Faberpolynombasierte lokale Zeitschrittverfahren für hyperbolische Systeme der Photonik

von der

Fakultät für Elektrotechnik und Informationstechnik
der
Technischen Universität Dortmund

genehmigte

DISSERTATION

zur Erlangung des akademischen Grades
Doktor der Ingenieurwissenschaften (Dr.-Ing.)

vorgelegt von

Wladimir Plotnikov, M.Sc.

Referent:	apl. Prof. Dr.-Ing. Dirk Schulz, Technische Universität Dortmund
Korreferent:	Prof. Dr. Jasmin Smajic, Eidgenössische Technische Hochschule Zürich

Tag der mündlichen Prüfung: 29.06.2026

Wladimir Plotnikov, M. Sc.
Faberpolynombasierte lokale Zeitschrittverfahren für hyperbolische Systeme der Photonik

Genehmigte Dissertation zur Erlangung des akademischen Grades Doktor der Ingenieurwissenschaften (Dr.-Ing) der Fakultät Elektrotechnik und Informationstechnik der Technischen Universität Dortmund

Prüfungskommission:
Prof. Dr.-Ing. Torsten Bertram, Technische Universität Dortmund (Vorsitz)
apl. Prof. Dr.-Ing. Dirk Schulz, Technische Universität Dortmund
Prof. Dr. Jasmin Smajic, Eidgenössische Technische Hochschule Zürich
Prof. Dr.-Ing. Onur Günlü, Technische Universität Dortmund

Ort und Tag der mündlichen Prüfung: Dortmund, 29.06.2026

"Whoever undertakes to set himself up as a judge of Truth and Knowledge is shipwrecked by the laughter of the gods."

— *Albert Einstein*

Kurzfassung

Klassischerweise wird die Propagation elektromagnetischer Wellen in den unterschiedlichsten Systemen über die Lösung der Maxwell-Gleichungen analysiert. Sofern es die Umstände erlauben, ist eine Visualisierung mit Hilfe einer Simulation stets eine sinnvolle Ergänzung zur theoretischen Betrachtung. Um dies zu ermöglichen, ist der Einsatz numerischer Verfahren notwendig, die die physikalischen Gesetzmäßigkeiten rechnerisch abbilden. Je nach Anwendungsfall kann die Auswertung eines Gleichungssystems mit zahlreichen Freiheitsgraden erforderlich sein. Typische Anwendungsbereiche, für die dies gilt, sind beispielsweise die Photonik und Terahertz-Technik. Hier wird für eine realistische Analyse häufig eine örtliche Auflösung bis in den Nanometerbereich benötigt, um charakteristische Effekte modellieren zu können. Des Weiteren treten in diesen Fällen oft Nichtlinearitäten auf, die sich durch Verstärkung oder Dämpfung bemerkbar machen.

Gemessen an diesem Anspruch ist die Lösung der zugrunde liegenden Gleichungssysteme entsprechend mit einer hohen Komplexität verbunden. Bei einigen etablierten Methoden kann zudem eine Beschränkung der Zeitschrittweite auftreten, welche die Komplexität zusätzlich erhöht. Damit trotz dieser Einschränkungen eine reibungslose Implementierung möglich ist, wäre eine Option, die Berechnung auf einem Rechencluster durchzuführen. Dadurch nimmt der Speicherbedarf vermeintlich eine untergeordnete Rolle ein. Dennoch kann sich diese Annahme als trügerisch erweisen, insbesondere wenn Randbedingungen, Materialmodelle oder 3D-Systeme berücksichtigt werden.

Ein alternativer Ansatz würde sich auf die reine Programmierung stützen. Wird anstelle impliziter Verfahren eine explizite Methode gewählt, lässt sich das Lösen von Gleichungssystemen vollständig umgehen. Stattdessen genügt die Auswertung von Matrix-Vektor-Multiplikationen, die wesentlich einfacher zu handhaben sind als die Bestimmung einer Matrixinversen oder von Eigenwerten. Darüber hinaus ließe sich der Ablauf ohne großen Aufwand parallelisieren, wodurch sich die Rechenzeit erheblich verkürzt. Daran anschließend kann durch eine Erhöhung der Zeitschrittweite die Anzahl der Rechenoperationen weiter reduziert werden.

Unter Berücksichtigung dieser Vorgaben wird der Ansatz auf Basis der Faberpolynome untersucht, der sich durch eine geringe Limitation und hohe Effizienz bei der Approximation des Matrixexponentials auszeichnet. Durch die Weiterentwicklung der konventionellen Faber-Methode, bei der anstatt eines globalen Operators beliebig viele separate lokale Operatoren evaluiert werden, wird eine deutlich höhere Flexibilität erreicht. Diesbezüglich werden theoretische und praktische Resultate miteinander verglichen, um die Gültigkeit der theoretischen Annahmen zu bewerten. Weiterhin wird gezeigt, dass sich die Konvergenz der Faber-Methode über geeignete numerische Abbildungen in andere Domänen beschleunigen lässt. Ferner kann durch den Einsatz von Unterraum-Verfahren die Gesamtkomplexität um mehrere Größenordnungen verringert werden, was die Modellierung anspruchsvoller Systeme wesentlich verbessert.

Danksagung

An erster Stelle möchte ich mich recht herzlich bei apl. Prof. Dr.-Ing. Dirk Schulz für die Möglichkeit bedanken, unter seiner Obhut promovieren zu dürfen. Durch seine offene und direkte Art, die ich bereits während meines Bachelorstudiums sehr geschätzt habe, sowie durch unsere gemeinsame Vorliebe für Australien zeichnete sich dieser Weg früh für mich ab. Ebenso möchte ich meine Dankbarkeit für seine kontinuierlichen Anregungen, Hinweise und konstruktiven Kritiken sowie seine stetige Unterstützung während meiner Tätigkeit als wissenschaftlicher Mitarbeiter zum Ausdruck bringen.

Im selben Atemzug möchte ich Prof. Dr. Jasmin Smajic meinen Dank aussprechen, der nach unserem sehr angenehmen Kennenlernen auf der *NEMO 2024* in Montreal bereitwillig die Funktion des Korreferenten übernommen hat.

Mein Dank gilt zudem meinen treuen Freunden und Kollegen, sowohl aus der Schul- als auch aus der Studienzeit. Eure ehrlichen Worte und wertvolle Unterstützung haben mir in den verschiedensten Phasen des Lebens sehr geholfen. Insbesondere geht ein großes Lob und Dankeschön an meinen langjährigen Weggefährten Valmir Ganiu raus, der mich stets mit interessanten Ansätzen bereicherte und entscheidend zur Korrektur der Arbeit beitrug. Darüber hinaus zolle ich meinen ehemaligen Studenten, die ich betreuen durfte, meine Anerkennung für ihre fleißige und kooperative Mitarbeit.

Nicht zu vergessen gilt meine Dankbarkeit dem *CC HPC Team* der Technischen Universität Dortmund, das mir durch seine engagierte Betreuung Berechnungen mit außergewöhnlich hohem Speicherbedarf ermöglichte.

Selbstredend wäre diese Arbeit ohne die finanzielle Unterstützung für das Projekt *SCHU 1016/6-2* der Deutschen Forschungsgemeinschaft niemals zustande gekommen, wofür ich meinen tiefsten Dank entgegenbringe.

Schließlich geht mein herzlichstes Dankeschön an meine Familie, bei der sich meine unendliche Dankbarkeit nicht in Worte fassen lässt.

Inhaltsverzeichnis

Abkürzungsverzeichnis

ABC	Absorbing Boundary Conditions
ADE	Auxiliary Differential Equations
AMD	Advanced Micro Devices
AMR	Adaptive Mesh Refinement
BLAS	Basic Linear Algebra Subprograms
C	Niedere Programmiersprache C
CE	Complex Envelope
CFL	Courant-Friedrichs-Lewy
CFS	Complex-Frequency-Shifted
CO_2	Kohlenstoffdioxid
CPU	Central Processing Unit
CRDT	Cross-Ratios of the Delaunay Triangulation
CSC	Compressed Sparse Column
CSR	Compressed Sparse Row
CUDA	Compute Unified Device Architecture
D	Dimension
DEIM	Discrete Empirical Interpolation Method
DFB	Distributed Feedback
DGL	Differentialgleichung
EXPRB32	Exponential Rosenbrock 3. Ordnung mit Fehler 2. Ordnung
FD	Finite-Difference
FDTD	Finite-Difference Time-Domain
FEM	Finite-Element-Method
FETI	Finite Element Tearing and Interconnecting
FLOPS	Floating Point Operations per Second
FOM	Full Order Model
FORTRAN	Höhere Programmiersprache FORmula TRAnslation
FPGA	Field-Progammable Gate Array
FVM	Finite-Volume-Method
FWHM	Full Width at Half Maximum
GDDR	Graphics Double Data Rate
GPU	Graphics Processing Unit
HPC	High Performance Computer
HPOD	Hybrid Proper Orthogonal Decomposition

