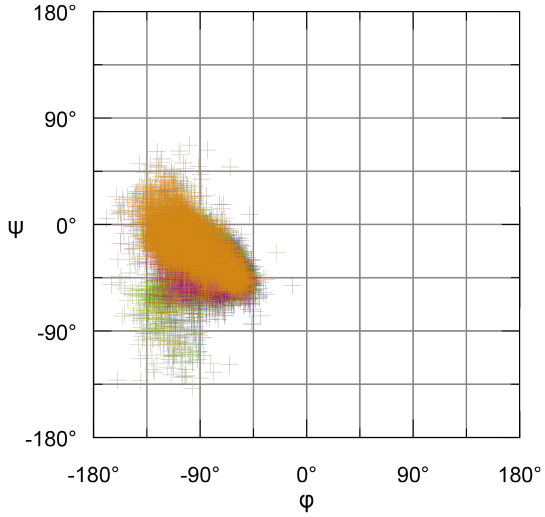


(c) Ramachandran plot for the dihedral angles ϕ_2 and ψ_2 (green) and ϕ_3 and ψ_3 (blue) of the *open-4 2* turns (X-ray representatives dataset) used for the extension of right-handed α- and 3_{10}-helices.

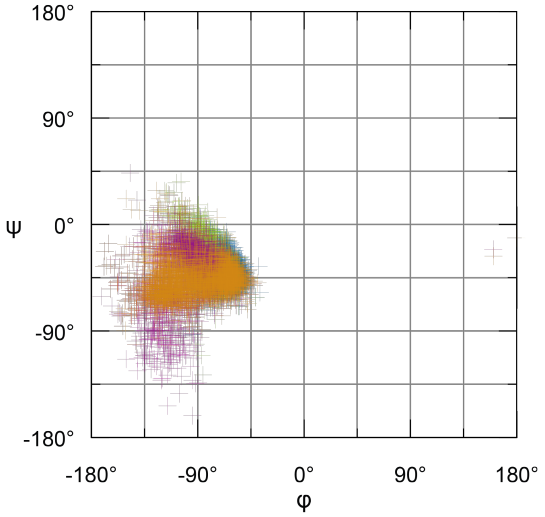


(d) Ramachandran plot for the dihedral angles ϕ_2 and ψ_2 (green), ϕ_3 and ψ_3 (blue), and ϕ_4 and ψ_4 (purple) of the *open-5 1* turns (X-ray representatives dataset) used for the extension of right-handed α- and 3_{10}-helices.

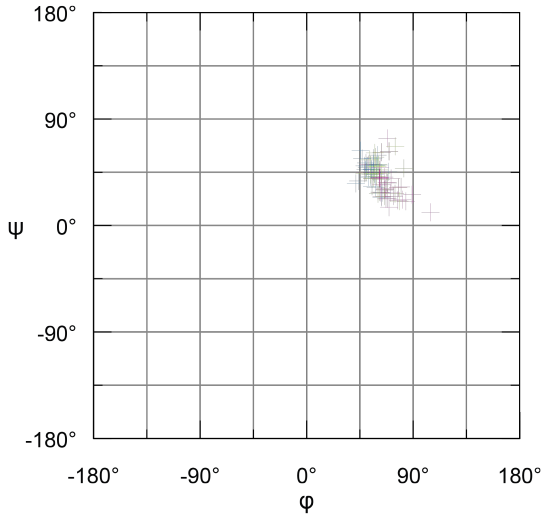
Figure 6.1: Continued on next page.



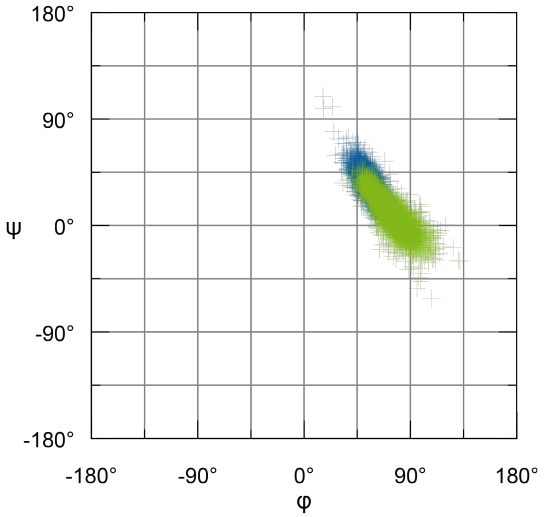
(e) Ramachandran plot for the dihedral angles ϕ_2 and ψ_2 (green), ϕ_3 and ψ_3 (blue), ϕ_4 and ψ_4 (purple), and ϕ_5 and ψ_5 (orange) of the *open-6 4* turns (X-ray representatives dataset) used for the extension of right-handed α- and 3_{10}-helices.



(f) Ramachandran plot for the dihedral angles ϕ_2 and ψ_2 (green), ϕ_3 and ψ_3 (blue), ϕ_4 and ψ_4 (purple), and ϕ_5 and ψ_5 (orange) of the *normal-6 2* turns (X-ray representatives dataset) used for the assignment of right-handed π-helices.

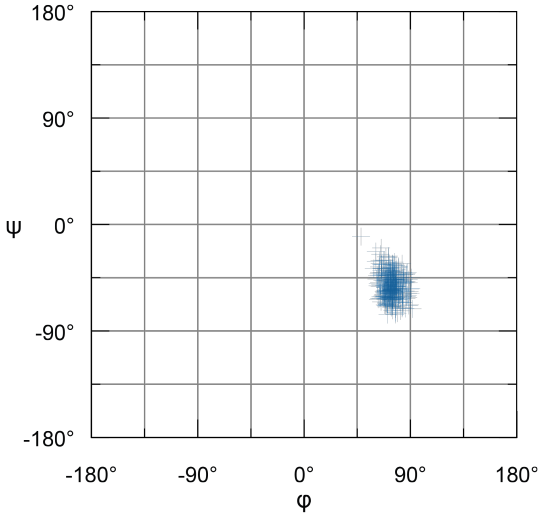


(g) Ramachandran plot for the dihedral angles ϕ_2 and ψ_2 (green), ϕ_3 and ψ_3 (blue), and ϕ_4 and ψ_4 (purple) of the *normal-5 9* turns (X-ray representatives dataset) used for the assignment of left-handed α-helices.

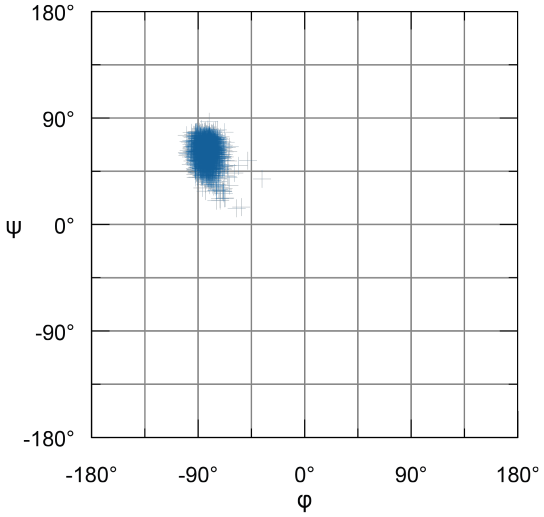


(h) Ramachandran plot for the dihedral angles ϕ_2 and ψ_2 (green) and ϕ_3 and ψ_3 (blue) of the *normal-4 3* turns (X-ray representatives dataset) used for the assignment of left-handed 3_{10}-helices.

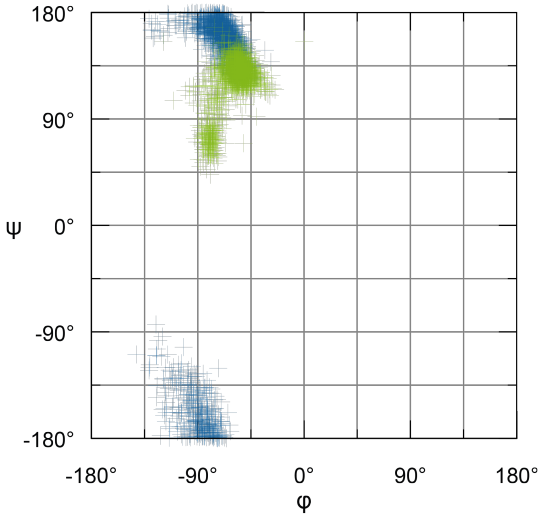
Figure 6.1: Continued on next page.



(i) Ramachandran plot for the dihedral angles ϕ_2 and ψ_2 (blue) of the *normal-3 2* turns (X-ray representatives dataset) used for the assignment of right-handed 2.2_7-helices.



(j) Ramachandran plot for the dihedral angles ϕ_2 and ψ_2 (blue) of the *normal-3 1* turns (X-ray representatives dataset) used for the assignment of left-handed 2.2_7-helices.



(k) Ramachandran plot for the dihedral angles ϕ_2 and ψ_2 (green) and ϕ_3 and ψ_3 (blue) of the *open-4 9* turns (X-ray representatives dataset) used for the assignment of PPII helices.

Figure 6.1: Ramachandran plots for the turn classes used for the SCOT SSE assignments.

6.2 SNOT

6.2.1 Geometry

H1	Distance	Twist	Rise	Radius	Vtor	BDA	φ	ψ	ω	B-factor	Length	Purity
SCOT	5.157	98.900	1.506	2.305	49.744	5.828	−66.025	−37.966	179.513	−0.138	11.731	0.871
σ	0.210	3.732	0.122	0.096	5.897	3.991	11.893	13.627	4.048	0.812	5.335	0.119
SCOT$_{kinked}$	5.155	98.889	1.505	2.306	49.707	5.723	−65.982	−38.049	179.503	−0.139	11.571	0.873
σ	0.206	3.693	0.120	0.095	5.806	3.833	11.811	13.505	4.042	0.811	5.298	0.118
ASSP	5.159	98.853	1.508	2.305	49.719	5.802	−65.708	−38.372	179.495	−0.139	10.787	n/a
σ	0.244	3.378	0.122	0.090	5.543	4.685	11.402	13.834	3.969	0.813	5.399	n/a
DISICL	5.157	98.928	1.509	2.303	49.819	5.768	−66.929	−39.912	179.793	−0.088	5.891	n/a
σ	0.221	3.240	0.116	0.085	5.318	4.202	15.504	38.819	4.514	0.857	5.494	n/a
MKDSSP	5.147	98.874	1.504	2.305	49.607	6.242	−64.270	−40.030	179.452	−0.166	10.660	n/a
σ	0.238	3.356	0.125	0.090	5.718	6.175	8.964	8.718	3.678	0.788	5.612	n/a
PDB	5.204	98.524	1.502	2.312	49.511	6.808	−67.219	−36.738	179.706	−0.103	13.018	n/a
σ	0.287	4.742	0.201	0.131	8.067	7.054	22.215	32.970	6.918	0.847	5.628	n/a
SEGNO	5.166	99.044	1.513	2.300	50.021	6.180	−64.490	−39.312	179.533	−0.127	10.862	n/a
σ	0.262	3.586	0.135	0.095	6.245	5.327	8.277	9.163	3.884	0.815	5.503	n/a
SHAFT	5.245	98.870	1.517	2.303	50.241	8.261	−67.259	−35.125	179.714	−0.081	14.501	n/a
σ	0.381	5.817	0.233	0.157	10.089	9.383	26.620	36.803	7.728	0.865	5.743	n/a
STRIDE	5.180	98.728	1.505	2.308	49.625	6.885	−65.787	−38.136	179.543	−0.136	12.175	n/a
σ	0.290	4.454	0.167	0.118	7.331	8.022	12.117	13.055	5.100	0.817	5.832	n/a

H5	Distance	Twist	Rise	Radius	Vtor	BDA	φ	ψ	ω	B-factor	Length	Purity
SCOT	5.535	105.453	1.726	2.143	62.082	13.578	−71.861	−20.356	−179.501	0.140	3.720	0.918
σ	0.373	7.364	0.236	0.177	12.598	7.847	19.194	17.912	4.542	0.933	1.242	0.143
SCOT$_{kinked}$	5.536	105.617	1.730	2.140	62.316	13.319	−71.783	−20.372	−179.508	0.140	3.694	0.914
σ	0.369	7.274	0.233	0.175	12.459	7.775	19.099	17.908	4.531	0.935	1.179	0.143
ASSP	5.588	105.822	1.750	2.127	63.044	16.069	−69.123	−22.933	179.901	0.089	3.712	n/a
σ	0.501	7.225	0.265	0.185	13.288	9.896	15.834	18.692	4.760	0.946	1.121	n/a
DISICL	6.023	108.369	1.861	2.067	69.077	36.287	−79.250	−12.699	−179.291	0.219	2.457	n/a
σ	0.912	13.551	0.439	0.326	24.000	20.697	28.088	27.831	5.700	0.999	0.841	n/a
MKDSSP	5.853	110.470	1.887	2.023	71.238	28.439	−70.185	−17.555	−179.591	0.163	3.095	n/a
σ	0.613	8.551	0.253	0.185	15.099	21.403	22.919	20.502	4.539	0.957	1.069	n/a
PDB	5.682	105.243	1.580	2.143	60.594	39.790	−74.655	−5.151	−179.603	0.192	4.664	n/a
σ	0.620	10.461	0.800	0.255	26.364	35.991	35.416	56.263	8.884	0.991	1.459	n/a
SEGNO	5.732	108.397	1.834	2.067	67.699	15.971	−70.040	−20.455	−179.593	0.189	3.261	n/a
σ	0.527	7.306	0.236	0.172	13.192	10.817	15.255	14.955	4.952	0.968	0.772	n/a
SHAFT	5.667	104.929	1.725	2.143	62.116	31.409	−77.547	−2.490	−179.431	0.153	5.745	n/a
σ	0.549	9.015	0.417	0.220	16.986	27.564	30.751	59.099	9.225	0.975	1.312	n/a
STRIDE	5.662	107.767	1.735	2.092	65.296	22.902	−68.559	−20.100	−179.683	0.156	3.435	n/a
σ	0.610	9.125	0.508	0.205	19.887	24.448	25.330	21.994	4.536	0.952	0.987	n/a