IEEE	Institute of Electrical and Electronics Engineers
IM	Interpolation Methods
Intel	Integrated Electronics
ISVD	Incremental Singular Value Decompostion
L	Level
LAPACK	Linear Algebra Package
LiDO3	Linux Cluster an der TU Dortmund der dritten Generation
LRA	Low-Rank Approximation
LTS	Local Time-Stepping
MAC	Multiply-Accumulate
MATLAB	Höhere Programmiersprache MATrix LABoratory
MMI	Multimode Interference
MOR	Model Order Reduction
MP	Multiprocessing
MT	Multithreading
MVM	Matrix-Vektor-Multiplikation
NB-TDBPM	Narrowband-TDBPM
PC	Personal Computer
PDGL	Partielle Differentialgleichung
PEC	Perfect Electric Conductor
PMC	Perfect Magnetic Conductor
PML	Perfectly Matched Layer
POD	Proper Orthogonal Decomposition
POSIX	Portable Operating System Interface
PU	Processing Units
RAM	Random-Access Memory
RK	Runge-Kutta
ROM	Reduced Order Model
SC	Schwarz-Christoffel
SCPACK	Scalable Linear Algebra PACKage
SVD	Singular Value Decompostion
SWR	Schwarz-Waveform-Relaxation
TDBPM	Time-Domain Beam Propagation Method
TE	Transversal elektrisch
TEM	Transversal elektromagnetisch
THz	Terahertz
TM	Transversal magnetisch
UPML	Uniaxial UPML

VRAM	Video-RAM
WB-TDBPM	Wideband-TDBPM

Symbolverzeichnis

A	Komplexe Konstante für die Translation bei SC
$\mathcal{A}_\mathsf{g}$	Submatrix mit den Elementen des grob diskretisierten Subgebiets
$\mathfrak{a}$	Koeffizient der konformen Abbildung
a	Kleine Halbachse der Ellipse
α	Innenwinkel eines Polygons mit π ausgedrückt
α_x	Frequenzverschiebungsfaktor
a	Mit $\mathcal{A}_\mathsf{g}$ korrespondierenden Einträge des Subvektors
$\vec{B}$	Magnetische Flussdichte
$\mathcal{B}_\mathsf{f}$	Submatrix mit den Elementen des fein diskretisierten Subgebiets
b	Große Halbachse der Ellipse
β	Außenwinkel eines Polygons mit π ausgedrückt
β	Phasenkonstante
b	Mit $\mathcal{B}_\mathsf{f}$ korrespondierenden Einträge des Subvektors
C	Komplexe Konstante für die Skalierung und Drehung bei SC
$\mathcal{C}$	Label zur Kennzeichnung einer Rechnung auf der CPU
C_{F2R}	Kriterium bei der hybriden POD für den Wechsel vom FOM zum ROM
C_{fein}	Kostenfunktion für die lokalen Operatoren der feinen Subgebiete
C_{glob}	Kostenfunktion für den globalen Operator
C_{grob}	Kostenfunktion für die lokalen Operatoren der groben Subgebiete
C_{lok}	Gesamte Kostenfunktion für die lokalen Operatoren
C_{R2F}	Kriterium bei der hybriden POD für den Wechsel vom ROM zum FOM
c	Lichtgeschwindigkeit im homogenen Medium, $c = \frac{1}{\sqrt{\epsilon\mu}}$
$\mathfrak{c}$	Halbe Breite des Rechtecks
$\mathfrak{c}_s$	Halbe Breite des skalierten Rechtecks
c_m	Entwicklungskoeffizient
c_{th}	Schwellenwert für die Entwicklungskoeffizienten c_m
$\vec{D}$	Elektrische Flussdichte
D	Skalierungsfaktor für die Komplexität
Δt_{CFL}	Zeitschrittweite nach CFL
$\Delta\mathcal{E}_i$	Relativer Fehler der i-ten POD-Mode
$\Delta\vec{N}$	Effektive Ladungsträgerdichte, $\Delta N = \vec{N_1} - \vec{N_2}$
ΔT	Intervalllänge
Δt	Zeitschrittweite
Δt_s	Skalierte Zeitschrittweite

Δx	Örtliche Diskretisierungsweite in x-Richtung
Δx_{f}	Örtliche Diskretisierungsweite im feinen Subgebiet in x-Richtung
Δx_{g}	Örtliche Diskretisierungsweite im groben Subgebiet in x-Richtung
Δy	Örtliche Diskretisierungsweite in y-Richtung
Δy_{f}	Örtliche Diskretisierungsweite im feinen Subgebiet in y-Richtung
Δy_{g}	Örtliche Diskretisierungsweite im groben Subgebiet in y-Richtung
Δz	Örtliche Diskretisierungsweite in z-Richtung
D_x^E	Untermatrix für die Rotation der $\vec{E}$-Komponente in x-Richtung
D_x^H	Untermatrix für die Rotation der $\vec{H}$-Komponente in x-Richtung
D_y^E	Untermatrix für die Rotation der $\vec{E}$-Komponente in y-Richtung
D_y^H	Untermatrix für die Rotation der $\vec{H}$-Komponente in y-Richtung
d	Breite des Rechtecks
d_s	Skalierte Breite des Rechtecks
d_x	Dicke der PML
$\vec{E}$	Elektrisches Feld
$\mathcal{E}_i$	Energie der i-ten POD-Mode
e	Elementarladung, $e = 1{,}602\,\mathrm{C}$
e_a	Absoluter Fehler
$\mathbb{1}$	Einheitsmatrix
ϵ	Permittivität, $\epsilon = \epsilon_0 \epsilon_r$
ϵ_0	Permittivität im Vakuum, $\epsilon_0 \approx 8{,}854 \cdot 10^{-12}\,\mathrm{As/Vm}$
ϵ_∞	Permittivitätskonstante bei $f \to \infty$
ε	Schwellenwert für die POD
ε_h	Schwellenwert für die HPOD
ϵ_r	Relative Permittivität
ϵ_s	Relative Permittivität bei $f \to 0$
e_r	Relativer Fehler
η	Urbildpunkt bei SC
$\exp(\Delta t \mathcal{H})$	Matrixexponential
F_m	Faberpolynom
f	Frequenz
f	Index zur Kennzeichnung eines Parameters hinsichtlich des feinen Subgebiets
$f(\mathcal{H})$	Linearer Matrixoperator
f_{f}	Lokaler Operator hinsichtlich des feinen Subgebiets
f_{g}	Lokaler Operator hinsichtlich des groben Subgebiets
G	Gebiet
Γ	Einhüllende
Γ_m	Halbwertsbreite

Γ_p	Polygon
G	Label zur Kennzeichnung einer Rechnung mit globalem Operator
$\mathcal{G}$	Label zur Kennzeichnung einer Rechnung auf der GPU
γ	Koeffizient der inversen konformen Abbildung
γ_{12}	Übergangsrate von *Niveau 1* zu *Niveau 2*
γ_{21}	Übergangsrate von *Niveau 2* zu *Niveau 1*
γ_D	Kollisionsfrequenz zwischen Elektronen und Atomen im Drude-Modell
g	Index zur Kennzeichnung eines Parameters hinsichtlich des groben Subgebiets
$\vec{H}$	Magnetisches Feld
$\mathcal{H}_s$	Skalierte Systemmatrix
$\mathcal{H}$	Systemmatrix
$\mathcal{H}_r$	Reduzierte Systemmatrix
$\hbar$	Reduzierte Planck-Konstante, $\hbar \approx 1,055 \cdot 10^{-34}$ Js
i	Laufvariable
$\vec{J}$	Elektrische Stromdichte
$\vec{J}_D$	Polarisationsstrom
J_m	Besselfunktion der Ordnung m
j	Imaginäre Zahl, j $= \sqrt{-1}$
K	Kontinuum
$\vec{K}$	Oberflächenstrom
k	Wellenvektor
k_0	Kreiswellenzahl, $k_0 = \frac{2\pi}{\lambda}$
κ_x	Absorptionsfaktor
k	Konstellation
Λ	Gitterperiode
$\mathcal{L}$	Koppellänge
L	Label zur Kennzeichnung einer Rechnung mit lokalen Operatoren
l	Länge des Rechtecks
λ	Wellenlänge
λ_s	Skalierungsfaktor
l	Länge der Struktur
l_s	Skalierte Länge des Rechtecks
$\vec{M}$	Magnetische Polarisation
M	Anzahl der Moden
m	Laufvariable
m	Anzahl der Zeilen
m_a	Skalierungsordnung
m_D	Masse eines Leitungselektrons im Metallleiter

m_{f}	Entwicklungsordnung hinsichtlich des feinen Subgebiets
m_{g}	Entwicklungsordnung hinsichtlich des groben Subgebiets
μ	Permeabilität, $\mu = \mu_0 \mu_r$
μ_0	Permeabilität im Vakuum, $\mu_0 \approx 1{,}256 \cdot 10^{-6}\,\mathrm{N/A^2}$
μ_r	Relative Permeabilität
N	Anzahl der gesamten Diskretisierungspunkte
$\vec{N_1}$	Besetzungsdichte in dem *Niveau 1*
$\vec{N_2}$	Besetzungsdichte in dem *Niveau 2*
N_{f}	Anzahl der Abtastpunkte zur Diskretisierung des feinen Subgebiets
N_{g}	Anzahl der Abtastpunkte zur Diskretisierung des groben Subgebiets
$\mathcal{N}$	Nichtlinearer Matrixoperator
N_{pol}	Entwicklungsordnung der Faberreihe
N_x	Anzahl der Diskretisierungspunkte in x-Richtung
N_y	Anzahl der Diskretisierungspunkte in y-Richtung
N_z	Anzahl der Diskretisierungspunkte in z-Richtung
n	Anzahl der Eckpunkte des Polygons Γ_p
n_0	Brechungsindex im Vakuum
n_1	Brechungsindex in *Medium 1*
n_2	Brechungsindex in *Medium 2*
n	Anzahl der Spalten
n_D	Elektronendichte
$\vec{\nabla}\cdot$	Divergenzoperator
n_{eff}	Effektiver Brechungsindex
n_L	Brechungsindex im Lasermedium
$\vec{\nabla}\times$	Rotationsoperator
n_S	Anzahl der Snapshots
$\vec{n}$	Oberflächennormalenvektor
Ω	Rechengebiet
Ω_{f}	Fein diskretisiertes Subgebiet
Ω_{g}	Grob diskretisiertes Subgebiet
ω	Eckpunkt eines Polygons
${\omega_D}^2$	Plasmafrequenz des Elektronengases
ω	Kreisfrequenz, $\omega = 2\pi f$
ω_m	Übergangsfrequenz
$\vec{P}$	Elektrische Polarisation
$\phi(\mathfrak{z})$	Inverse konforme Abbildung
$\vec{P}_m$	Nichtlinearer Polarisationsdichte des m-ten elektronischen Übergangs
P	Orthogonale Projektionsmatrix

$\vec{\Psi}$	Zustandsvektor
ψ	Snapshotmatrix
Ψ	Teillösung in ψ
$\psi(\mathfrak{w})$	Konforme Abbildung
p	Entwicklungskoeffizient bei POD
φ	Hilfsfunktion zur Berechnung des nichtlinearen Teils bei Exponential-Euler
π	Kreiszahl, $\pi \approx 3,1416$
p	Index zur Kennzeichnung des parallelisierten Ablaufs
$\vec{\Psi}_{ref}$	Referenzlösung
q	Zeitfaktor, $q = \Delta t / \Delta t_{CFL}$
R	Reflexionsfaktor
$R_{\Delta x}$	Refinemenfaktor in x-Richtung
$R_{\Delta y}$	Refinementfaktor in y-Richtung
$\vec{r}$	Ortsvektor
ρ	Ladungsdichte
ϱ	Logarithmische Kapazität
r	Rang der Matrix
Σ	Diagonalmatrix
σ_i	Singulärwert der Snapshotmatrix
S	Skalierungsfaktor
σ	Elektrische Leitfähigkeit
σ_g	Varianz
$\sigma_s(\mathcal{H})$	Eigenwertspektrum der Systemmatrix $\mathcal{H}$
$\sigma_s(\mathcal{H}_s)$	Skaliertes Eigenwertspektrum der Systemmatrix $\mathcal{H}$
σ_{m}	Kopplungsfaktor zwischen $\vec{E}$ und $\Delta\vec{N}$
σ_s	Elektrische Leitfähigkeit bei $f \to 0$
σ_x	Abklingfaktor
s	Index zur Kennzeichnung des seriellen Ablaufs
s	Index zur Kennzeichnung des Splittingansatzes
s_x	Streckungskoeffizient
t	Zeitkonstante
τ	Relaxationszeit
θ	Winkel
t_{off}	Intervalllänge der Offline-Phase
t_{on}	Intervalllänge der Online-Phase
U	Orthogonale Matrix
u	Linker Singulärvektor bzw. POD-Mode von $\mathcal{H}$
u	Hilfsvariable

V	Orthogonale Matrix
v	Rechter Singulärvektor von $\mathcal{H}$
w	Breite der Struktur
$\mathfrak{w}$	Komplexe Ebene
Ξ	Hilfsfunktion
x	Raumrichtung im kartesischen Koordinatensystem
x_0	Mittelpunkt in x-Richtung
x	Konstante
y	Raumrichtung im kartesischen Koordinatensystem
y_0	Mittelpunkt in y-Richtung
z	Raumrichtung im kartesischen Koordinatensystem
$\mathfrak{z}_0$	Mittelpunkt, $\mathfrak{z}_0 = x_0 + jy_0$
$\mathfrak{z}$	Komplexe Ebene, $\mathfrak{z} = x + jy$

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1

Einleitung

Seit dem Einläuten des digitalen Zeitalters Mitte des 20. Jahrhunderts erfährt die Menschheit einen technologischen Wandel in einer Geschwindigkeit, wie sie ihn in der Geschichte bis dato noch nie durchlebt hat [1]. In diesem Kontext werden stetig höhere Anforderungen an die Informationstechnik gestellt. Im Alltag äußern sich diese u.a. über eine hohe Verfügbarkeit, Zuverlässigkeit und Sicherheit. Um den damit verbundenen Abläufen gerecht zu werden, ist eine Erhöhung der Datenrate unabdingbar. Während Übertragungen im Gigabit-Bereich längst keine Herausforderung mehr darstellen, richtet sich die aktuelle Forschung verstärkt auf Kommunikationssysteme im Terahertz (THz)-Bereich [2]. Im Zuge dessen wird die Minimierung der benötigten Hardware aus Kosten-, Platz- und Energieeffizienzgründen angestrebt [3]. Idealerweise lässt sich der Einsatz der Komponenten vollständig durch realistische Simulationen vermeiden. Dabei beruht die Modellierung auf dem Lösen von Gleichungssystemen.

In der klassischen Elektrotechnik bilden die Maxwell-Gleichungen [4] die Grundlage für die Modellierung elektromagnetischer Phänomene sowie hyperbolischer Systeme. Trotz ihres langen Bestehens und der fortgeschrittenen Technologie des 21. Jahrhunderts bleibt das Lösen der Maxwell-Gleichungen je nach Kontext eine anspruchsvolle Herausforderung. Eine analytische Lösung ist in der Regel nur für stark vereinfachte, idealisierte Systeme mit hoher Symmetrie oder homogenen Materialeigenschaften vorhanden [5]. Aus jenem Grund muss für komplexere Systeme auf numerische *Solver* ausgewichen werden. Ein typisches Merkmal solcher komplexen Systeme, wie sie etwa in der THz-Technik und der Photonik vorkommen, ist das Auftreten von Nichtlinearitäten. Dort ist die Analyse der Wechselwirkung zwischen Licht und Materie respektive einer äußeren Anregung und der resultierenden Materialantwort von zentraler Bedeutung [6, 7].

Vor diesem Hintergrund wurden in den letzten Jahrzehnten diverse numerische *Solver* entwickelt, die auf unterschiedliche Weise eine Näherungslösung für die Propagation einer elektromagnetischen Welle ermöglichen. In diesem Zusammenhang weist jeder von ihnen spezifische Vor- und Nachteile auf, deren Eignung vom jeweiligen Einsatzgebiet abhängt. Eine Klassifizierung kann beispielsweise anhand der Domäne im Frequenz- oder Zeitbereich, der zugrunde liegenden Integral- oder Differenzialformulierung sowie einer expliziten oder impliziten Darstellung erfolgen [8]. Ergänzend dazu kommt der räumlichen Diskretisierung eine besondere Bedeutung zu, weil sie pro Abtastpunkt möglichst speichereffizient umgesetzt werden muss [9]. Unter diesen Gesichtspunkten ist der am häufigsten eingesetzte *Solver* die Finite-Difference Time-Domain (FDTD)-Methode [10], die auf einer expliziten Differentialformulierung im Zeitbereich basiert. Mit ihr lässt sich neben dem Ort auch die Zeit diskretisieren, was elementar für die Approximation zeitabhängiger Prozesse ist. Hierbei koppelt das Courant-Friedrichs-Lewy (CFL)-Kriterium [11] die Orts- und die Zeitauflösung, wodurch bei hoher örtlicher Auflösung außergewöhnlich kleine Zeitschrittweiten folgen. Um dieser Limitierung entgegenzuwirken, eignet sich der Matrixexponential-Ansatz für die Zeitintegration [12]. Dennoch stößt auch der Matrixexponential-Ansatz an seine Grenzen,

sobald eine extrem hohe Anzahl an Diskretisierungspunkten vorliegt [13]. Infolgedessen ist eine Approximation des Matrixexponentials unvermeidlich.