H3	Distance	Twist	Rise	Radius	Vtor	BDA	φ	ψ	ω	B-factor	Length	Purity
SCOT	5.910	82.679	1.177	2.733	31.280	14.153	−77.914	−43.241	−176.918	−0.266	6.470	0.565
σ	0.450	6.684	0.392	0.253	11.790	7.531	21.099	14.585	4.835	0.661	1.669	0.071
SCOT$_{kinked}$	5.898	82.817	1.179	2.729	31.384	14.074	−77.878	−43.094	−176.901	−0.271	6.224	0.568
σ	0.448	6.654	0.389	0.251	11.718	7.618	20.919	14.566	4.850	0.654	1.688	0.073
ASSP	5.807	84.294	1.226	2.692	33.218	11.744	−79.795	−38.685	−177.368	−0.133	6.041	n/a
σ	0.433	7.542	0.286	0.243	10.691	6.756	20.118	21.292	5.295	0.786	1.560	n/a
DISICL	6.463	75.900	0.977	2.983	23.677	0.000	−88.800	−38.764	−176.677	0.070	2.287	n/a
σ	0.405	7.744	0.556	0.357	15.951	0.000	22.446	13.011	5.910	0.969	0.485	n/a
MKDSSP	5.886	81.413	1.075	2.796	28.178	11.686	−79.446	−43.181	−176.527	−0.246	4.694	n/a
σ	0.366	6.540	0.371	0.264	11.331	8.981	22.185	15.190	4.963	0.687	1.672	n/a
SHAFT	5.844	84.539	1.212	2.681	33.259	15.208	−76.777	−41.404	−177.579	−0.279	8.362	n/a
σ	0.539	8.456	0.425	0.280	14.081	8.334	25.017	24.981	4.846	0.712	2.832	n/a
STRIDE	5.951	81.442	0.943	2.863	26.348	9.831	−82.190	−43.836	−177.251	−0.419	5.139	n/a
σ	0.559	10.788	0.748	0.916	21.430	5.884	34.435	37.196	15.445	0.522	0.833	n/a

Table 6.2: Continued on next page.

H0	Distance	Twist	Rise	Radius	Vtor	BDA	φ	ψ	ω	B-factor	Length	Purity
SCOT	5.358	101.008	1.597	2.249	54.389	13.905	−71.096	−29.496	−179.322	−0.071	5.878	0.495
σ	0.358	6.707	0.237	0.170	11.621	9.498	18.861	18.910	4.675	0.858	1.801	0.025
SCOT$_{kinked}$	5.343	100.991	1.591	2.251	54.207	11.929	−71.238	−29.524	−179.294	−0.025	5.575	0.495
σ	0.342	6.673	0.235	0.170	11.572	8.449	19.066	18.424	4.724	0.904	1.716	0.025
SEGNO	5.226	98.834	1.516	2.305	50.134	7.741	−64.734	−39.392	179.593	−0.207	12.461	n/a
σ	0.367	5.535	0.194	0.150	8.936	7.898	9.406	9.769	3.790	0.752	5.731	n/a

H6	Distance	Twist	Rise	Radius	Vtor	BDA	φ	ψ	ω	B-factor	Length	Purity
SCOT	4.958	95.283	−1.333	2.413	−42.064	n/a	61.887	41.309	179.965	−0.428	4.000	1.000
σ	0.114	1.159	0.074	0.042	2.539	n/a	10.039	11.620	5.271	0.350	0.000	0.000
ASSP	5.224	96.960	−1.456	2.345	−46.871	n/a	64.208	35.759	−179.828	−0.410	4.000	n/a
σ	0.602	4.930	0.319	0.192	12.118	n/a	25.268	31.837	6.034	0.582	0.000	n/a
MKDSSP	5.172	97.581	−1.503	2.321	−48.705	n/a	−88.215	−18.593	−178.909	0.424	1.019	n/a
σ	n/a	n/a	n/a	n/a	n/a	n/a	32.662	29.316	7.178	1.124	0.151	n/a

H11	Distance	Twist	Rise	Radius	Vtor	BDA	φ	ψ	ω	B-factor	Length	Purity
SCOT	5.882	109.986	−1.891	2.032	−70.887	n/a	64.384	24.364	179.905	−0.074	3.139	0.987
σ	0.726	9.331	0.300	0.223	16.516	n/a	15.949	17.004	4.303	0.767	0.483	0.065
ASSP	6.659	123.903	−1.434	1.817	−90.509	n/a	65.248	26.326	179.598	0.096	3.270	n/a
σ	1.842	24.965	1.652	0.475	62.390	n/a	46.740	24.991	4.271	1.002	0.871	n/a
MKDSSP	6.002	113.735	−1.953	1.970	−76.179	n/a	−51.146	7.081	179.692	0.287	1.157	n/a
σ	0.976	10.575	0.357	0.263	19.488	n/a	67.381	51.097	5.452	1.072	0.429	n/a

H6+11+13	Distance	Twist	Rise	Radius	Vtor	BDA	φ	ψ	ω	B-factor	Length	Purity
SCOT	5.687	106.874	−1.773	2.113	−64.737	n/a	64.269	25.135	179.908	−0.090	3.170	0.988
σ	0.750	10.235	0.354	0.254	18.762	n/a	15.739	17.153	4.351	0.757	0.500	0.064
ASSP	5.997	108.366	−1.404	2.124	−56.804	n/a	64.799	28.672	179.665	−0.018	3.435	n/a
σ	1.503	23.551	1.165	0.497	48.582	n/a	44.422	29.441	4.678	0.938	0.860	n/a
DISICL	5.608	106.141	−1.738	2.132	−62.971	n/a	54.554	6.421	−179.992	0.375	2.034	n/a
σ	0.759	9.990	0.357	0.248	18.639	n/a	53.743	56.001	7.011	1.117	0.207	n/a
MKDSSP	5.405	103.614	−1.632	2.202	−57.694	n/a	−15.732	11.527	−179.456	0.096	1.488	n/a
σ	0.528	8.872	0.303	0.219	16.184	n/a	69.276	37.265	5.173	1.221	1.152	n/a

H10	Distance	Twist	Rise	Radius	Vtor	BDA	φ	ψ	ω	B-factor	Length	Purity
SCOT	9.045	121.934	−2.751	1.375	−111.013	41.761	−78.858	142.983	177.964	0.309	4.194	0.995
σ	0.544	20.254	0.995	0.312	29.321	35.770	22.318	23.363	5.119	1.038	1.468	0.033
SCOT$_{no\text{-}strands}$	9.247	130.532	−2.802	1.280	−122.345	43.296	−87.498	139.118	177.998	0.039	4.502	0.996
σ	0.567	22.967	0.934	0.330	31.016	35.962	24.587	22.339	5.417	0.953	1.534	0.028
ASSP	8.987	122.598	−2.966	1.352	−110.661	13.667	−77.186	141.743	177.741	0.195	3.339	n/a
σ	0.390	13.495	0.140	0.175	16.886	7.742	21.679	18.358	5.691	0.974	0.703	n/a
DISICL	8.900	122.360	−2.360	1.423	−111.472	20.406	−83.337	137.234	178.329	0.114	2.266	n/a
σ	0.519	23.024	1.575	0.330	34.237	9.576	34.561	49.021	11.221	0.975	0.607	n/a
SEGNO	9.132	123.576	−2.905	1.333	−112.802	29.297	−77.364	144.751	177.871	0.286	3.568	n/a
σ	0.555	19.248	0.604	0.293	25.149	21.433	20.976	16.259	10.350	1.034	1.033	n/a

S0	Distance	Twist	Rise	Radius	Vtor	BDA	φ	ψ	ω	B-factor	Length	Purity
SCOT	9.840	153.916	−1.797	1.090	−161.130	23.809	−117.052	136.308	178.194	−0.396	5.407	1.000
σ	0.726	25.890	2.213	1.045	36.087	18.380	26.393	26.790	7.644	0.635	2.751	0.000
SCOT$_{kinked}$	10.003	157.491	−2.019	0.954	−161.496	18.639	−117.071	136.328	178.193	−0.396	4.739	1.000
σ	0.388	17.883	2.133	0.232	25.159	12.010	26.289	26.604	7.559	0.634	2.295	0.000
ASSP	9.951	160.116	−2.659	0.962	−159.115	14.159	−108.679	133.835	178.388	−0.279	4.335	n/a
σ	0.368	11.904	1.379	0.123	15.383	7.769	28.375	31.371	7.454	0.762	1.666	n/a
DISICL	9.996	157.095	−2.047	0.957	−160.867	18.761	−111.008	138.845	178.177	−0.206	4.176	n/a
σ	0.415	16.748	2.110	0.773	24.259	12.139	28.199	18.260	10.014	0.813	2.330	n/a
MKDSSP	9.822	153.984	−1.781	1.093	−161.467	23.633	−116.694	136.444	178.233	−0.374	5.407	n/a
σ	0.765	25.811	2.216	1.182	36.295	18.623	27.317	27.971	7.576	0.657	2.670	n/a
PDB	9.818	153.721	−1.769	1.096	−161.358	23.970	−116.786	136.503	178.224	−0.379	5.560	n/a
σ	0.770	25.948	2.224	1.227	36.571	18.851	27.471	28.322	8.061	0.655	2.811	n/a
SEGNO	9.951	155.047	−2.057	0.970	−159.113	19.932	−113.191	137.741	178.244	−0.266	5.712	n/a
σ	0.437	18.678	2.109	0.332	27.160	13.019	28.521	21.211	8.695	0.769	2.357	n/a
STRIDE	9.816	153.723	−1.784	1.092	−161.127	23.936	−116.200	136.708	178.224	−0.368	5.422	n/a
σ	0.766	25.773	2.215	1.120	36.352	18.714	27.733	28.240	7.685	0.662	2.732	n/a

Table 6.2: Parameters of all SSEs on the X-ray representatives dataset. The mean geometric parameters, the dihedral angles, the scaled B-factors, the lengths, and the Purities of right-handed α-, 3_{10}-, π-, and mixed helices, left-handed α- and 3_{10}-helices, left-handed helices, PPII helices, and β-strands assigned by the SSAMs on the X-ray representatives dataset. The Twist, the Vtor, the BDA, φ, ψ, and ω are given in degrees. Their standard deviations were calculated according to [95, 96]. The Distance, the Rise, and the Radius are given in Å. SCOT$_{no\text{-}strands}$ is the SCOT PPII classification without the interference by β-strands.

H6	Distance	Twist	Rise	Radius	Vtor	BDA	φ	ψ	ω	B-factor	Length	Purity
SCOT	5.087	98.779	−1.460	2.322	−48.444	n/a	61.339	40.202	−179.757	−0.443	4.861	0.854
σ	0.177	4.435	0.184	0.128	8.193	n/a	11.616	15.810	4.296	1.260	2.193	0.146
ASSP	5.349	101.834	−1.594	2.235	−54.959	n/a	61.542	33.727	179.984	−0.273	4.146	n/a
σ	0.522	7.491	0.297	0.203	14.523	n/a	16.200	18.958	5.169	1.031	0.794	n/a
MKDSSP	5.172	99.534	−1.496	2.300	−50.148	n/a	−44.279	5.082	−178.842	0.106	1.431	n/a
σ	0.288	6.033	0.220	0.162	10.825	n/a	66.854	38.453	5.096	1.255	1.263	n/a

H11	Distance	Twist	Rise	Radius	Vtor	BDA	φ	ψ	ω	B-factor	Length	Purity
SCOT	5.752	107.244	−1.817	2.090	−66.093	n/a	62.198	26.128	179.001	0.008	4.169	0.968
σ	0.673	8.984	0.290	0.217	16.259	n/a	20.079	21.917	5.859	0.982	0.677	0.096
ASSP	5.764	110.871	−1.882	2.018	−71.291	n/a	60.544	27.423	179.862	−0.215	3.568	n/a
σ	0.385	5.119	0.152	0.106	8.877	n/a	17.812	14.628	6.602	0.715	0.689	n/a
MKDSSP	5.970	113.543	−1.963	1.966	−76.302	n/a	26.609	20.884	179.748	0.091	1.598	n/a
σ	0.554	6.590	0.204	0.151	11.888	n/a	65.747	33.601	5.202	1.191	0.996	n/a

H6+11+13	Distance	Twist	Rise	Radius	Vtor	BDA	φ	ψ	ω	B-factor	Length	Purity
SCOT	5.468	103.613	−1.665	2.189	−58.437	n/a	61.886	31.205	179.440	−0.151	4.389	0.932
σ	0.616	8.489	0.306	0.217	15.952	n/a	17.569	21.022	5.392	1.108	1.385	0.126
ASSP	5.349	101.834	−1.594	2.235	−54.959	n/a	61.542	33.727	179.984	−0.273	4.146	n/a
σ	0.522	7.491	0.297	0.203	14.523	n/a	16.200	18.958	5.169	1.031	0.794	n/a
DISICL	5.455	103.186	−1.645	2.202	−57.560	n/a	62.588	30.741	179.556	−0.161	4.197	n/a
σ	0.645	8.864	0.329	0.228	16.836	n/a	14.156	19.595	5.347	1.032	0.797	n/a
MKDSSP	5.405	103.614	−1.632	2.202	−57.694	n/a	−15.732	11.527	−179.456	0.096	1.488	n/a
σ	0.528	8.872	0.303	0.219	16.184	n/a	69.276	37.265	5.173	1.221	1.152	n/a

Table 6.3: Parameters of left-handed helices on the non-redundant set of structures with left-handed helices dataset. The mean geometric parameters, the dihedral angles, the scaled B-factors, the lengths, and the Purities of left-handed α- and 3_{10}-helices and left-handed helices assigned by the SSAMs on the non-redundant set of structures with left-handed helices dataset. The Twist, the Vtor, the BDA, φ, ψ, and ω are given in degrees. Their standard deviations were calculated according to [95, 96]. The Distance, the Rise, and the Radius are given in Å.