Zu diesem Anlass bietet sich besonders die Faber-Methode [14] an, die in der Elektrotechnik bislang kaum Beachtung fand, jedoch bereits bemerkenswertes Potenzial zur effizienten Approximation des Matrixexponentials gezeigt hat [15]. So erlaubt die Faber-Methode die Approximation des zeitabhängigen Propagators sowohl in linearen als auch in nichtlinearen Systemen, selbst unter dem Einfluss von Dämpfungen und Verstärkungen [16]. Da die Systemmatrix direkt aus der FDTD-Methode hervorgeht, lassen sich Quellterme, Materialmodelle und Randbedingungen intuitiv integrieren, indem die Systemmatrix mit den entsprechenden Einträgen erweitert wird [16, 17]. Zudem kann bei der Approximation mit den Faberpolynomen und den zugehörigen Entwicklungskoeffizienten auf analytische Ausdrücke zurückgegriffen werden, was die numerische Umsetzung signifikant vereinfacht [18]. Weiterhin sind für die Auswertung ausschließlich Matrix-Vektor-Multiplikationen (MVMs) notwendig, wodurch eine gezielte Parallelisierung implementiert werden kann. Ebenso übertrifft die Genauigkeit der Faber-Methode jene der FDTD-Methode um mehrere Größenordnungen [19]. Da die Berechnung im Eigenwertraum erfolgt, besteht der einzige wesentliche Nachteil in der erforderlichen Kenntnis des Eigenwertspektrums der Systemmatrix. Aus diesem Umstand ergeben sich zwei zentrale Konsequenzen. Erstens hängt die Zeitschrittweite unmittelbar von der Qualität der Eigenwertabschätzung ab. Je präziser diese durchgeführt wird, desto höhere Zeitschrittweiten können verwendet werden. Zweitens müssen zur Sicherstellung der Stabilität der Faber-Methode alle Eigenwerte innerhalb der gewählten Kontur liegen. Ist dies nicht der Fall, kommt es zwangsläufig zur Divergenz der Faberpolynome [20].

1.1 Motivation

Im Folgenden werden die Beweggründe dieser Arbeit erläutert, indem der aktuelle Stand der Technik dargelegt wird. Dabei wird aufgezeigt, welche Herausforderungen und Hürden in der numerischen Evaluierung bestehen. Teilweise sind die Motivationen eng miteinander verknüpft und können daher nur gemeinsam betrachtet werden.

Laufzeit

Unabhängig vom Zweck dienen Simulationen als effektives Mittel zur Modellierung realer Systeme, wenngleich ihre Laufzeiten mit zunehmender räumlicher Auflösung erheblich ansteigen können. In der Photonik werden für Anwendungen wie Nanowellenleiter [21] und Resonatorstrukturen [22] Bauelemente im Nanometerbereich gefertigt. Werden diese Komponenten für die numerische Berechnung mittels der klassischen FDTD-Methode diskretisiert, ergibt sich gemäß der CFL-Bedingung eine maximale Zeitschrittweite in der Größenordnung von $\Delta t \approx 10^{-18}\,\mathrm{s}$. In Zeiten, in denen die Ansprüche bis hin zu Echtzeitsimulationen reichen, erweisen sich derart kleine Zeitschrittweiten als nicht tragbar. Des Weiteren ist es gängige Praxis, anspruchsvolle Berechnungen auf Rechenclustern durchzuführen. Dementsprechend wirken sich lange Rechenzeiten negativ auf den Energieverbrauch aus, da sie zu einem erhöhten Strombedarf führen. Die dabei entstehende Abwärme muss aktiv abgeführt werden, was zusätzlichen Energieeinsatz erfordert und in Summe den Kohlenstoffdioxid (CO_2)-Fußabdruck des Systems deutlich erhöht [23]. Aus diesen Gründen besteht eine Motivation dieser Arbeit darin, die Simulationsdauer nachhaltig zu reduzieren.

Genauigkeit

Weiterhin soll die Qualität der generierten Ergebnisse ein hohes Niveau aufweisen. Dadurch sollen ungünstige Entscheidungen beim Design von Bauteilen, wie etwa Fehlanpassungen von Antennen [24] oder unzureichende Abschirmkonzepte gegen Ein- und Ausstrahlungen [25], vermieden und Optimierungspotenziale bestmöglich ausgeschöpft werden können. Ebenso lassen sich elektromagnetische Störeffekte bezüglich THz-Anwendungen präziser vorhersagen, zu denen u.a. Streuverluste [26] und nichtlineare Polarisationseffekte [27] zählen. Ferner kann es in optischen Systemen an bestimmten Stellen durch Kanten oder Materialdefekte zu übermäßigen Feldkonzentrationen kommen [28], die schlimmstenfalls in optischen Durchbrüchen enden [29]. Angesichts dieser Tatsachen ist das Erreichen einer adäquaten Genauigkeit der Approximation zwingend erforderlich und dient somit zugleich als zusätzliche Motivation.

Komplexität

Da komplexe Modelle nicht immer unmittelbar nachvollziehbar sind, ist es ratsam, sie so transparent und flexibel wie möglich zu gestalten. Wird dieses Grundprinzip berücksichtigt, lassen sich einfache Systeme ohne großen Mehraufwand für detailliertere Analysen skalieren. Dieser Gedankengang lässt sich auf eine Vielzahl von Konstrukten übertragen. Dazu zählen der Übergang von linearen zu nichtlinearen Systemen, das Einbinden von Randbedingungen sowie die Kombination aus etablierten und neuen *Solvern*. Im Hinblick auf die Graphics Processing Unit (GPU)-gestützte Beschleunigung der Approximation stellt die verfügbare Video Random Access Memory (VRAM)-Kapazität den limitierenden Flaschenhals dar. Zwar bieten Advanced Micro Devices (AMD) [30] und Nvidia [31] mittlerweile Workstations mit ungefähr 100 GB VRAM-Kapazität, allerdings erweist sich selbst diese Speichergröße bei sehr hoch aufgelösten Rechengebieten als unzureichend. Daher ist auch die Reduktion der Komplexität eine Motivation dieser Arbeit, da sie maßgeblich zur Laufzeit eines Algorithmus beiträgt.

1.2 Zielsetzung

Mit den Publikationen [16, 17, 19, 32–35] sowie deren Zusammenfassung in der Dissertation [15] legte Kleene den Grundstein zur numerischen Approximation der Maxwell-Gleichungen mittels der Faberpolynome im Zeitbereich. Aufbauend auf den vielversprechenden Ergebnissen seiner Forschungen verfolgt diese Arbeit das Ziel, die Effizienz der Faber-Methode weiter zu optimieren. Hierbei sind besonders die Komplexität und die damit verbundene Laufzeit hervorzuheben. Wird eine Zeitschrittweite gewählt, die deutlich über dem CFL-Kriterium liegt, so ist die Genauigkeit bei der Faber-Methode signifikant höher als bei der FDTD-Methode [19]. Aus diesem Grund spielt die Genauigkeit eine untergeordnete Rolle, wobei trotzdem eine tolerierbare Größenordnung gefordert wird. Zudem sei darauf hingewiesen, dass die Stabilität der Approximation nicht als separates Ziel definiert wird, da eine garantierte Stabilität durch eine angemessene Eigenwertabschätzung der inhärenten Systemmatrix angenommen werden kann [20].

Konvergenz

Unter Berücksichtigung der Zielsetzung beschäftigt sich demnach ein Ansatz mit der Reduktion der Anzahl der Faberpolynome, die jeweils für die Evaluierung eines Zeitschritts benötigt wird. Weil die Faberpolynome über eine Reihe entwickelt werden [36], liegt es nahe, die Konvergenz mit einer niedrigen Entwicklungsordnung zu erreichen. Diesbezüglich kann mit Hilfe der Ergebnisse der Voruntersuchungen die Verteilung der Eigenwerte des gegebenen Systems präzise approximiert werden. Gleichzeitig eröffnet sich dadurch die Möglichkeit, die charakteristische Kontur zur Eingrenzung der Eigenwerte noch genauer zu ermitteln. Außerdem soll eine Erkenntnis gewonnen werden, die eine systemunabhängige Skalierung zulässt.

Dekomposition

Eine weitere, bisher nicht untersuchte Methode beruht auf der Anwendung einer Dekomposition, bei der ein großes Problem in mehrere kleinere Teilprobleme unterteilt wird. Analog dazu kann ein *Operatorsplitting* betrieben werden. Dies soll ebenfalls unabhängig vom jeweiligen System geschehen. Hinsichtlich der Approximation mit der Faber-Methode sind separate Auswertungen vorgesehen, die am Ende der Entwicklung wieder zusammengefügt werden. Basierend darauf kann neben der Software auch die Hardware einbezogen werden, wodurch sich der sequentielle Prozessablauf aufgrund der expliziten Formulierung in einen parallelen überführen lässt. In erster Linie bietet sich dazu das Multiprocessing (MP) auf der Central Processing Unit (CPU) an. Da die Entwicklung hauptsächlich über MVMs erfolgt, ist auch die Parallelisierung auf der GPU mittels Multithreading (MT) gerechtfertigt. Darüber hinaus repräsentiert die Kombination der CPU und GPU eine pragmatischen Ansatz, um den optimalen Prozessablauf zu verwirklichen.

Projektion

Ungeachtet dessen soll zusätzlich die Projektion des zeitabhängigen Propagators in einen Unterraum vorgenommen werden. Der Informationsverlust, bedingt durch die Kompression, wird auf ein Minimum reduziert. Zu diesem Zweck wird das System zunächst trainiert, um den Propagator anschließend mit verringertem Rechenaufwand extrapolieren zu können. Hierdurch wird die Komplexität um mehrere Größenordnungen gesenkt, was die Methode besonders für Anwender mit begrenztem Speicher attraktiv macht, die keinen Zugriff auf High Performance Computer (HPC) oder verteilte Rechnersysteme haben. Angesichts des limitierten Speichers einer GPU kann dieser Ansatz ebenfalls von Vorteil sein, sodass selbst komplexere Systeme auf ihr berechnet werden können.