6.2.2 Residues

	H0 (mixed)		H1 (α)		H3 (π)		H5 (3_{10})	
	+	-	+	-	+	-	+	-
SCOT	L	G	ARQEILKMX	NDCGHPSTV	ILFYV	AGPS	DQELKFPSWY	NGHIMTV
ASSP	n/a	n/a	ARQEILKMX	NDCGHPSTYV	LFY	AGPS	AQELKFSWY	NGHITV
DISICL	n/a	n/a	ARDQELK	NCGHFPSTYVX	EIKYV	AGP	ARNDQEKFS	GILMTV
MKDSSP	n/a	n/a	ARQEILKMWX	NDCGHPSTYV	EILMFYV	AGPS	DQEKPSW	GIMTV
PDB	n/a	n/a	ARQELKM	NDCGHFPSTYV	n/a	n/a	NDEFPSW	AGIMTV
SEGNO	ARQEILKM	NDGHPSTVX	ARQEILKMW	NDCGHPSTYVX	n/a	n/a	DQEPSW	GIMTVX
SHAFT	n/a	n/a	ARQELKM	NDCGHIFPSTYV	Y	-	NDEFPSWY	AGIMTV
STRIDE	n/a	n/a	ARQEILKMW	NDCGHPSTVX	-	-	DQEKPSWX	CGHIMTV

Table 6.4: Over- and underrepresented residues in right-handed helices. Over- and underrepresented residues in right-handed helices as assigned by different SSAMs for the X-ray representatives dataset. This Table is extracted from [7].

	H6 (α)		H11 (3_{10})		HL	
	+	-	+	-	+	-
SCOT	ANGY	L	GW	LTV	NGSW	RLKTV
ASSP	ANGSWY	L	GW	-	NGSWY	RLKTV
DISICL	n/a	n/a	n/a	n/a	NGSW	RLKTV
MKDSSP	STY	RD	GY	V	NGSTY	RPV

Table 6.5: Over- and underrepresented residues in left-handed helices. Over- and underrepresented residues in left-handed helices as assigned by different SSAMs for the non-redundant set of structures with left-handed helices. HL are left-handed helices in general. This Table is extracted from [7].

	H10 (PPII)		S0 (β)	
	+	-	+	-
SCOT	P	NDCGHIFSWY	CILFTWYV	ARNDQEGHKMPS
ASSP	KP	NDCGHFSWY	CILFTWYV	ARNDQEGHKMPS
DISICL	DPST	AQEGHILMFWYVX	CILFPTWYV	ARNDQEGHKMSX
MKDSSP	n/a	n/a	CILFTWYV	ARNDQEGHKMPS
PDB	n/a	n/a	CILFTWYV	ARNDQEGHKPS
SEGNO	KP	ANDEGHIFWYVX	CILFTWYV	ARNDQEGHKMPSX
SHAFT	n/a	n/a	n/a	n/a
STRIDE	n/a	n/a	CILFTWYV	ARNDQEGHKPSX

Table 6.6: Over- and underrepresented residues in extended conformations. Over- and underrepresented residues in extended conformations (PPII helices, β-strands) as assigned by different SSAMs for the X-ray representatives dataset. This Table is extracted from [7].

H1							
+	Ncap$_{-3}$	Ncap$_{-2}$	Ncap$_{-1}$	Ncap	Ncap$_{+1}$	Ncap$_{+2}$	Ncap$_{+3}$
SCOT	GPS	GLMP	DGNPST	AELPW	ADEPQ	ADEQT	AFILMRVWY
SCOT$_{kinked}$	GPS	GLMP	DGNPST	AELPW	ADEPQ	ADEQT	AFILMRVWY
ASSP	GPS	GLMP	DGNPST	AEMPW	ADEQ	ADEQ	AFILMQRVWY
DISICL	CFGPWY	DGPSY	DFGLNTWY	CDNPST	ADEKP	ADENQS	ACDEFLQY
MKDSSP	GNPS	GLMP	DGNPST	AELPW	ADEPQ	ADEQ	AFILMRVWY
SEGNO	GPS	GLMP	DGNPST	AEPW	ADEQS	ADEQ	AFILMRVWY
SHAFT	GP	GPSY	FGILMP	DGNPST	EPW	ADEQS	ADEQT
STRIDE	GPS	FGILMP	DGNPST	EPW	ADEQ	ADEQT	AFILMQRVWY

-	Ncap$_{-3}$	Ncap$_{-2}$	Ncap$_{-1}$	Ncap	Ncap$_{+1}$	Ncap$_{+2}$	Ncap$_{+3}$
SCOT	AILRVX	AEHKRX	AEFIKLMQRVWYX	CDGHNSTV	CFGHILMNRTVYX	CFGHIKLNPRSX	DEGHKNPSTX
SCOT$_{kinked}$	ILRVX	AEHKRX	AEFIKLMQRVWYX	CDGHNST	CFGHILMNRTVYX	CFGHILNPRSX	DEGHKNPSTX
ASSP	AILRVX	AEHKRX	AEFIKLMQRVWYX	CDGHINSTVX	CFGHILMNPTVYX	CGHIKNPRSX	DEGHNPSTX
DISICL	ADEHX	AEHKLQRX	AEHKQRSX	AEFGIKLMQRVWYX	CFGHILMNQTVYX	CFGHILMPRVX	HKNPRSX
MKDSSP	AILQRVX	AEHKQRX	AEFIKLMQRVWYX	CDGHINSTV	CFGHILMNRTVYX	CGHIKLNPRSX	DEGHKNPSTX
SEGNO	AILVX	AEKQRX	AEFIKLMQRVWYX	CDGHINSTVX	CFGHILMNPTVYX	CGHIKNPRSX	DEGHNPSTX
SHAFT	AELX	AILRVX	AEHKQRX	AEFIKLMQRVWYX	CDGHINSTVYX	CFGHILMNPRTVYX	CFGHIKLNPRSX
STRIDE	AILRVX	AEHKQRX	AEFIKLMQRVWYX	CDGHINSTVX	CFGHILMNPTVYX	CFGHIKLMNPRSWYX	DEGHKNPSTX

+	Ccap$_{-3}$	Ccap$_{-2}$	Ccap$_{-1}$	Ccap	Ccap$_{+1}$	Ccap$_{+2}$	Ccap$_{+3}$
SCOT	ACFILMWY	AEIKLMQR	AEKLQR	AFHKLMNQRTY	GN	DKP	DEKNP
SCOT$_{kinked}$	ACFILMWY	AEIKLMQR	AEKLQR	AFHKLMNQRTY	GN	DKP	DEKNP
ASSP	ACFILMW	AEIKLMQRW	AEKLQR	AFHKLMNQRY	GN	DGKP	DEKNP
DISICL	AFILWY	ACFIKLRWY	ADEKLPR	ADEKLPQRS	DFGHKNQRSY	GNP	DIKP
MKDSSP	AEFILMQWY	ACFLMQRW	AEIKLQR	AEKLMQR	ACFHKLMNQRSY	GKNPQ	DFGIKP
SEGNO	AEFILMQW	AFIKLMR	AEKLQR	AEKLMQRS	FGHNY	DGKNPS	DEKNP
SHAFT	AEIKLMQR	AEKLQR	AFHKLMNQRTY	GN	DKP	DEKNP	IPV
STRIDE	ACFILMW	AEIKLMQRW	AEKLMQR	AFKLMNQRSY	GN	DKP	DEKNPS

-	Ccap$_{-3}$	Ccap$_{-2}$	Ccap$_{-1}$	Ccap	Ccap$_{+1}$	Ccap$_{+2}$	Ccap$_{+3}$
SCOT	DEGHKNPSTX	CDFGHNPSTVX	CDFGHIPTVWYX	DEGIPV	ADEFILMPRTVWYX	AHLMQSTVWYX	ACFGHILMVX
SCOT$_{kinked}$	DEGHKNPSTX	CDFGHNPSTVX	CDFGHIPTVWYX	DEGIPV	ADEFILMPQRSTVWYX	AHLMSTVYX	ACFGHILMVYX
ASSP	DEGHKNPSTX	CDGHNPSTVX	CFGHIPTVWYX	DGIPVWX	ADEFILMPTVWYX	AEFHILMTVWYX	ACGHILMVX
DISICL	DGHKNPSTX	DEGHNPSTX	FGMNTYX	CFGHIMNTVWYX	ILMPVX	ADEFHILMRTVWYX	AEHLMX
MKDSSP	DGHNPSTX	DGHNPSTVX	CDFGHNPSTVYX	DFGHIPTVW	DEGIPVX	ACEFHILMSTVWYX	AHMSTX
SEGNO	DGHKNPSTX	DGHNPSTX	CDFGHNPSTVYX	DFGHINPVWYX	ADEIKPSTVWX	AEFILMVWYX	ACFHILMTVYX
SHAFT	CDFGHNPSTVX	CDFGHIPTVWYX	DGIPVX	AEFIKLMPRSTVWYX	AEFHLMQSTVWYX	ACFGHILMVWYX	AHMQRSX
STRIDE	DGHKNPSTVX	CDGHNPSTX	CDFGHINPTVWYX	DGIPVWX	ADEFILMPTVWYX	AEHMNSTWX	ACFGILMVYX

Table 6.7: Ncap and Ccap residue preferences of right-handed α-helices. Given are the significantly overrepresented (d > 3.3) and underrepresented (d < -3.3) residues in the proximity of and at the N-terminal and the C-terminal residue of right-handed α-helices. This Table is extracted from [7].

H5							
+	**Ncap$_{-3}$**	**Ncap$_{-2}$**	**Ncap$_{-1}$**	**Ncap**	**Ncap$_{+1}$**	**Ncap$_{+2}$**	**Ncap$_{+3}$**
SCOT			FW	FILMWY	I	E	NY
SCOT$_{kinked}$		E	FW	FILMY	FI	E	NY
ASSP	A	F	FY	FLM	IV	ENY	DV
DISICL	FY	CDFMWY	DE	DEKN	CFITVY	CFLWY	GKP
MKDSSP		DE	FWY	FILMWY	EM	ENT	VY
SEGNO	n/a	n/a	n/a	n/a	n/a	n/a	n/a
SHAFT		M			Y	L	E
STRIDE		F				G	

-	**Ncap$_{-3}$**	**Ncap$_{-2}$**	**Ncap$_{-1}$**	**Ncap**	**Ncap$_{+1}$**	**Ncap$_{+2}$**	**Ncap$_{+3}$**
SCOT			G	GP	GP	P	AGP
SCOT$_{kinked}$				GPS	GP	P	AGP
ASSP			P	G	GP	AGP	AGP
DISICL	GPSX	GSX	IVX	FGIPVX	ADGPSX	AEKPQRX	AELVX
MKDSSP		GP	GKP	DGP	GP	AP	AGS
SEGNO	n/a	n/a	n/a	n/a	n/a	n/a	n/a
SHAFT							
STRIDE							

+	**Ccap$_{-3}$**	**Ccap$_{-2}$**	**Ccap$_{-1}$**	**Ccap**	**Ccap$_{+1}$**	**Ccap$_{+2}$**	**Ccap$_{+3}$**
SCOT	EN	VY	FVY	L	P	F	LP
SCOT$_{kinked}$	N		FVY	L	P	F	LP
ASSP	IV	EN	V	FLY	GK	F	P
DISICL	CDM	DEN	EK	FILVY	CFWY	GKP	
MKDSSP	EM	ENT	FVY	ILVY	FL	GP	FL
SEGNO	n/a	n/a	n/a	n/a	n/a	n/a	n/a
SHAFT	E			LM	P		
STRIDE		G				P	

-	**Ccap$_{-3}$**	**Ccap$_{-2}$**	**Ccap$_{-1}$**	**Ccap**	**Ccap$_{+1}$**	**Ccap$_{+2}$**	**Ccap$_{+3}$**
SCOT	P	P	P	P	DE		T
SCOT$_{kinked}$	P	GP	P	P	DE		T
ASSP	AP	AGP	G	PS	IV		
DISICL	GX	IVX	GPX	ADGPSX	AERX	AEILMVYX	X
MKDSSP	GP	AIP	PS	PS	PV	DEILV	T
SEGNO	n/a	n/a	n/a	n/a	n/a	n/a	n/a
SHAFT							
STRIDE							

Table 6.8: Ncap and Ccap residue preferences of right-handed 3_{10}-helices. Given are the significantly overrepresented (d > 3.3) and underrepresented (d < -3.3) residues in the proximity of and at the N-terminal and the C-terminal residue of right-handed 3_{10}-helices. This Table is extracted from [7].