1.3 Gliederung

Die vorliegende Arbeit ist in sechs Hauptkapitel und ein Fazit unterteilt. Die ersten drei Kapitel befassen sich mit der etablierten Theorie, während die darauffolgenden drei Hauptkapitel die Analyse bewährter Verfahren zur potenziellen Effizienzsteigerung der Faber-Methode behandeln.

Grundlagen

In Kapitel 2 werden die relevanten physikalischen Grundlagen, gängige Randbedingungen sowie die im weiteren Verlauf verwendeten Materialmodelle erläutert. Um die physikalischen Modelle

maschinell verarbeiten zu können, werden in Kapitel 3 die numerischen Grundlagen erläutert. Dabei liegt der Fokus auf der örtlichen Diskretisierung nach Yee sowie auf der Operatornotation und dem Konzept der Systemmatrix und des Zustandsvektors, die in dieser Arbeit im Mittelpunkt stehen. Das Kapitel 4 thematisiert die Approximation des zeitabhängigen Propagators durch die Faberpolynome. Insbesondere werden die Abschätzung und Skalierung des Eigenwertspektrums der Systemmatrix, die praktische Umsetzung der Faber-Methode sowie deren Anwendung und der Stand der Forschung im Kontext der Maxwell-Gleichungen beschrieben.

Hauptteil

In Kapitel 5 wird die Frage untersucht, ob eine engere Umschließung des Eigenwertspektrums durch die charakteristische Kontur zur Reduktion der Laufzeit beitragen kann. Dazu werden bekannte, analytisch bestimmte Konturen mit solchen verglichen, die mittels der Schwarz-Christoffel-Transformation numerisch ermittelt werden können. Das Kapitel 6 ist den neu entwickelten lokalen Operatoren gewidmet, die eine Erweiterung der Faber-Methode um den globalen Operator darstellen. Anhand von unterschiedlichen Analysen wird die Implementierung der lokalen Operatoren detailliert betrachtet und auf mögliche weitere Effizienzoptimierungen eingegangen. Das Kapitel 7 setzt sich mit der Komplexitätsreduktion der Faber-Methode durch die Anwendung der Proper Orthogonal Decomposition auseinander. Speziell die Modifizierung zur Hybrid Proper Orthogonal Decomposition steht im Blickpunkt, die eine höhere Dynamik der statischen Evaluation ermöglicht.

Fazit

Abschließend werden in Kapitel 8 die gewonnenen Erkenntnisse zusammengefasst und im Detail ausgewertet. Diesbezüglich werden die Resultate kritisch hinterfragt und ihr tatsächlicher Mehrwert für die Forschung diskutiert. Auf Grundlage dessen wird ein Ausblick auf zukünftige Untersuchungen gegeben, indem offene Fragestellungen und mögliche Forschungsperspektiven skizziert werden.

Kapitel 2

Physikalische Grundlagen und Modelle

Im Folgenden werden die benötigten physikalischen Grundlagen im Rahmen dieser Arbeit rudimentär dargelegt. Als Einstieg wird die Lehre des klassischen Elektromagnetismus nach Maxwell behandelt. Daraufhin werden verschiedene Materialmodelle näher erörtert, die das Nachbilden von realitätsnahen Systemen erlauben.

2.1 Maxwell-Gleichungen

Die Grundlage für die heutige elektromagnetische Feldtheorie hat James Clerk Maxwell bereits im Jahre 1865 geschaffen [4]. Die ihm zu Ehren genannten Maxwell-Gleichungen ermöglichen es das Wechselwirken zwischen einer elektromagnetischen Welle und angrenzender Materie zu beschreiben. Die Formulierung kann sowohl in Integralform als auch in Differentialform erfolgen. Im weiteren Verlauf wird auf Letzteres Bezug genommen, da es mit den später vorgestellten numerischen Methoden korrespondiert. Dementsprechend sind die vier Maxwell-Gleichungen in differentieller Form mit dem Divergenzoperator $\vec{\nabla}\cdot$ und dem Rotationsoperator $\vec{\nabla}\times$ definiert als

$$\vec{\nabla}\cdot\vec{D} = \rho, \qquad (2.1) \qquad\qquad \vec{\nabla}\cdot\vec{B} = 0, \qquad (2.2)$$

$$\vec{\nabla}\times\vec{E} = -\frac{\partial\vec{B}}{\partial t}, \qquad (2.3) \qquad\qquad \vec{\nabla}\times\vec{H} = \vec{J} + \frac{\partial\vec{D}}{\partial t}. \qquad (2.4)$$

Die Gleichung (2.1) ist bekannt als Gaußsches Gesetz und besagt, dass die Ladungsdichte ρ ein elektrisches Feld $\vec{E}$ mit der elektrischen Flussdichte $\vec{D}$ verursacht. Daraus abgeleitet ergibt sich das Gaußsche Gesetz für Magnetfelder aus der Gleichung (2.2) mit der Aussage, dass ein Magnetfeld $\vec{H}$ mit der magnetischen Flussdichte $\vec{B}$ quellenfrei ist. Weiterhin bewirkt die zeitliche Änderung der magnetischen Flussdichte ein elektrisches Wirbelfeld, wie es mit dem Induktionsgesetz aus der Gleichung (2.3) gezeigt wird. Das Durchflutungsgesetz aus der Gleichung (2.4) beschreibt den Zusammenhang zwischen der Entstehung eines Magnetfeldes und einem zeitlich veränderlichen Strom sowie der elektrischen Stromdichte $\vec{J}$ in einem leitfähigen Material.

Des Weiteren sind in diesem Kontext die charakteristischen Materialgleichungen von Bedeutung, da sie die Feldstärken, Flussdichten und Stromdichte miteinander in Relation setzen. Durch die Einführung der elektrischen Polarisation $\vec{P}$ und der magnetischen Polarisation $\vec{M}$ sind die Beziehungen im allgemeinen Fall wie folgt gegeben:

$$\vec{D} = \epsilon_0\vec{E} + \vec{P}, \qquad (2.5) \qquad\qquad \vec{B} = \mu_0(\vec{H} + \vec{M}), \qquad (2.6) \qquad\qquad \vec{J} = \sigma\vec{E}. \qquad (2.7)$$

Die hier in Erscheinung tretenden Parameter ϵ_0 aus der Gleichung (2.5) und μ_0 aus der Gleichung (2.6) repräsentieren jeweils die Permittivität und Permeabilität im Vakuum. Die Verknüpfung zwischen dem elektrischen Feld und der elektrischen Stromdichte ergibt sich über die Gleichung (2.7) unter Einbeziehung der elektrischen Leitfähigkeit σ. Sollte ein homogenes, isotropes und lineares System vorliegen, vereinfachen sich die Ausdrücke aus der Gleichung (2.5) und Gleichung (2.6) zu

$$\vec{D} = \epsilon\vec{E} = \epsilon_0\epsilon_r\vec{E}, \tag{2.8}$$

$$\vec{B} = \mu\vec{H} = \mu_0\mu_r\vec{H}. \tag{2.9}$$

Hierbei ist ϵ_r die relative Permittivität, während μ_r die relative Permeabilität kennzeichnet. Beide Größen gelten als materialabhängig und dimensionslos. Sofern Quellenfreiheit angenommen wird, kann die elektrische Stromdichte vernachlässigt werden. Demzufolge kann die Gleichung (2.8) in die Gleichung (2.3) sowie die Gleichung (2.9) in die Gleichung (2.4) eingesetzt werden, sodass der unmittelbare Zusammenhang zwischen dem elektrischen Feld $\vec{E}$ und dem magnetischen Feld $\vec{H}$ hergestellt werden kann:

$$\frac{\partial\vec{E}}{\partial t} = \frac{1}{\epsilon}\vec{\nabla}\times\vec{H}, \tag{2.10}$$

$$\frac{\partial\vec{H}}{\partial t} = -\frac{1}{\mu}\vec{\nabla}\times\vec{E}. \tag{2.11}$$

Da das kartesische Koordinatensystem mit den Raumrichtungen x, y und z als Bezugssystem dient, ergeben sich die Vektoren $\vec{E} = [E_x, E_y, E_z]^T$ und $\vec{H} = [H_x, H_y, H_z]^T$ mit jeweils drei Feldkomponenten. Hierdurch kann die Wellenausbreitung im 1D-, 2D- oder 3D-Raum über partielle Differentialgleichungen (PDGLs) beschrieben werden.