H3							
+	**Ncap$_{-3}$**	**Ncap$_{-2}$**	**Ncap$_{-1}$**	**Ncap**	**Ncap$_{+1}$**	**Ncap$_{+2}$**	**Ncap$_{+3}$**
SCOT	G	FIV	DHNPS	ALPW	ADEPSW	DEFKLNQWY	CFILMVWY
SCOT$_{kinked}$	G	FIV	DHNPS	ALPW	ADEPSW	DEFKLNQY	CFILRVWY
ASSP	G	none	DNPS	ALMPW	ADEKQSW	DEFKLQY	FILVWY
DISICL	AFILVWY	ACFILWY	DNPS	ADEKNPRS	DEHKNQSTY	DGNS	DKP
MKDSSP	G	FIPVY	DGHNPS	APW	ADENQS	DEFHKLNQY	CFILRVWY
SEGNO	GP	FIVWY	DGHNPS	LPW	ADEKS	DEKNQ	CFILVWY
SHAFT	FP	GNP	FIVY	DHNPS	PW	ADESW	DEFLNQY
STRIDE	GN	CFILVY	DHNPS	APW	ADEKQSW	DEFKLNQY	CFILRVWY

-	**Ncap$_{-3}$**	**Ncap$_{-2}$**	**Ncap$_{-1}$**	**Ncap**	**Ncap$_{+1}$**	**Ncap$_{+2}$**	**Ncap$_{+3}$**
SCOT	X	EHRX	AEFIKLMQRVX	DGHNQTV	CFGILMTVYX	GIPTVX	DEGKNPSTX
SCOT$_{kinked}$	X	EHRX	AEFIKLMQRVX	DGHNQTV	CFGILMTVYX	GIPTVX	DEGKNPSTX
ASSP	X	X	AEFIKVX	DGHNTVX	CFGILPVYX	GIPVX	ADENPSTX
DISICL	DGHNPSX	DEGHKNPSTX	AEFILMQVYX	FGHILMTVWY X	GIVX	AEILMPTVX	AEGHMX
MKDSSP	X	DEX	AEFIKLMQRVX	DFHIMNQTVY	CFGILMTVYX	AGIPTVX	ADEHKNPSTX
SEGNO	AX	ADEHQSX	AEFILMQVWY X	DGHKNQTX	FGILMPVYX	GIPTVX	ADEKPSTX
SHAFT	AEX	AX	AEHQRSX	AEFIKLMQVWY X	DGHKNQTVX	FGILMPVYX	GIPTVX
STRIDE	X	AEHRX	AEFIKLMQRVX	DGHIMNQTVY X	FGILMTVYX	GIPTVX	ADEKNPSTX

+	**Ccap$_{-3}$**	**Ccap$_{-2}$**	**Ccap$_{-1}$**	**Ccap**	**Ccap$_{+1}$**	**Ccap$_{+2}$**	**Ccap$_{+3}$**
SCOT	DNPSW	AEPW	DEKQS	DFKLNQWY	CFGILVW	DGKP	KP
SCOT$_{kinked}$	DNPSW	AEPW	ADEKQS	DFKLNQWY	CFGILVW	DGKP	KP
ASSP	DLPSW	AELPW	ADEKQRS	FKLMQY	FGILVWY	DP	DKP
DISICL	ACDFLWY	DNPS	ADEKNPQRS	DFHKLNQRST WY	CGN	DKPT	DFIKPWY
MKDSSP	DHNPS	ALPW	ADENPQS	DEFHKLNQY	CFILMVWY	DGNP	KP
SEGNO	DNPS	LPW	ADEKS	DEFKLNQY	CFGILVWY	DGKPT	DP
SHAFT	ALPW	DEKQRS	DFHKLNQY	CFILVWY	DGNPT	KP	P
STRIDE	DNPS	AELPW	ADEKQS	DEFKLNQY	CFGILVWY	DGP	DKP

-	**Ccap$_{-3}$**	**Ccap$_{-2}$**	**Ccap$_{-1}$**	**Ccap**	**Ccap$_{+1}$**	**Ccap$_{+2}$**	**Ccap$_{+3}$**
SCOT	AEGIKQRVX	CGHINQTVX	FGILMTVYX	AGIPSV	DEHKNPSTX	ACFILMVYX	GIVX
SCOT$_{kinked}$	AEGIKQRVX	CGHINQTVX	FGILMTVYX	AGIPSV	ADEHKNPSTX	ACFILMVYX	CGIVX
ASSP	GIKTVX	DGHNTVX	FGIPTVX	AGIPSVX	ADENPSTX	ALQVX	GSX
DISICL	EGHKQSTVX	FGIMTVYX	CFGHILMTVW YX	GIPVX	AEHIKLMPQRS TVX	ACFGHILMVYX	AGHMQSX
MKDSSP	AEIKLMQRVX	GHIKNQTVX	FGILMTVYX	AGIPTV	DEGKNPX	ACEFILMVYX	AX
SEGNO	AEFILMQRVX	DGHNQTVYX	CFGILMPTVX	GIPTVX	ADEKPSTX	AEFILMVYX	ALX
SHAFT	GHINQTVX	FGILMPTVX	AGIPVX	ADEKNPST	AEFILMVYX	X	AX
STRIDE	AEFIKLMQRVX	DGHINQTVX	FGILMTVYX	AGIPTVX	ADEKPSTX	AEFILMVX	AILMVX

Table 6.9: Ncap and Ccap residue preferences of right-handed π-helices. Given are the significantly overrepresented (d > 3.3) and underrepresented (d < -3.3) residues in the proximity of and at the N-terminal and the C-terminal residue of right-handed π-helices. This Table is extracted from [7].

H1+H3+H5							
+	**Ncap$_{-3}$**	**Ncap$_{-2}$**	**Ncap$_{-1}$**	**Ncap**	**Ncap$_{+1}$**	**Ncap$_{+2}$**	**Ncap$_{+3}$**
SCOT	GP	FGILMP	DGHNPST	AELPW	ADEPQS	ADEQ	ACFILMRVWY
SCOT$_{kinked}$	GP	FGILMP	DGHNPST	AELPW	ADEPQS	ADEQ	ACFILMRVWY
ASSP	GP	GLMP	DGNPST	AELMPW	ADEKQS	ADELQ	AFILMRVWY
DISICL	CFILVWY	ACFILPWY	DGNPST	DENPST	ADEKNPQS	DEGNQS	CDFPQW
MKDSSP	CGNPSY	FGILMP	DGNPST	AELPW	ADEKPQS	ADEQY	ACFILMRVWY
SEGNO	GNPS	FGILMPV	DGNPST	AELPW	ADEKQS	ADEQ	AFILMRVWY
SHAFT	CFGP	GNPSTY	FGILMPV	DGHNPST	AEPW	ADEQS	ADEQ
STRIDE	GNP	FGILMPVW	DGNPST	AELPW	ADEKQS	ADEQ	ACFILMQRVWY

-	**Ncap$_{-3}$**	**Ncap$_{-2}$**	**Ncap$_{-1}$**	**Ncap**	**Ncap$_{+1}$**	**Ncap$_{+2}$**	**Ncap$_{+3}$**
SCOT	AILQVX	AEHKQRX	AEFIKLMQRVWYX	CDGHINQSTV	CFGHILMNRTVYX	CGHINPRSVX	DEGHKNPSTX
SCOT$_{kinked}$	AHILQVX	AEHKRX	AEFIKLMQRVWYX	CDGHNQSTV	CFGHILMNRTVYX	CGHINPRSVX	DEGHKNPSTX
ASSP	HILRVX	AEHKRX	AEFIKLMQRVWYX	CDGHINSTVX	CFGHILMNPTVYX	CGHINPRSVX	DEGHNPSTX
DISICL	DHKNSX	EHKQRSX	AEHIKQVX	FGHILMQRVWYX	CFGHILMVX	CFILMPRTVX	AHMNRSX
MKDSSP	AILQRVX	AEHKQRX	AEFIKLMQRVWYX	CDGHINQSTV	CFGHILMTVYX	CGHINPRSVX	DEGHKNPSTX
SEGNO	AILVX	AEHKQRX	AEFIKLMQRVWYX	CDGHINQSTVX	CFGHILMNPTVYX	GHINPRSVX	DEGHKNPSTX
SHAFT	AEX	AEIKLQRVX	AEHKQRSX	AEFIKLMQRVWYX	CDGHIMNQRSTVYX	CFGHILMNPTVYX	GHINPRSVX
STRIDE	AHILRVX	AEHKQRX	AEFIKLMQRVWYX	CDGHINQSTVX	CFGHILMNPTVYX	CGHINPRSVX	DEGHKNPSTX

+	**Ccap$_{-3}$**	**Ccap$_{-2}$**	**Ccap$_{-1}$**	**Ccap**	**Ccap$_{+1}$**	**Ccap$_{+2}$**	**Ccap$_{+3}$**
SCOT	CFILTVY	CFILVWY	CFILTVWY	CDFITVWY	CDGNST	DGNPS	DEGNPST
SCOT$_{kinked}$	CFGILTVWY	CFILTVWY	CFILTVWY	CDFITVWY	CDGNST	DGNPS	DEGNPST
ASSP	CFGITVY	CFILTVWY	CFILPTVWY	CDFIPTVY	CDGNPST	DGNPST	DGNPST
DISICL	CFGITVY	CFGITVWY	CFILRTVWY	CDINPSTV	DGNP	DEGNPST	DEGNPST
MKDSSP	CFILTVY	CFILVWY	CFILTVWY	CDFISTVWY	CDGNST	DGNPST	DEGNPST
SEGNO	CFILTVWY	CFILTVWY	CFILTVWY	CDINSTV	DGNPS	DEGNST	DEGNPST
SHAFT	n/a	n/a	n/a	n/a	n/a	n/a	n/a
STRIDE	CFILTVY	CFILVWY	CFILTVWY	CDFISTVY	CDGNPST	DGNPST	DEGNPST

-	**Ccap$_{-3}$**	**Ccap$_{-2}$**	**Ccap$_{-1}$**	**Ccap**	**Ccap$_{+1}$**	**Ccap$_{+2}$**	**Ccap$_{+3}$**
SCOT	ADEHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQSX	AEGKMNPQR	AEFIKLMPQRVWYX	AFHILMQRVWYX	AFILMRVYX
SCOT$_{kinked}$	ADEHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQRSX	AEGKMNPQR	AEFIKLMPQRVYX	AFHIKLMQRVWYX	AFILMRVYX
ASSP	ADEHKNPQRSX	ADEGHKNPQRSX	ADEGHKNQRSX	AEGHKQRX	AHIKLMQRVYX	AFHIKLMQRVWYX	AFHIKLMRVX
DISICL	ADEHKQRSX	ADEHKNQRSX	ADEGHNPSX	AEFGKLMQRWYX	AEFHIKLMQRVWYX	ACFHIKLMRVYX	AFHILMRVX
MKDSSP	ADEHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQSX	AEGKMQRX	AEFIKLMQRVWYX	AFHIKLMQRVWYX	ACFILMRVYX
SEGNO	ADEHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQSX	AEGKLMPQRX	ACEFHIKLMQRVWYX	AFILMPQRVYX	AFILMRVX
SHAFT	n/a	n/a	n/a	n/a	n/a	n/a	n/a
STRIDE	ADEHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQSX	AEGKMQRX	AEFIKLMQRVWYX	AFHIKLMQRVWYX	ACFILMRVYX

Table 6.10: Ncap and Ccap residue preferences of right-handed helices. Given are the significantly overrepresented (d > 3.3) and underrepresented (d < -3.3) residues in the proximity of and at the N-terminal and the C-terminal residue of right-handed helices. This Table is extracted from [7].

S0							
+	**Ncap$_{-3}$**	**Ncap$_{-2}$**	**Ncap$_{-1}$**	**Ncap**	**Ncap$_{+1}$**	**Ncap$_{+2}$**	**Ncap$_{+3}$**
SCOT	DGKNPS	DGKNP	DGKNPR	CFIKRTVWY	CFILTVWY	CFILTVWY	CFILTVWY
SCOT$_{kinked}$	DEGKNPQST	DGKNP	DGKNP	CFIKRTVWY	CFILTVWY	CFILTVWY	CFILTVWY
ASSP	DGKNPS	DGKNPS	DGKN	CFIMPTVWY	FILTVWY	FILTVWY	CFILTVY
DISICL	DEGKNPRS	DGNPS	DGKNP	CFIKPRTVWY	CFIPTVWY	CFILPTVWY	CFIPTVY
MKDSSP	DEGKNPS	DGKNP	DGKNP	CFIKRTVWY	CFILTVWY	CFILTVWY	CFILTVWY
SEGNO	DEGKNPRS	DGNPS	DGKN	FGKPVY	CFIRTVWY	CFILTVWY	CFILTVY
SHAFT	n/a	n/a	n/a	n/a	n/a	n/a	n/a
STRIDE	DEGKNPST	DGKNP	DGKNP	CFIKRTVWY	CFILTVWY	CFILTVWY	CFILTVWY

-	**Ncap$_{-3}$**	**Ncap$_{-2}$**	**Ncap$_{-1}$**	**Ncap**	**Ncap$_{+1}$**	**Ncap$_{+2}$**	**Ncap$_{+3}$**
SCOT	AFILVWYX	ACFHILMRTVWYX	AEFHILMSVYX	ADEGNPS	ADEGHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQRSX
SCOT$_{kinked}$	ACFILVWYX	AEFHILMQRSTVWYX	AEFHILMQSVYX	ADEGLNPS	ADEGHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQRSX
ASSP	ACFHILVWX	AEFHILMVWYX	AEHILMPQSVWX	ADEGHLNQSX	ADEGHKNQSX	ADEGHKNQRSX	ADEGHKQRX
DISICL	ACFILVWX	AEFHILMRVWYX	ACEFHILMQRSTVWYX	ADEGNSX	ADEGHLMNQX	ADEGHKMNQRSX	AEHKMQRX
MKDSSP	ACFILVWYX	ACFHILMRTVWYX	AEFHILMSVYX	ADEGNPS	ADEGHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQRSX
SEGNO	ACFHILVWYX	FHILMQRVWYX	ACEFHILMPSVWYX	ADELNX	ADEGHNPQSX	ADEGKNPQRSX	ADEGHKNQRSX
SHAFT	n/a	n/a	n/a	n/a	n/a	n/a	n/a
STRIDE	AFILVWYX	ACFHILMRTVWYX	AEFHILMQSTVWYX	ADEGNPSX	ADEGHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQRSX

+	**Ccap$_{-3}$**	**Ccap$_{-2}$**	**Ccap$_{-1}$**	**Ccap**	**Ccap$_{+1}$**	**Ccap$_{+2}$**	**Ccap$_{+3}$**
SCOT	CFILTVY	CFILVWY	CFILTVWY	CDFITVWY	CDGNST	DGNPS	DEGNPST
SCOT$_{kinked}$	CFGILTVWY	CFILTVWY	CFILTVWY	CDFITVWY	CDGNST	DGNPS	DEGNPST
ASSP	CFGITVY	CFILTVWY	CFILPTVWY	CDFIPTVY	CDGNPST	DGNPST	DGNPST
DISICL	CFGITVY	CFGITVWY	CFILRTVWY	CDINPSTV	DGNP	DEGNPST	DEGNPST
MKDSSP	CFILTVY	CFILVWY	CFILTVWY	CDFISTVWY	CDGNST	DGNPST	DEGNPST
SEGNO	CFILTVWY	CFILTVWY	CFILTVWY	CDINSTV	DGNPS	DEGNST	DEGNPST
SHAFT	n/a	n/a	n/a	n/a	n/a	n/a	n/a
STRIDE	CFILTVY	CFILVWY	CFILTVWY	CDFISTVY	CDGNPST	DGNPST	DEGNPST

-	**Ccap$_{-3}$**	**Ccap$_{-2}$**	**Ccap$_{-1}$**	**Ccap**	**Ccap$_{+1}$**	**Ccap$_{+2}$**	**Ccap$_{+3}$**
SCOT	ADEHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQSX	AEGKMNPQR	AEFIKLMPQRVWYX	AFHILMQRVWYX	AFILMRVYX
SCOT$_{kinked}$	ADEHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQRSX	AEGKMNPQR	AEFIKLMPQRVYX	AFHIKLMQRVWYX	AFILMRVYX
ASSP	ADEHKNPQRSX	ADEGHKNPQRSX	ADEGHKNQRSX	AEGHKQRX	AHIKLMQRVYX	AFHIKLMQRVWYX	AFHIKLMRVX
DISICL	ADEHKQRSX	ADEHKNQRSX	ADEGHNPSX	AEFGKLMQRWYX	AEFHIKLMQRVWYX	ACFHIKLMRVYX	AFHILMRVX
MKDSSP	ADEHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQSX	AEGKMQRX	AEFIKLMQRVWYX	AFHIKLMQRVWYX	ACFILMRVYX
SEGNO	ADEHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQSX	AEGKLMPQRX	ACEFHIKLMQRVWYX	AFILMPQRVYX	AFILMRVX
SHAFT	n/a	n/a	n/a	n/a	n/a	n/a	n/a
STRIDE	ADEHKNPQRSX	ADEGHKNPQRSX	ADEGHKNPQSX	AEGKMQRX	AEFIKLMQRVWYX	AFHIKLMQRVWYX	ACFILMRVYX

Table 6.11: Ncap and Ccap residue preferences of β-strands. Given are the significantly overrepresented (d > 3.3) and underrepresented (d < -3.3) residues in the proximity of and at the N-terminal and the C-terminal residue of β-strands. This Table is extracted from [7].

6.2.3 Consensus

H1	SCOT	ASSP	DISICL	MKDSSP	PDB	SHAFT	SEGNO	STRIDE
SCOT	1	0.8683	0.6199	0.8715	0.8407	0.6307	0.8121	0.9188
ASSP		1	0.6086	0.8446	0.7908	0.6117	0.7523	0.8523
DISICL			1	0.6455	0.6373	0.4642	0.6190	0.6314
MKDSSP				1	0.8340	0.6488	0.7682	0.8887
PDB					1	0.5712	0.9089	0.8532
SEGNO						1	0.5434	0.6301
SHAFT							1	0.8050
STRIDE								1

(a) Right-handed α-helices

H5	SCOT	ASSP	DISICL	MKDSSP	PDB	SHAFT	SEGNO	STRIDE
SCOT	1	0.2775	0.1694	0.6079	0.4241	0.4412	0.4007	0.6383
ASSP		1	0.1235	0.2659	0.1884	0.1788	0.1785	0.2455
DISICL			1	0.1733	0.1575	0.1078	0.1415	0.1745
MKDSSP				1	0.5902	0.4250	0.4280	0.6967
PDB					1	0.2940	0.5859	0.4475
SEGNO						1	0.3252	0.4501
SHAFT							1	0.3585
STRIDE								1

(b) Right-handed 3_{10}-helices

H3	SCOT	ASSP	DISICL	MKDSSP	SHAFT	STRIDE
SCOT	1	0.2198	0.0796	0.4432	0.0632	0.0256
ASSP		1	0.0568	0.2634	0.0305	0.0088
DISICL			1	0.1102	0.0073	0.0039
MKDSSP				1	0.0341	0.0247
SHAFT					1	0.0776
STRIDE						1

(c) Right-handed π-helices

H0	SCOT	SEGNO
SCOT	1	0.0028
SEGNO		1

(d) Right-handed mixed helices

H10	SCOT	$SCOT_{wostrands}$	ASSP	DISICL	SEGNO
SCOT	1	0.5555	0.1644	0.0735	0.1687
$SCOT_{wostrands}$		1	0.1457	0.0845	0.1478
ASSP			1	0.0936	0.2021
DISICL				1	0.1708
SEGNO					1

(e) PPII helices

S0	SCOT	ASSP	DISICL	MKDSSP	PDB	SEGNO	STRIDE
SCOT	1	0.4788	0.5158	0.8982	0.8811	0.5916	0.8567
ASSP		1	0.5001	0.4859	0.4893	0.5439	0.4826
DISICL			1	0.535	0.541	0.6565	0.5431
MKDSSP				1	0.9755	0.6065	0.9252
PDB					1	0.613	0.9067
SEGNO						1	0.6048
STRIDE							1

(f) β-strands

Table 6.12: Consensus of different SSAMs for all SSEs except left-handed helices. Consensus of different SSAMs in the assignment of right-handed α, 3_{10}, π-, and mixed helices, PPII helices, and β-strands for the X-ray representatives dataset. The most similar methods to SCOT are highlighted in blue. This Table is extracted from [7].

H6	SCOT	ASSP	MKDSSP
SCOT	1	0.5886	0.1925
ASSP		1	0.2110
MKDSSP			1

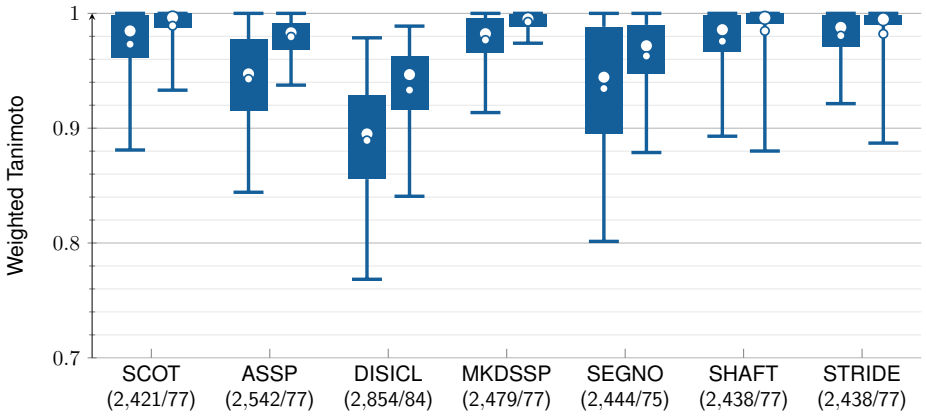
(a) Left-handed α-helices

H11	SCOT	ASSP	MKDSSP
SCOT	1	0.3725	0.3360
ASSP		1	0.1640
MKDSSP			1

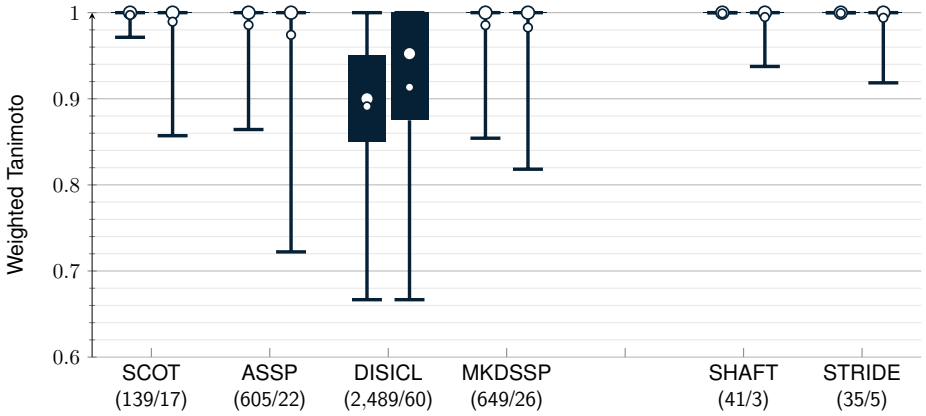
(b) Left-handed 3_{10}-helices

Table 6.13: Consensus of different SSAMs for left-handed helices. Consensus of different SSAMs in the assignment of left-handed α- and 3_{10}-helices for the non-redundant set of structures with left-handed helices. The most similar methods to SCOT are highlighted in blue. This Table is extracted from [7].

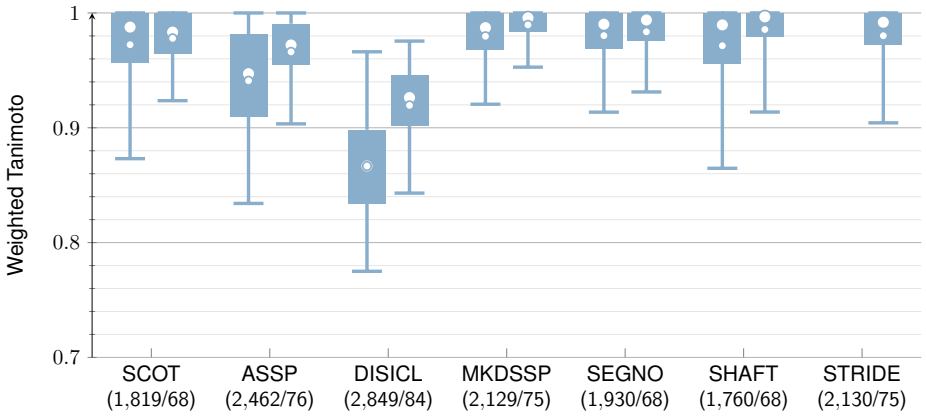
6.2.4 Consistency



(a) Right-handed α-helices (H1)

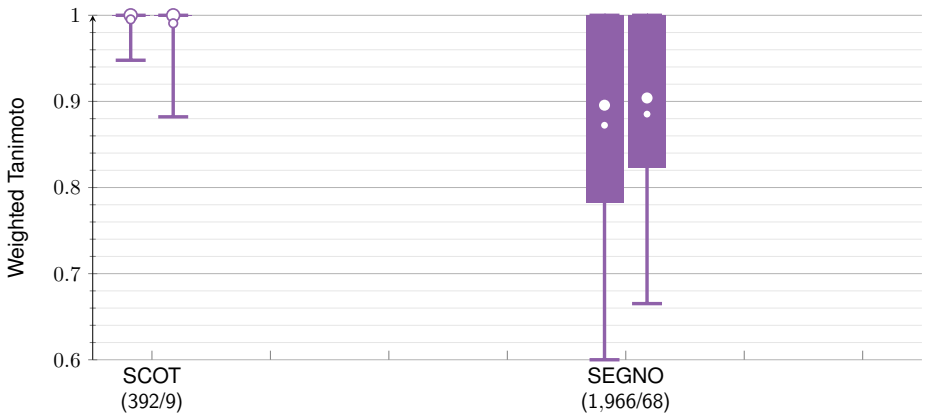


(b) Right-handed π-helices (H3)

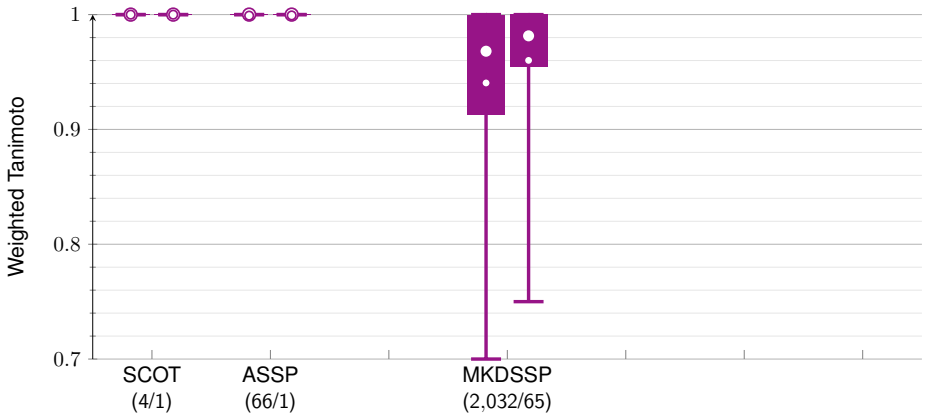


(c) Right-handed 3_{10}-helices (H5)

Figure 6.2: Continued on next page.



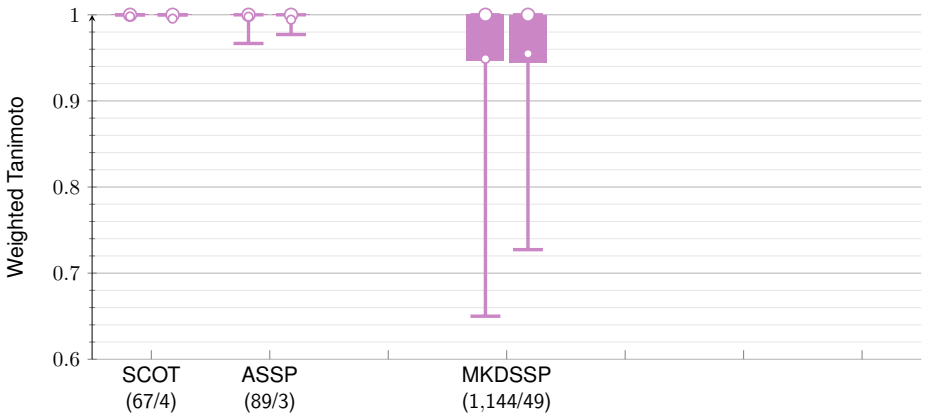
(d) Right-handed mixed helices (H0)



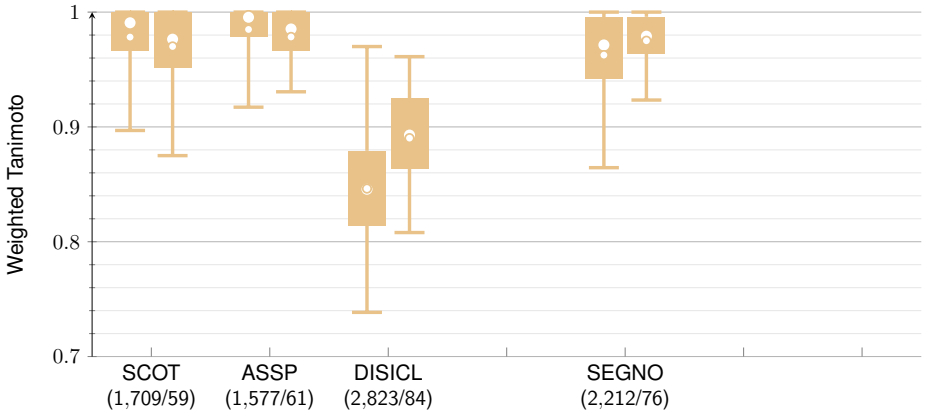
(e) Left-handed α-helices (H6)

(f) Left-handed 2.2_7-helices (H8)

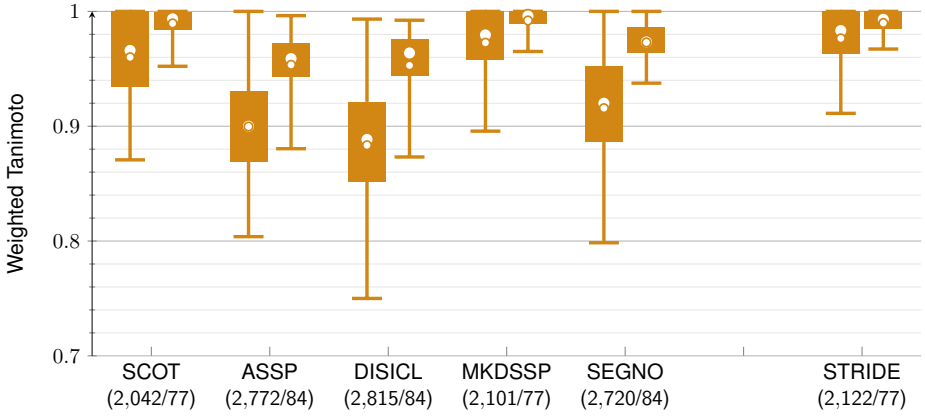
Figure 6.2: Continued on next page.



(g) Left-handed 3_{10}-helices (H11)



(h) PPII helices (H10)



(i) β-strands (S0)

Figure 6.2: Boxplots showing the SSE class-specific consistency of different SSAMs based on the weighted Tanimoto coefficient. Boxplots showing the overall consistency of different SSAMs for the assignment of different SSE classes based on the weighted Tanimoto coefficient for the NMR (left, 2,856 ensembles) and X-ray (right, 84 ensembles) ensembles datasets. The median is indicated by a big and the mean by a small white dot. The numbers of ensembles in which SSEs were classified by each SSAM and for each dataset are given in parentheses. Outliers were omitted in favor of a concise visualization. This Figure is extracted from [7].

6.2.5 Left-Handed Helices

H	PDB-ID	SCOT	Manual[111]	ASSP[65]	DISICL[54]	FLH[198]	MKDSSP[22]	Name	CATH domain	Sequence
3_{10}	1autL	101-104	101-104	101-104	101-104	n.i.	101-103	Activated protein C	2.10.25.10	DNGG
3_{10}	1b9wA	52-55	52-55	n.i.	52-55	n.i.	52-54	Merozoite surface protein 1 (*Plasmodium cynomolgi*)	2.10.25.10	KNGG
3_{10}	1b9wA	n.i.	n.i.	n.i.	83-85	n.i.	83-84	Merozoite surface protein 1 (*Plasmodium cynomolgi*)	2.10.25.10	FEG
3_{10}	1g2lB	258-261	258-261	259-261	258-261	n.i.	258-260	Coagulation factor X	2.10.25.10	DNGD
3_{10}	1h21A	n.i.	n.i.	n.i.	n.i.	n.i.	39-40	Split-soret cytochrome C (*Desulfovibrio desulfuricans*)	–	YK
3_{10}	1jv1A	182-185	182-185	182-185	182-185	182-184	182-184	Glcnac1p uridyltransferase	3.90.550.10	KYFG
3_{10}	1kdgA	n.i.	n.i.	n.i.	n.i.	n.i.	450-451	Cellobiose dehydrogenase (*Phanerochaete chrysosporium*)	3.30.410.10	PN
3_{10}	1kdgA	532-535	532-535	532-535	532-535	532-535	532-534	Cellobiose dehydrogenase (*Phanerochaete chrysosporium*)	3.30.410.10	YENW
3_{10}	1kliL	94-97	94-97	95-97	94-97	94-96	94-96	Coagulation factor VII	2.10.25.10	ENGG
3_{10}	1n1iA	57-60	57-60	57-60	57-60	n.i.	57-59	Merozoite surface protein 1 (*Plasmodium knowlesi*)	2.10.25.10	NNGG
3_{10}	1n1iA	88-90	n.i.	n.i.	88-90	n.i.	88-89	Merozoite surface protein 1 (*Plasmodium knowlesi*)	2.10.25.10	FEG
3_{10}	1ob1C	52-55	52-55	53-55	52-55	n.i.	52-54	Merozoite surface protein 1 (*Plasmodium falciparum*)	2.10.25.10	NNGG
3_{10}	1ob1C	87-89	n.i.	n.i.	87-89	n.i.	87-88	Merozoite surface protein 1 (*Plasmodium falciparum*)	2.10.25.10	FDG
3_{10}	1pb5A	29-32	28-32	29-32	28-32	n.i.	29-31	Lnr module from Notch	–	GWDGG
3_{10}	1rfnB	91-94	91-94	92-94	91-94	n.i.	90-93	Coagulation factor IX	2.10.25.10	KNGR
3_{10}	2gsaA	65-67	65-68	65-67	65-67	65-67	65-67	Glutamate-1-semialdehyde aminotransferase (*Synechococcus sp.*)	3.90.1150.10	GTWG
3_{10}	2gsaA	n.i.	n.i.	n.i.	n.i.	n.i.	24-25	Glutamate-1-semialdehyde aminotransferase (*Synechococcus sp.*)	3.90.1150.10	PG
$\neq$	1h21A	n.i.	77-81 (3_{10})	77-81 (pi)	n.i.	n.i.	77-78 (3_{10})	Split-soret cytochrome C (*Desulfovibrio desulfuricans*)	–	GGISD
$\neq$	1hxxA	n.i.	143-146 (3_{10})	143-146 (α)	143-146	143-145	143-145 (α)	Ompf porin (*Escherichia coli*)	2.40.160.10	NFFG
$\neq$	1j9qA	105-107 (3_{10})	105-108 (3_{10})	105-108 (α)	105-108	n.i.	105-107 (3_{10})	Nitrate reductase (*Alcaligenes faecalis*)	2.60.40.240	ALGG
$\neq$	1nifA	n.i.	105-108 (3_{10})	105-108 (α)	105-108	n.i.	105-107 (3_{10})	Nitrate reductase (*Achromobacter cycloclastes*)	2.60.40.240	ALGG
$\neq$	1oe1A	n.i.	99-102 (3_{10})	99-102 (α)	99-102	n.i.	99-101 (3_{10})	Nitrate reductase (*Alcaligenes xylosoxydans xylosoxydans*)	2.60.40.240	ALGG
$\neq$	1qj5A	n.i.	50-53 (3_{10})	50-53 (α)	n.i.	n.i.	n.i.	7,8-Diaminopelargonic acid synthase (*Escherichia coli*)	3.90.1150.10	SSWW
$\neq$	2oatA	83-86 (3_{10})	83-86 (3_{10})	83-86 (α)	83-86	83-85	83-85 (3_{10})	Ornithine aminotransferase	3.90.1150.10	SSYS
α	1ak0A	n.i.	131-134	131-134	131-134	131-133	130-133	P1 nuclease (*Penicillium citrinum*)	1.10.575.10	AVGG
α	1bd0A	40-44	40-44	40-44	40-44	40-43	40-43	Alanine racemase (*Geobacillus stearothermophilus*)	3.20.20.10	ANAYG
α	1bnlA	n.i.	135-138	135-138	135-138	135-138	135-137	Endostatin	3.10.100.10	CETW
α	1bqbA	223-226	223-226	223-226	223-226	n.i.	223-225	Aureolysin (*Staphylococcus aureus*)	1.10.390.10	DNGG
α	1dy2A	207-210	207-210	207-210	207-210	207-210	207-209	Endostatin (*Mus musculus*)	3.10.100.10	CEAW
α	1hzmA	n.i.	61-64	60-63	61-64	61-64	61-63	Protein phosphatase 6	3.40.250.10	IMLR
α	1kdgA	n.i.	n.i.	n.i.	n.i.	n.i.	552-553	Cellobiose dehydrogenase (*Phanerochaete chrysosporium*)	3.30.410.10	AN
α	1koeA	266-269	266-269	266-269	266-269	266-269	266-268	Endostatin (*Mus musculus*)	3.10.100.10	CETW
α	1kwsA	298-301	298-301	298-301	298-301	298-301	298-300	Beta-1,3-glucuronyltransferase 3	3.90.550.10	AANC
α	1npcA	227-230	227-230	227-230	227-230	227-230	227-229	Neutral protease (*Bacillus cereus*)	1.10.390.10	DNGG
α	1ohvA	70-73	70-73	70-73	70-73	70-73	70-72	4-Aminobutyrate aminotransferase (*Sus scrofa*)	3.90.1150.10	SQIS
α	8tlnE	226-229	226-229	226-229	226-229	226-229	226-228	Thermolysin (*Bacillus thermoproteolyticus*)	1.10.390.10	DNGG
?	1mzrA	n.i.	191-194 (3_{10})	n.i.	n.i.	n.i.	n.i.	2,5-Diketo-D-gluconate reductase (*Escherichia coli*)	3.20.20.10	AQGG
?	1ptmA	211-215 (3_{10})	211-216 (3_{10})	n.i.	211-213	n.i.	211-215 (3_{10})	4-Hydroxythreonine-4-phosphate dehydrogenase (*Escherichia coli*)	3.40.718.10	HAGEGG

Table 6.14: Assignments for left-handed helices by different SSAMs. Assignments of left-handed helices by different SSAMs for the dataset of Novotny and Kleywegt[111]. Additionally, the Perl script findlefthanded.pl (FLH) was used to define left-handed helices based on the dihedral angles. The assigned helices are grouped by equally, different ($\neq$), and questionable (?) assigned helix classes (H). Human proteins are given without the name of the corresponding organism. This Table is extracted from [7].

6.3 SLOT

6.3.1 Configuration File and Parameters

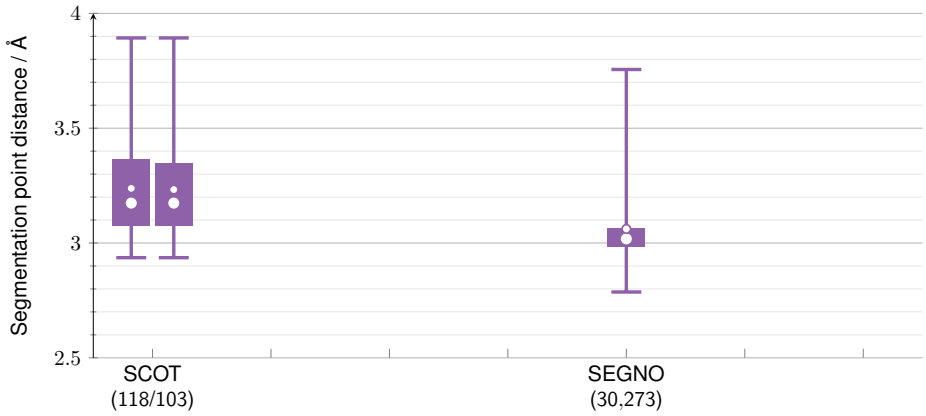
```
dataset1
 name:lsc query
 identifier
  string:1gos 2ejr 2bxr
 protein
  directory:/datasets/lsc/proteins/
   file-extension:.pdb
 pocket
  atom
   directory:/datasets/lsc/pockets/
   file-extension:.pdb
   pocket-number-delimiter:_
   max-number-pockets:3
 collector
  pocket
   helices:y
   sheets:y
   turns:n
 modeler
  min-helix-segmentation-points:2
  min-strand-segmentation-points:2
 model-writer
  pymol
   directory:/results/models/
   file-extension:.py
   vertex-diameter:1
   vertex-color:20 95 153
   edge-diameter:0.2
   edge-color:20 95 153
   segmentation-vertex-color:131 184 26
   segmentation-vertex-diameter:0.5
   segmentation-edge-color:131 184 26
   segmentation-edge-diameter:0.1
```

Figure 6.3: Continued on next page.