Die Simulationen von 1D-Systemen eignen sich besonders für den Einstieg in den klassischen Elektromagnetismus, weil sie die Komplexität auf das Nötigste reduzieren und eine einfache Nachvollziehbarkeit ermöglichen [37]. Technisch betrachtet können potenzielle Randbedingungen oder verlustbehaftete Systeme zunächst isoliert untersucht und im Anschluss gezielt für detailliertere Analysen verfeinert werden [8]. In diesem Kontext lässt sich auch die Wellenausbreitung in einer Doppeladerleitung modellieren, wie sie etwa in der Telefonie oder Datenübertragung verwendet wird [38]. Wird diesbezüglich die Wellenausbreitung ausschließlich in x-Richtung angenommen und die Polarisation auf die y- und z-Richtung beschränkt, ergeben sich die folgenden transversal elektromagnetischen Moden (TEM-Moden) für ein 1D-System:

$$\left.\begin{aligned}\frac{\partial E_y}{\partial t} &= -\frac{1}{\epsilon}\frac{\partial H_z}{\partial x}\\ \frac{\partial H_z}{\partial t} &= -\frac{1}{\mu}\frac{\partial E_y}{\partial x}\end{aligned}\right\} \begin{array}{l}x\text{-gerichtete,}\\ y\text{-polarisierte}\\ \text{TEM-Mode,}\end{array} \tag{2.12}$$

$$\left.\begin{aligned}\frac{\partial E_z}{\partial t} &= \frac{1}{\epsilon}\frac{\partial H_y}{\partial x}\\ \frac{\partial H_y}{\partial t} &= \frac{1}{\mu}\frac{\partial E_z}{\partial x}\end{aligned}\right\} \begin{array}{l}x\text{-gerichtete,}\\ z\text{-polarisierte}\\ \text{TEM-Mode.}\end{array} \tag{2.13}$$

Die Erweiterung auf eine 2D-Struktur bietet die Möglichkeit, physikalische Phänomene noch realistischer zu erforschen. In der Leitungstheorie lassen sich geführte Wellen in komplexeren Wellenleiterstrukturen modellieren, wie beispielsweise in Streifenleitungen [39] oder Panelantennen [40]. Für die Simulation einer 2D-Struktur befindet sich die Feldverteilung in der x-y-Ebene, während sich die Welle in z-Richtung ausbreitet. Die korrespondierenden transversal elektrische Mode (TE_z-Mode) und transversal magnetische Mode (TM_z-Mode) sind über ihre jeweiligen longitudinalen Feldkomponenten definiert:

$$\left.\begin{aligned}\frac{\partial E_x}{\partial t} &= \frac{1}{\epsilon}\frac{\partial H_z}{\partial y}\\ \frac{\partial E_y}{\partial t} &= -\frac{1}{\epsilon}\frac{\partial H_z}{\partial x}\\ \frac{\partial H_z}{\partial t} &= -\frac{1}{\mu}\left(\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y}\right)\end{aligned}\right\}\ \text{TE}_z\text{-Mode}, \tag{2.14}$$

$$\left.\begin{aligned}\frac{\partial E_z}{\partial t} &= \frac{1}{\epsilon}\left(\frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y}\right)\\ \frac{\partial H_x}{\partial t} &= -\frac{1}{\mu}\frac{\partial E_z}{\partial y}\\ \frac{\partial H_y}{\partial t} &= \frac{1}{\mu}\frac{\partial E_z}{\partial x}\end{aligned}\right\}\ \text{TM}_z\text{-Mode}. \tag{2.15}$$

Offensichtlich können mit der TE_z- und TM_z-Mode zwei voneinander entkoppelte Systeme beschrieben werden, da keine gegenseitige Abhängigkeit durch eine gemeinsame Feldkomponente vorliegt. Dennoch sei angemerkt, dass unterschiedliche Auffassungen hinsichtlich der Definitionen der Moden üblich sind. Zum einen existiert die in der Numerik gebräuchliche Notation, wie sie in der Gleichung (2.14) und Gleichung (2.15) dargestellt wird [8, 9]. Zum anderen werden in der Leitungstheorie die beiden Moden jeweils entgegengesetzt definiert [41, 42]. Aufgrund der im Folgenden angewandten numerischen Methoden wird die erste Notation verwendet.

Schließlich kann der 3D-Fall unter Berücksichtigung aller Raumrichtungen formuliert werden, wodurch sich im Allgemeinen die Propagation im freien Raum beschreiben lässt:

$$\frac{\partial E_x}{\partial t} = \frac{1}{\epsilon}\left(\frac{\partial H_z}{\partial y} - \frac{\partial H_y}{\partial z}\right), \tag{2.16}$$

$$\frac{\partial H_x}{\partial t} = -\frac{1}{\mu}\left(\frac{\partial E_z}{\partial y} - \frac{\partial E_y}{\partial z}\right), \tag{2.17}$$

$$\frac{\partial E_y}{\partial t} = \frac{1}{\epsilon}\left(\frac{\partial H_x}{\partial z} - \frac{\partial H_z}{\partial x}\right), \tag{2.18}$$

$$\frac{\partial H_y}{\partial t} = -\frac{1}{\mu}\left(\frac{\partial E_x}{\partial z} - \frac{\partial E_z}{\partial x}\right), \tag{2.19}$$

$$\frac{\partial E_z}{\partial t} = \frac{1}{\epsilon}\left(\frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y}\right), \tag{2.20}$$

$$\frac{\partial H_z}{\partial t} = -\frac{1}{\mu}\left(\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y}\right). \tag{2.21}$$

Dieses System aus sechs gekoppelten PDGLs stellt die Grundlage für die numerische Methoden dar, auf die im weiteren Verlauf noch eingegangen wird. Insbesondere in der Numerik ist es entscheidend, das System effizient zu lösen, um den Rechenaufwand sowie die Fehlerrate zu minimieren.

2.2 Materialmodelle

Die Maxwell-Gleichungen beschreiben die grundlegende Wechselwirkung zwischen dem elektrischen und magnetischen Feld, wobei ein stark vereinfachtes Modell angenommen wird. Üblicherweise wird von einem homogenen isotropen Medium bzw. Vakuum ausgegangen. Dahingegen weichen die Permittivität und Permeabilität der Materialien in der Praxis von denen im Vakuum ab, was die Propagation der elektromagnetischen Welle signifikant beeinflusst. Diese Tatsache trifft ebenso auf physikalische Phänomene wie die Absorption und Dispersion zu, die nicht berücksichtigt werden. Ferner wird ein lineares Verhalten des Systems angenommen, sodass nichtlineare Effekte außer Acht gelassen werden. Folglich werden die im Rahmen dieser Arbeit verwendeten linearen und nichtlinearen Materialmodelle vorgestellt.

2.2.1 Lineare Materialmodelle

Die linearen Materialmodelle bilden die Grundlage zahlreicher ingenieurwissenschaftlicher Anwendungen. Sie bieten eine praxisnahe und numerisch effiziente Näherung zur Beschreibung materialabhängiger Effekte in technischen Simulationen. Typischerweise besteht dabei ein linearer Zusammenhang zwischen der äußeren Anregung und der resultierenden Materialantwort. In diesem Kontext wird bei der Anregung von einer elektrischen Einstrahlung ausgegangen, die eine Reaktion im Material in Form einer Polarisation hervorruft.

Drude-Modell

Das Drude-Modell ist ein fundamentales Materialmodell in der Elektrotechnik, das vom namensgebenden Physiker Drude in [43] publiziert wurde. Gängige Anwendungsgebiete des Drude-Modells sind die Plasmonik [44], die Realisierung photonischer Kristalle [45] und THz-Modulatoren [46]. Das Modell basiert auf der Annahme, dass sich die Elektronen in einem Metall wie freie ungeladene Teilchen durch das Kristallgitter des Metalls bewegen. Dabei erfahren die Elektronen zu unregelmäßigen Zeitpunkten elastische Stöße mit den Gitteratomen, die als Stöße mit Ionen modelliert werden, sobald ein externes elektrisches Feld $\vec{E}$ angelegt wird. Durch die Bewegungsgleichung

$$m_D \frac{\partial^2 \vec{x}(t)}{\partial t^2} + m_D \gamma_D \frac{\partial \vec{x}(t)}{\partial t} = -e\vec{E}(t) \tag{2.22}$$

mit der Masse m_D eines Leitungselektrons im Metall, der zeitabhängigen Verschiebung des Elektrons $\vec{x}(t)$ und der Elementarladung e lässt sich die Dynamik eines freien Elektrons in einem Metall bestimmen [47]. Wird zur Lösung der Gleichung (2.22) für die vektoriellen Größen jeweils der harmonische Ansatz verwendet respektive in den Frequenzbereich transformiert [48], so ergibt sich nach Umformung der Ausdruck

$$\vec{x}(t) = \frac{e}{m_D(\omega^2 + j\gamma_D\omega)}\vec{E}(t). \tag{2.23}$$

Mit der makroskopischen Polarisation

$$\vec{P}(t) = -n_D e\vec{x}(t) \tag{2.24}$$

und der Verwendung der Gleichung (2.23) kann der Polarisationsvektor als Funktion von $\vec{E}$ ausgedrückt werden:

$$\vec{P}(t) = -\frac{n_D e}{m_D(\omega + j\gamma_D\omega)}\vec{E}(t). \tag{2.25}$$

Durch das Einsetzen der Gleichung (2.25) in die Gleichung (2.5) lässt sich die folgende Beziehung für die elektrische Flussdichte im Metall erschließen:

$$\vec{D}(t) = \epsilon_0\vec{E}(t) + \vec{P}(t) = \epsilon_0\left(1 - \frac{{\omega_D}^2}{\omega + j\gamma_D\omega}\right)\vec{E}(t). \tag{2.26}$$

Die Plasmafrequenz des Elektronengases in der Gleichung (2.26) ist definiert durch

$${\omega_D}^2 = \frac{n_D e^2}{\epsilon_0 m_D}. \tag{2.27}$$

Unter Berücksichtigung der Gleichung (2.8) ist ersichtlich, dass die relative Permittivität in der Gleichung (2.26) wie folgt gegeben ist:

$$\epsilon_r(\omega) = 1 - \frac{{\omega_D}^2}{\omega + j\gamma_D\omega}. \tag{2.28}$$

Da die bisherige Herleitung für ein ideales Metall mit frei beweglichen Elektronen durchgeführt wurde, wird die Gleichung (2.28) zur Beschreibung eines realen Metalls ausgewertet. Im Falle des freien Elektronenmodells gilt $\omega \gg {\omega_D}^2$ und demnach $\epsilon_r(\omega) \to 1$. Im Gegensatz dazu gilt bei den Edelmetallen $\omega > {\omega_D}^2$, weshalb die Gleichung (2.28) modifiziert werden muss [49]:

$$\epsilon_r(\omega) = \epsilon_\infty - \frac{{\omega_D}^2}{\omega + j\gamma_D\omega}. \tag{2.29}$$

Der Parameter ϵ_∞ in der Gleichung (2.29) repräsentiert die relative Permittivitätskonstante für Frequenzen $\omega \to \infty$ und kann in technischen Handbüchern für spezifische Materialien gefunden werden. Klassischerweise wird dieser Wert auf $\epsilon_\infty \in [1, 10]$ beschränkt [47, 49].