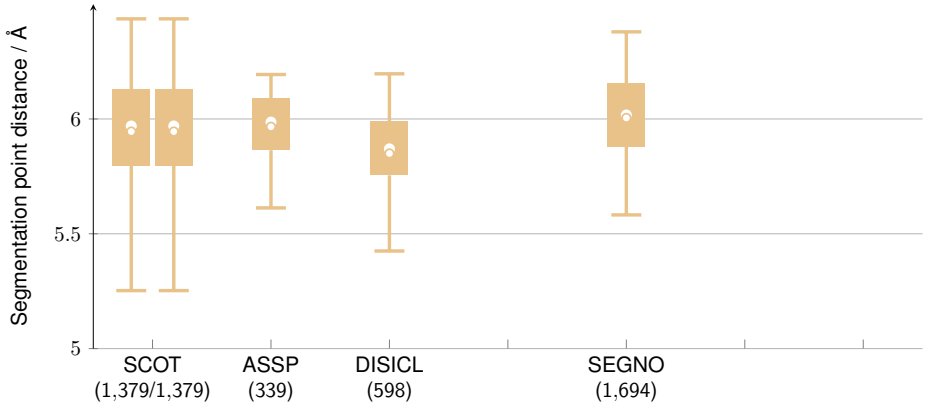
```
dataset2
 name:lsc target
 identifier
  directory:/datasets/lsc/proteins/
  file-extension:.pdb
 protein
  directory:/datasets/lsc/proteins/
 collector
  protein
   helices:y
   sheets:y
   turns:n
 modeler
  min-helix-segmentation-points:2
  min-strand-segmentation-points:2

comparator
 min-match-size-static:0
 min-match-size-dynamic:0.3
 min-match-size-minimum:3
 vertices
  segmentation-points-deviation-static:0
  segmentation-points-deviation-dynamic:0.6
 edges
  outer-distance-deviation-static:1
  outer-distance-deviation-dynamic:0.15
  cross-distance-deviation-static:1
  cross-distance-deviation-dynamic:0.15
  angle-deviation-static:25
  angle-temperature-limit:1
  angle-temperature-factor:5
  displacement-deviation-static:1
  displacement-deviation-dynamic:0.25
  min-segmentation-points-match-size:2

match-writer
 pymol
  directory:/results/matches/pymol/
  file-extension:.py
  protein-color1:255 255 255
  protein-color2:128 128 128
  protein-transparency:0.6

judge
 file:/results/scores.txt
```

Figure 6.3: Example of a configuration file shown. The example is given in a two column representation due to page length limitations. Almost all possible parameters are given for a comparison of protein pockets of the LSD1 dataset (see Section 2.3.8.1) to a target dataset of entire proteins. Colors have to be given in RGB and an integer value in [0, 255] for each color channel.

6.3.2 Segmentation Point Distances



(a) Right-handed mixed helices (H0)



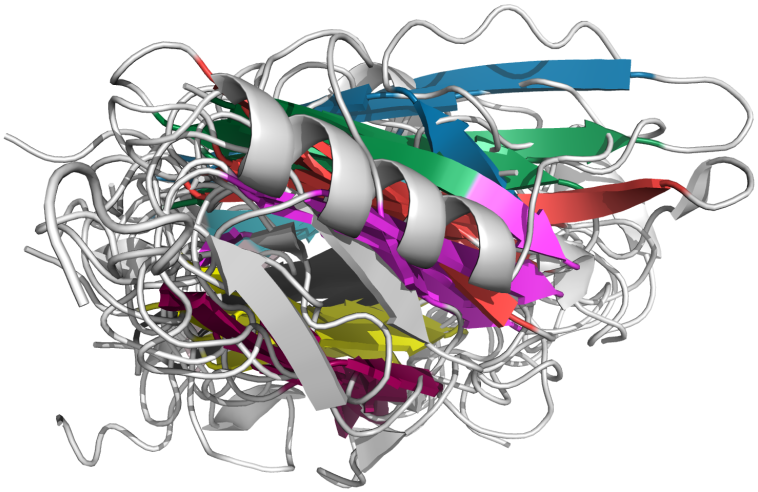
(b) PPII helices (H10)

Figure 6.4: Boxplots showing the segmentation point distances in right-handed mixed and PPII helices for different SSAMs. Boxplots showing the segmentation point distances between two neighboring points in right-handed mixed and PPII-helices obtained for the X-ray representatives dataset for different SSAMs. For SCOT, these distances are given for the standard settings (left) and for the split helices at kink positions (right). The numbers of analyzed distances are given in parentheses. The median is indicated by a big and the mean by a small white dot. Outliers were omitted in favor of a concise visualization.

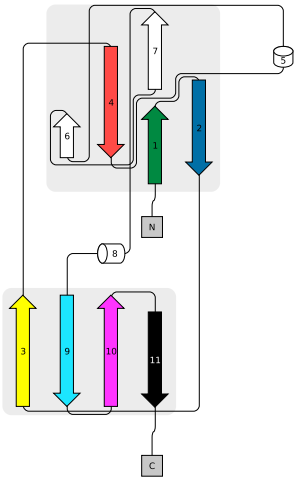
6.3.3 Hunting for Domain Pairs

PDB-ID	CATH-ID	SLOT		LOCK2		DaliLite	TM-align
		SCOT	MKDSSP	SCOT	MKDSSP		
1nc7A00	2.60.290.11	0.8000	0.6000	159.46	113.69	0.00	0.3548
3ld7A00	2.60.320.10	0.7778	0.8750	128.76	129.66	2.40	0.3361
1f00l02	2.60.40.10	0.7778	0.6000	102.98	109.13	0.00	0.3186
4unuA00	2.60.40.10	0.7778	0.7778	109.81	110.85	0.00	0.3066
4iauA02	2.60.20.10	0.7778	0.8750	104.82	107.67	0.00	0.2860
4a0tA03	2.60.320.30	0.9000	0.8750	126.61	139.97	0.00	0.2831
1rl6A02	3.90.930.1	0.7778	0.7500	82.88	107.25	0.00	0.2408

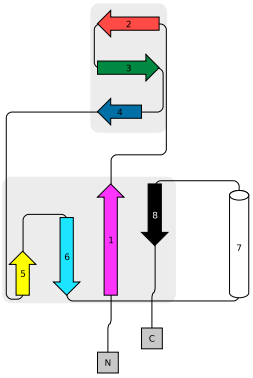
Table 6.15: List of domain pairs with high scores obtained by SLOT using SCOT-based SSE annotations for the query domain 4f01B01@cath (2.60.34.10), for which low scores were obtained by all other SSCMs. The attribution of high and low is based on the boxplots shown in Figure 5.16. A topologically distant domain with respect to the query is highlighted in blue and the corresponding topology diagrams are presented in Figure 6.5.



(a)



(b) Topology diagram of 4f01B01@cath.



(c) Topology diagram of 1rl6A02@cath.

Figure 6.5: Superposition and topology diagrams for the query domain 4f01B01@cath and a topologically distant domain. (a) Superposition according to SLOT with SCOT-based SSE annotations of the query domain 4f01B01@cath and the domains listed in Table 6.15. The topology diagrams are given for the query (b) and a high scored (matched), but topologically distant domain (c). The diagrams were created using Pro-origami [185]. Matched SSEs are highlighted in the same color in all figures.

6.3.4 Searching for Ligand-Sensing Cores

Rank	Query		Target					Matching	
	PDB-ID	SSEs	PDB-ID	EC No	UniProt	ChEMBL	SSEs	SSEs	Score
1	1fj2A	14	1fj2A	3.1.4.39	O75608		14	14	1.000
2	1fj2A	14	1fj2B	3.1.4.39	O75608		14	14	1.000
3	1fj2A	14	4h0cA		C6VYE5		12	11	0.7857
4	1fj2A	14	3trdA		Q83AV9		14	11	0.7857
5	1fj2A	14	1jjfA	3.2.1.8	P10478		14	11	0.7857
6	1fj2A	14	1r1dA	3.1.1.1	Q06174		15	11	0.7333
7	1fj2A	14	1zk4A	1.1.1.2	Q84EX5		15	11	0.7333
8	1fj2A	14	2qruA		Q838Q5		15	11	0.7333
9	1fj2A	14	3f67A	3.1.1.-	A6TGL0		15	11	0.7333
10	1fj2A	14	4le1A		O34723		11	10	0.7143
11	1fj2A	14	4iqgA		Q12GY8		13	10	0.7143
12	1fj2A	14	1i6wA	3.1.1.3	P37957		12	10	0.7143
13	1fj2A	14	2plwA	2.1.1.-	Q8IEL9		14	10	0.7143
14	1fj2A	14	2i3dA		A9CIK7		14	10	0.7143
15	1fj2A	14	3wynA		F7IX06		16	11	0.6875
16	1fj2A	14	4nbrA		Q577I6		16	11	0.6875
17	1fj2A	14	3dkrA		B2CZF3		16	11	0.6875
18	1fj2A	14	1yxmA	1.3.1.8	Q9BY49		15	10	0.6667
19	1fj2A	14	4nimA		C0RJU5		15	10	0.6667
20	1fj2A	14	3is3A	1.1.1.62	O93874		15	10	0.6667
21	1fj2A	14	4gkbA	1.1.1.100	A0A0H3KNE7		15	10	0.6667
22	1fj2A	14	1ooeA	1.6.99.7	Q9XVJ3		15	10	0.6667
23	1fj2A	14	4n5iX		C7TF14		18	12	0.6667
24	1fj2A	14	2qm0A		Q81A57		18	12	0.6667
25	1fj2A	14	2fhpA		Q831P8		15	10	0.6667
26	1fj2A	14	3e0xA		Q97KV0		17	11	0.6471
27	1fj2A	14	3g9tA	3.1.1.-	Q0GMU2		17	11	0.6471
28	1fj2A	14	1zemA	1.1.1.9	Q8GR61		16	10	0.6250
29	1fj2A	14	4nk4A	1.3.1.9	M4Q2P0		16	10	0.6250
30	1fj2A	14	3sx2A		Q73W00		16	10	0.6250
31	1fj2A	14	1pbtA	3.1.1.31	Q9X0N8		16	10	0.6250
32	1fj2A	14	4lvuA	1.-.-.-	Q2SZC0		16	10	0.6250
33	1fj2A	14	2apjA		Q8L9J9		16	10	0.6250
34	1fj2A	14	1xg5A		Q6UWP2		16	10	0.6250
35	1fj2A	14	4fc7A	1.3.1.34	Q9NUI1		16	10	0.6250
36	1fj2A	14	3iccA		A0A0F7RDR0		16	10	0.6250
37	1fj2A	14	4esoA	1.-.-.-	Q92N93		16	10	0.6250
38	1fj2A	14	4q82A		D0LMJ0		16	10	0.6250
39	1fj2A	14	4mxdA	4.2.99.20	P37355		18	11	0.6111
40	1fj2A	14	2hdwA		Q01609		18	11	0.6111
41	1fj2A	14	3ga7A	3.1.1.-	Q8ZRA1		18	11	0.6111
42	1fj2A	14	3d0kA		Q7W3B7		18	11	0.6111
43	1fj2A	14	1uj2A	2.7.1.48	Q9BZX2		17	10	0.5882
44	1fj2A	14	3awdA		Q5FNX9		17	10	0.5882
45	1fj2A	14	3pk0A		A0QQJ6		17	10	0.5882
46	1fj2A	14	3u49A		P39644		17	10	0.5882
47	1fj2A	14	1yb1A	1.1.1.62	Q8NBQ5		17	10	0.5882
48	1fj2A	14	1j1iA	3.7.1.8	Q84II3		17	10	0.5882
49	1fj2A	14	1zmtA		Q93D82		17	10	0.5882
50	1fj2A	14	3e9qA		A0A0H2UYY2		17	10	0.5882
51	1fj2A	14	3gemA		Q48MN0		17	10	0.5882
52	1fj2A	14	4g9eA		D2J2T6		17	10	0.5882
53	1fj2A	14	1l7aA	3.1.1.41	P94388		19	11	0.5789
54	1fj2A	14	1q0rA		Q54528		19	11	0.5789
55	1fj2A	14	4htaA		Q9SZU7		19	11	0.5789
56	1fj2A	14	1mtzA	3.4.11.5	P96084		19	11	0.5789
57	1fj2A	14	4j7aA		Q4TZQ3		19	11	0.5789
58	1fj2A	14	2o7rA	3.1.1.1	Q0ZPV7		21	12	0.5714

Table 6.16: Continued on next page.

Rank	Query		Target					Matching	
	PDB-ID	SSEs	PDB-ID	EC No	UniProt	ChEMBL	SSEs	SSEs	Score
59	1fj2A	14	4mqmA	2.3.1.122 (2.3.1.20)	P9WQN9	●	18	10	0.5556
60	1fj2A	14	1zoiA	3.1.-.-	Q3HWU8		18	10	0.5556
61	1fj2A	14	3berA	3.6.1.-	Q9H0S4		18	10	0.5556
62	1fj2A	14	2r11A	3.1.1.1	P96688		20	11	0.5500
63	1fj2A	14	3wwcA	3.7.1.7	Q9LCQ7		19	10	0.5263
64	1fj2A	14	2gk4A		A0A0H2UQ78		19	10	0.5263
65	1fj2A	14	4g4iA	3.1.1.-	G2QJR6		19	10	0.5263
66	1fj2A	14	3thrA		P13255		19	10	0.5263
67	1fj2A	14	1uk8A	3.7.1.9	P96965		19	10	0.5263
68	1fj2A	14	1t64A		Q9BY41	●	19	10	0.5263
69	1fj2A	14	3rd5A		Q741V7		19	10	0.5263
70	1fj2A	14	3lcrA	3.1.2.-	A4KCE5		19	10	0.5263
71	1fj2A	14	3c5vA		Q9Y570	●	21	11	0.5238
72	1fj2A	14	3fsgA		Q04D10		21	11	0.5238
73	1fj2A	14	3d59A	3.1.1.47	Q13093	●	21	11	0.5238
74	1fj2A	14	1ne7A	3.5.99.6	P46926		20	10	0.5000
75	1fj2A	14	2iksA		P0ACP1		20	10	0.5000
76	1fj2A	14	3icwA	3.1.1.3	P41365		20	10	0.5000
77	1fj2A	14	3tnlA	1.1.1.25	Q8Y9N5		22	11	0.5000
78	1fj2A	14	3qitA	3.1.2.-	D0E8E2		20	10	0.5000
79	1fj2A	14	3cxuA	3.3.2.10	Q41415		20	10	0.5000
80	1fj2A	14	1xu7A	1.1.1.146 1.1.1.-	P28845	●	20	10	0.5000
81	1fj2A	14	4i4cA		D9IR22		23	11	0.4783
82	1fj2A	14	1r6dA	4.2.1.46	Q9ZGH3		23	11	0.4783
83	1fj2A	14	2fqxA		P29724		21	10	0.4762
84	1fj2A	14	1mj5A	3.8.1.5	A0A1L5BTC1		21	10	0.4762
85	1fj2A	14	1ekkA	2.7.1.50	P39593		21	10	0.4762
86	1fj2A	14	2b61A	2.3.1.31	P45131		22	10	0.4545
87	1fj2A	14	1nm2A	2.3.1.39	P72391		22	10	0.4545
88	1fj2A	14	3jyoA	1.1.1.24	Q9X5C9		22	10	0.4545
89	1fj2A	14	2q1sA		O87989		22	10	0.4545
90	1fj2A	14	1q8fA	3.2.2.8	P33022		22	10	0.4545
91	1fj2A	14	3oosA		A0A0F7RDE1		22	10	0.4545
92	1fj2A	14	3ikhA	2.7.1.15	A6T989		22	10	0.4545
93	1fj2A	14	1y1pA	1.1.1.2	Q9UUN9		22	10	0.4545
94	1fj2A	14	3efbA		A0A0H2V3A6		22	10	0.4545
95	1fj2A	14	3b12A	3.8.1.3	Q1JU72		22	10	0.4545
96	1fj2A	14	3k89A		Q5H4I7		22	10	0.4545
97	1fj2A	14	3i8sA		P33650		23	10	0.4348
98	1fj2A	14	2dfdA	1.1.1.37	P40926	●	23	10	0.4348
99	1fj2A	14	1n7hA	4.2.1.47	P93031		24	10	0.4167
100	1fj2A	14	3e48A				24	10	0.4167
101	1fj2A	14	4gxtA		C7RF86		24	10	0.4167
102	1fj2A	14	3k0bA		Q71YC9		24	10	0.4167
103	1fj2A	14	3kv1A	2.7.1.69	Q5E0H6		24	10	0.4167
104	1fj2A	14	4g5hA	4.2.1.115	A0A0H3JPH0		21	12	0.4138
105	1fj2A	14	1orrA	5.1.3.-	P14169		27	11	0.4074
106	1fj2A	14	1k8qA	3.1.1.3	P80035		27	11	0.4074
107	1fj2A	14	3kd6A		Q8KDR9		25	10	0.4000
108	1fj2A	14	4jnkA	1.1.1.27	P00338	●	25	10	0.4000
109	1fj2A	14	3g02A	3.3.2.9	Q9UR30		25	10	0.4000
110	1fj2A	14	3ljsA	2.7.1.4	Q87CC0		25	10	0.4000
111	1fj2A	14	2hrzA		A9CHF5		28	11	0.3929
112	1fj2A	14	1k8qB	3.1.1.3	P80035		28	11	0.3929
113	1fj2A	14	4ooeA	1.1.1.267	P9WNS1	●	28	11	0.3929
114	1fj2A	14	2glqA	3.1.3.1	P05187		26	10	0.3846
115	1fj2A	14	3epwA	3.2.2.1	Q9GPQ4	●	26	10	0.3846
116	1fj2A	14	4juiA		B3Y018		29	11	0.3793
117	1fj2A	14	3ntxA	3.5.1.1	A0A384KLA7		28	10	0.3571
118	1fj2A	14	3rufA		Q7BJX9		28	10	0.3571

Table 6.16: Continued on next page.

Rank	Query		Target					Matching	
	PDB-ID	SSEs	PDB-ID	EC No	UniProt	ChEMBL	SSEs	SSEs	Score
119	1fj2A	14	1sg6A	4.2.3.4	P07547		28	10	0.3571
120	1fj2A	14	2qmaA		Q87NC6		28	10	0.3571
121	1fj2A	14	4eezA	1.1.1.1	D2BLA0		29	10	0.3448
122	1fj2A	14	2zb4A	1.3.1.48	Q8N8N7		29	10	0.3448
123	1fj2A	14	2h6eA	1.1.1.117	Q97YM2		29	10	0.3448
124	1fj2A	14	1ntoA	1.1.1.1	P39462		29	10	0.3448
125	1fj2A	14	4h3vA		D2PU28		29	10	0.3448
126	1fj2A	14	3nrsA		Q8D0U0		30	10	0.3333
127	1fj2A	14	1llfA	3.1.1.3	Q6S5M9		36	12	0.3333
128	1fj2A	14	4hxyA		Q6V1M8		30	10	0.3333
129	1fj2A	14	1yb5A	1.6.5.5	Q08257		30	10	0.3333
130	1fj2A	14	3ju8A	1.2.1.71	O50174		34	11	0.3235
131	1fj2A	14	3ndiA		B5L6K6		31	10	0.3226
132	1fj2A	14	1cs0B	6.3.5.5	P0A6F1		31	10	0.3226
133	1fj2A	14	2ejlA	1.5.1.12	Q5SI02		35	11	0.3143
134	1fj2A	14	1pl8A	1.1.1.14	Q00796	●	32	10	0.3125
135	1fj2A	14	3wmtA	3.1.1.73	Q2UP89		32	10	0.3125
136	1fj2A	14	3TG0A	3.1.3.1	P00634		32	10	0.3125
137	1fj2A	14	4kp7A	1.1.1.267	O96693		32	10	0.3125
138	1fj2A	14	3e2dA	3.1.3.1	Q93P54		36	11	0.3056
139	1fj2A	14	4lmpA	1.4.1.1	I6YEC9		33	10	0.3030
140	1fj2A	14	3uplA		Q2YIM3		33	10	0.3030
141	1fj2A	14	4m9dA	6.3.4.4	Q81JI9		34	10	0.2941
142	1fj2A	14	3iteA		K7NCP5		34	29	0.2941

Table 6.16: Complete list of the matches calculated by SLOT for the query chain 1fj2A@pdb. Complete list of the matches calculated by SLOT for the query protein 1fj2A@pdb in the LSC query target chains dataset for which the score was not 0 and the number of matching SSEs was at least 10. The information of these matches were extended by the EC numbers (EC No), the UniProt accession numbers [186] (UniProt), and the availability (●) of bioactive molecules in the ChEMBL database [187]. Rows highlighted in blue contain query target protein pairs of the *ligand-sensing cores*.

6.4 Scripts

These scripts were written for the creation and preparation of our datasets. All scripts are written in Python and provide a documentation using the command line option `-h` or `--help`.

6.4.1 PDBFTP

The `PDBFTP` script utilizes the FTP interface of the PDB [9] to download the requested protein structure files to a local machine. Due slow transmission rates and unstable connections, it uses a connection timeout of 10 seconds and keeps retrying to reconnect and download files on connection errors. It provides the command line arguments as shown in Table 6.17.

Argument	Description
`<output directory>`	path to the output directory
`-d`	FTP URL to server (ftp.wwpdb.org)
`-u`	FTP directory (/pub/pdb/data/structures/all/pdb/)
`-p`	prefix of input files (pdb)
`-s`	suffix of output files (.pdb)
`-l`	log file path containing a list of synchronized PDB-IDs
`-f`	PDB ids to download
`-r`	PDB line prefixes to be removed (e.g., ANISOU,COMPDN)
`-f`	PDB-IDs file
`--no-unpack`	do not unpack/unzip files
`--lower-pdbid`	lowers the PDB-ID
`--clean-models`	retains only the first model

Table 6.17: Command line arguments, options, and flags of the `PDBFTP` script. Default values are given in parentheses.

All parameters required for the connection and the file handling can be set via command line arguments. Once the script is started, it synchronizes the local directory with the one of the FTP server. The proteins or files to be synchronized can be limited by providing a file containing a list of PDB-IDs (`-f`).

All protein files are available in compressed form. These are extracted automatically. However, this extraction can be suppressed by the use of `--no-unpack`.

A list containing prefixes of lines that are removed from the extracted files can be given via the option `-r`. For instance, `-r REMARK` removes all `REMARK` lines regardless of their remark code.

The PDB file format supports one secondary structure element annotation although an NMR file consists of multiple models with different conformations. Therefore, the flag `--clean-models` can be used to retain only the first model of NMR ensembles and removes all other. `--lower-pdbid` changes the 4 character PDB identifier in the `HEADER` to lowercase to correspond to the filename.

HEADER, SOURCE, AUTHOR, OBSLTE, KEYWDS, REVDAT, TITLE, EXPDTA, SPRSDE, SPLT, NUMMDL, JRNL, CAVEAT, MDLTYP, REMARK, COMPND

List 6.1: Global (non-chain-specific) header PDB line prefixes.

CISPEP, SEQADV, DBREF, DBREF1, DBREF2, LINK, SSBOND, SEQRES, MODRES, HELIX, SHEET, TURN, SITE, ATOM, HETATM, ANISOU, TER, HET

List 6.2: Chain-specific PDB line prefixes.

6.4.2 PDBChainSplitter

The PDBChainSplitter script splits a PDB file into separate chain files. Each chain file contains the global header information but only the chain-specific information in each chain file. Lists 6.1 and 6.2 show the PDB line prefixes containing the global header respectively the chain-specific information. The chain output file names have the chain id as the file name suffix in front of the file extension. Table 6.18 shows the supported command line parameters.

Argument	Description
<input file/directory>	path to the input file or directory
<output file/directory>	path to the output file or directory
<threads>	number of parallel threads
-e	input and output file extension (.pdb)
-l	ligand handling (0)
-t	ligand chain distance threshold (3)
--no-header	no header lines except HEADER
--lower-pdbid	lowers the PDB-ID
--modres-2-atom	HETATM modres lines to ATOM
--rename-chainid	rename chain ids of ligands

Table 6.18: Command line arguments, options, and flags of the PDBChainSplitter script. Default values are given in parentheses.

In some PDB files ligands are not assigned to a chain which requires a specific ligand handling by the use of option -l. We support three different ways of ligand handling:

- 0: use input file chain annotation for the assignment
- 1: assign/copy a ligand to each chain if any ligand's atom is within a (Euclidean) distance threshold -t to any atom of the chain
- 2: copy all ligands to all chain files

If --rename-chainid is not given, the ligand information is copied without modification to the chain files. Otherwise, the chain identifier may be modified according to the selected ligand handling. The differentiation between modified and ligand residues for HETATM lines is based on the MODRES information. If a residue's name is listed in that section of the PDB file, each HETATM

line is interpreted as a modified residue line respectively ligand line otherwise. The line prefixes of modified residues can be changed to `ATOM` if `--modres-2-atom` is given. `HET` lines are per definition interpreted as ligand lines.

The script supports threads. We noticed that due to the I/O-speed limitations of hard drives, the number of threads should be limited to 5. Otherwise, the performance may saturate or even decrease.

6.4.3 PDBModelSplitter

What the `PDBChainSplitter` script (see Section 6.4.2) is for proteins and their chains, the `PDBModelSplitter` script is for NMR ensembles and their models. It splits the models of NMR ensembles into separate PDB files retaining and copying the global information to each model file while keeping only the atom and the missing residue information for a model in each file. The supported command line arguments are given in Table 6.19.

Argument	Description
`<input file/directory>`	path to the input file or directory
`<output file/directory>`	path to the output file or directory

Table 6.19: Command line arguments, options, and flags of the `PDBModelSplitter` script. Default values are given in parentheses.

The models are separated on the basis of `MODEL` and `ENDMDL` lines. The information between such a pair of termini is model-specific and only appear in the respective model file. In addition, the information of missing residues for each model is extracted from the `REMARK 465` lines and assigned to a model based on the model identifier column (column 14).

The termini lines are not retained in the output model files. The output file names consist of the original file name plus a consecutive integer model id and the input file extension.

Affirmation

I hereby affirm that this thesis was written by myself, no further resources and means were used except from the ones mentioned herein, and all citations were linked to their original sources.

Dortmund, February 5, 2020

Tobias Brinkjost