Debye-Modell

Das Debye-Modell [50] des Physikers Debye gehört wie das Drude-Modell zu den etablierten Materialmodellen in der Elektrotechnik und findet Anwendung in der dielektrischen Spektroskopie [51], der Biomedizin [52] sowie der Sensorik [53]. Es beschreibt die Reaktion von Molekülen mit permanenten Dipolen auf ein statisches oder niederfrequentes elektrisches Feld, indem es ihre Polarisation durch eine Relaxationszeit τ modelliert. Die Relaxationszeit ist ein Maß für die Zeit, die ein Molekül oder Dipol benötigt, um auf eine Änderung des äußeren elektrischen Feldes zu reagieren und in den neuen Gleichgewichtszustand zu gelangen. Mathematisch betrachtet können für einen einzelnen Relaxationsprozess bis zu vier verschiedene Werte für die relative Permittivität auftreten, abhängig vom Ausmaß der Relaxation und den experimentell verfügbaren Frequenzen [54]. In Anlehnung an die Gleichung (2.29) wird der folgende Ansatz gewählt:

$$\epsilon_r(\omega) = \epsilon_\infty + \frac{\epsilon_s - \epsilon_\infty}{1 + j\omega\tau}. \tag{2.30}$$

Der Parameter ϵ_s kennzeichnet die relative Permittivität im Grenzfall $\omega \to 0$. Um die elektrische Leitfähigkeit

$$\sigma_s = \frac{\epsilon_0(\epsilon_\infty - \epsilon_s)}{\tau} \tag{2.31}$$

in die Berechnung mit einzubeziehen, wird die Gleichung (2.30) entsprechend erweitert [47]:

$$\epsilon_r(\omega) = \epsilon_\infty + \frac{\epsilon_s - \epsilon_\infty}{1 + j\omega\tau} + \frac{\sigma_s}{j\omega\epsilon_0}. \tag{2.32}$$

Analog zu ϵ_s wird bei σ_s in der Gleichung (2.32) der Grenzwert $\omega \to 0$ betrachtet. Die frei einstellbaren Parameter sind ϵ_s, ϵ_∞, τ und σ_s [44]. Beispiele für mit dem Debye-Modell berechnete Materialparameter werden in [47] angeführt.

2.2.2 Nichtlineare Materialmodelle

Mit Hilfe nichtlinearer Materialmodelle lassen sich Effekte wie der Kerr-Effekt [55] und die Raman-Streuung [56] beschreiben, die bei hohen Feldstärken oder anisotropen bzw. stark

dispersiven Materialien auftreten können. Diese sind speziell für die optische Telekommunikation von Bedeutung [57]. Zudem berücksichtigen die nichtlinearen Modelle im Gegensatz zu den linearen Modellen das Materialverhalten in Abhängigkeit von der Amplitude, Frequenz und Orientierung des äußeren Feldes, wie sie insbesondere bei Laseranwendungen vorkommen [6]. Dementsprechend werden im Anschluss das Zwei- und Mehr-Niveau-Modell erörtert.

Zwei-Niveau-Modell

Das Zwei-Niveau-Modell erlaubt es in erster Näherung ein Lasersystem zu modellieren, wenn auch stark vereinfacht. Der Ausgangspunkt ist der diskrete Grundzustand bzw. das *Niveau 1*, in dem sich die Elektronen zunächst frei von äußeren Einflüssen befinden. Damit jene Elektronen zur Emission von monochromatischem, kohärentem und gerichtetem Licht beitragen, müssen sie in den diskreten angeregten Zustand bzw. das *Niveau 2* versetzt werden. In Gas- und Festkörperlasern erfolgt die Anregung in der Regel durch optisches Pumpen mittels eines hochenergetischen externen Lichtstrahls [58]. Dagegen kann bei einer Laserdiode die Anregung mittels eines elektrischen Stroms erzielt werden [59], während bei einem Femtosekundenlaser die Kerr-Anregung ausgenutzt wird [60]. Für den Laserbetrieb ist es von entscheidender Bedeutung, dass mehr Elektronen im angeregten Zustand als im Grundzustand vorhanden sind, was als Besetzungsinversion bezeichnet wird. Hierbei können durch stimulierte Emission mehr Photonen erzeugt werden als durch spontane Emission verloren gehen. Schließlich kann der Laserstrahl in einem Resonator weiter verstärkt werden [61].

Wird eine Verstärkung im aktiven Medium des Laser-Systems angestrebt, kann diese durch das Einbinden der makroskopischen Polarisation in die Maxwell-Gleichung erreicht werden. Diesbezüglich wurde in [62] gezeigt, dass die zeitliche Entwicklung der Polarisation über den harmonischen Lorentz-Oszillator modelliert werden kann, bedingt durch die Kopplung zwischen der Besetzungsinversion und dem externen elektrischen Feld [63]. Um die Verstärkung des Systems vorweg mit einer Sättigung zu versehen, wird der Ansatz aus [64] gewählt:

$$\frac{\partial^2 \vec{P}_m}{\partial t^2} + \Gamma_m \frac{\partial \vec{P}_m}{\partial t} + \left({\omega_m}^2 + \frac{{\Gamma_m}^2}{4} \right) \vec{P}_m = \Delta \vec{N} \sigma_\mathrm{m} \vec{E}. \tag{2.33}$$

In der Gleichung (2.33) repräsentieren $\vec{P}_m$ die nichtlineare Polarisationsdichte und ω_m die Übergangsfrequenz des m-ten elektronischen Übergangs. Mit Γ_m wird die Full Width at Half Maximum (FWHM) beschrieben und mit σ_m die Kopplung zwischen dem elektrischen Feld und der nichtlinearen Polarisation. Die Besetzungsinversion infolge der Polarisation wird mit $\Delta \vec{N} = \vec{N_1} - \vec{N_2}$ angegeben, dabei stellt $\vec{N_1}$ die Besetzungsdichte in *Niveau 1* und $\vec{N_2}$ die Besetzungsdichte in *Niveau 2* dar. Bei $\Delta \vec{N} > 0$ verstärkt das Medium, während es bei $\Delta \vec{N} < 0$ absorbiert [65]. Die Ratengleichungen für die jeweiligen Niveaus sind wie folgt definiert [64]:

$$\frac{\partial \vec{N_1}}{\partial t} = -\frac{1}{\hbar \omega_m} \vec{E} \left(\frac{\partial \vec{P}_m}{\partial t} + \Gamma_m \frac{\vec{P}_m}{2} \right) - \gamma_{12} \vec{N_1} + \gamma_{21} \vec{N_2}, \tag{2.34}$$

$$\frac{\partial \vec{N_2}}{\partial t} = \frac{1}{\hbar \omega_m} \vec{E} \left(\frac{\partial \vec{P}_m}{\partial t} + \Gamma_m \frac{\vec{P}_m}{2} \right) + \gamma_{12} \vec{N_1} - \gamma_{21} \vec{N_2}. \tag{2.35}$$

Der Term $\pm \frac{1}{\hbar \omega_m} \vec{E} \frac{\partial \vec{P}_m}{\partial t}$ in der Gleichung (2.34) und Gleichung (2.35) stellt die stimulierte Emissionsrate dar, die von der reduzierten Planck-Konstante $\hbar$ abhängt. Die Übergangsrate

vom *Niveau 1* zum *Niveau 2* wird mit γ_{12} bezeichnet, die entgegengesetzte Übergangsrate entsprechend mit γ_{21}.

Mehr-Niveau-Modell

Durch die Erweiterung des Zwei-Niveau-Modells auf ein Mehr-Niveau-Modell können die Besetzungsinversionen zwischen den Zuständen bzw. die Polarisation im aktiven Medium noch präziser und realistischer abgebildet werden [66]. Analog zum Zwei-Niveau-Modell bildet die Gleichung (2.33) die Grundlage für die Berechnung gesättigter nichtlinearer Medien. Ferner können die Ratengleichungen zwischen den jeweiligen Zuständen so formuliert werden, dass sie in der Theorie für eine beliebige Anzahl an Zuständen gelten [64]:

$$\frac{\partial N_i}{\partial t} = -\sum_{\mathsf{x}} \gamma_{i\mathsf{x}} N_i + \sum_{\mathsf{x}} \gamma_{\mathsf{x}i} N_x + \sum_m \Xi_{i,m} \left[\frac{1}{\omega_m \hbar} \vec{E} \left(\frac{\partial}{\partial t} + \frac{\Gamma_m}{2} \right) \vec{P}_m \right] \tag{2.36}$$

Die Hilfsfunktion $\Xi \in [-1, 0, 1]$ in der Gleichung (2.36) ermöglicht es, während des Übergangs m das elektrische Feld und die Polarisation dynamisch in die Betrachtung einzubeziehen. Entspricht i dem oberen Zustand beim Übergang m, so gilt $\Xi = 1$. Im inversen Fall ergibt sich $\Xi = -1$, andernfalls ist $\Xi = 0$.

Dennoch sei darauf hingewiesen, dass mit zunehmender Anzahl der Zustände die Komplexität der Approximation deutlich ansteigt. Das Hauptziel der Arbeit ist es, die Rechenzeit bei zugleich ausreichender Genauigkeit zu minimieren. Aus diesem Grund wird das Mehr-Niveau-Modell nur der Vollständigkeit halber erläutert.

Kapitel 3

Numerische Grundlagen und Modelle

Damit rechnergestützte Simulationen kontinuierliche Prozesse der realen Welt nachbilden können, ist eine Digitalisierung durch Diskretisierung unabdingbar. Andernfalls wären die Daten jener Systeme mit einem Personal Computer (PC) respektive digitalen Messgeräten nicht zu verarbeiten, da unendlich viel Speicher vonnöten wäre. Trotz der stetig wachsenden Rechenleistung der PCs gilt es, die Problemstellung ressourceneffizient anzunähern. Im Mittelpunkt stehen hierbei der Ort und die Zeit als Parameter, die im Folgenden getrennt voneinander betrachtet werden. Den Auftakt bildet die Ortsdiskretisierung nach Yee. Ergänzend wird die Operatornotation eingeführt, um die relevanten Größen übersichtlich darzustellen. Darüber hinaus werden Implementierungsansätze vorgestellt, die der Speicheroptimierung dienen.

3.1 Diskretisierung des Ortes nach Yee

Im Laufe des 20. Jahrhunderts wurden etliche Verfahren entwickelt, um die Maxwell-Gleichungen im Zeitbereich lösen zu können. Je nach Geometrie ist eine analytische Lösung realisierbar, jedoch werden dafür starke Vereinfachungen vorausgesetzt [5]. Infolgedessen haben sich numerische *Solver* etabliert. In diesem Zusammenhang ist das FDTD-Verfahren am populärsten, das von Kane S. Yee in [10] publiziert wurde und in gleicher Weise als Yee-Methode bekannt ist. Bemerkenswerterweise fand die Yee-Methode erst mehr als ein Jahrzehnt nach ihrer Veröffentlichung breite Anerkennung [67].

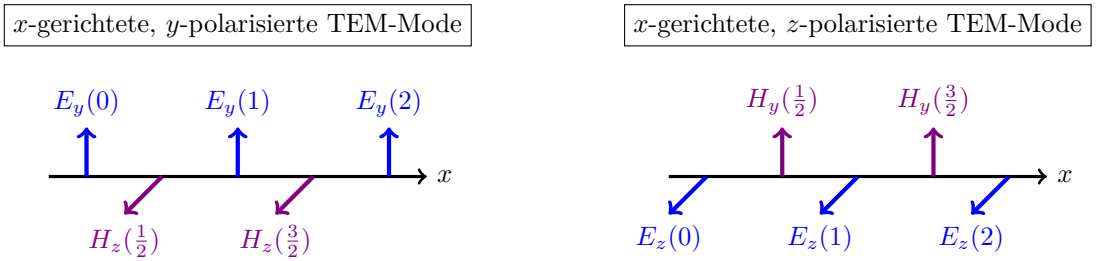


Abbildung 3.1: 1D Yee-Gitter für die x-gerichtete TEM-Mode mit unterschiedlicher Polarisierung.

Zu Beginn der Ortsdiskretisierung nach Yee findet eine räumliche Aufteilung der $\vec{E}$- und $\vec{H}$-Feldkomponenten statt, sodass im Allgemeinen quaderförmige Zellen zwei sogenannte duale Gitter bilden. Wenn Δx, Δy und Δz die örtlichen Diskretisierungsweiten in den jeweiligen Raumrichtungen repräsentieren, so sind die Gitter um $\Delta x/2$, $\Delta y/2$ sowie $\Delta z/2$ zueinander versetzt ausgerichtet. Die tatsächlichen Diskretisierungspunkte befinden sich in der Mitte einer Kante.

Basierend auf den im Abschnitt 2.1 beschriebenen Zusammenhängen wird die Ortsdiskretisierung jeweils an einer 1D-, 2D- und 3D-Geometrie veranschaulicht. Das Kernstück zur Approximation der örtlichen Ableitung ist der zentrale Differenzenquotient [68], der in jeder Geometrie an einem Diskretisierungspunkt pro Raumrichtung vorgeführt wird. Generell wäre die Verwendung einer Vorwärts- oder Rückwärtsdifferenz legitim, dennoch ist bekanntermaßen die Fehlerordnung mit $\mathcal{O}(\Delta x)$ bei beiden Differenzen höher als jene der zentralen Differenz mit $\mathcal{O}(\Delta x^2)$. Nur in Ausnahmefällen, wie etwa bei der Berücksichtigung von Randbedingungen, kann der zentrale Differenzenquotient aufgrund der zwei benötigten Funktionswerte der Nachbarpunkte weniger effizient sein als die einfacheren Vorwärts- oder Rückwärtsdifferenzen [69]. Weiterhin werden die Diskretisierungspunkte im kartesischen Koordinatensystem über ihre Koordinaten (x,y,z) positioniert. Der Lesbarkeit halber werden nur so viele Koordinaten angegeben, wie es der Dimension des betrachteten Systems entspricht.

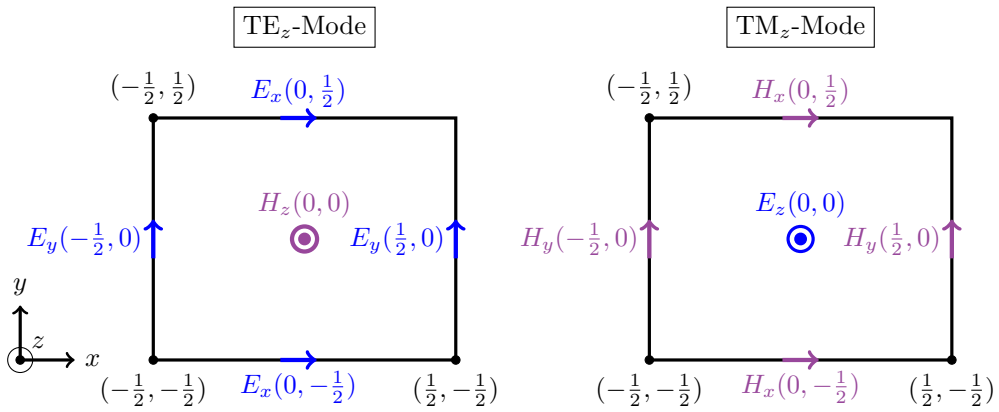


Abbildung 3.2: 2D Yee-Gitter für die TE$_z$- und TM$_z$-Mode.

Hinsichtlich des 1D-Yee-Gitters ist zwischen der y- und z-Polarisierung zu differenzieren, wenn in x-Richtung propagiert wird. Die Anordnung der $\vec{E}$- und $\vec{H}$-Komponenten für die TEM-Mode ist in beiden Fällen in der Abbildung 3.1 visualisiert. Gemäß der Dimension geschieht die Kopplung nur über die beiden benachbarten Diskretisierungspunkte in x-Richtung. Folglich ergeben sich auf Basis der Gleichung (2.12) und Gleichung (2.13) die Beziehungen

$$\left.\begin{aligned} \frac{\partial E_y(1)}{\partial t} &\approx -\frac{1}{\mu}\frac{H_z(\frac{3}{2}) - H_z(\frac{1}{2})}{\Delta x} \\ \frac{\partial H_z(\frac{1}{2})}{\partial t} &\approx -\frac{1}{\epsilon}\frac{E_y(1) - E_y(0)}{\Delta x} \end{aligned}\right\} \begin{array}{l} x\text{-gerichtete,} \\ y\text{-polarisierte} \\ \text{TEM-Mode,} \end{array} \tag{3.1}$$

$$\left.\begin{aligned} \frac{\partial E_z(1)}{\partial t} &\approx -\frac{1}{\mu}\frac{H_y(\frac{3}{2}) - H_y(\frac{1}{2})}{\Delta x} \\ \frac{\partial H_y(\frac{1}{2})}{\partial t} &\approx -\frac{1}{\epsilon}\frac{E_z(1) - E_z(0)}{\Delta x} \end{aligned}\right\} \begin{array}{l} x\text{-gerichtete,} \\ z\text{-polarisierte} \\ \text{TEM-Mode.} \end{array} \tag{3.2}$$

Die Erweiterung auf ein 2D-Yee-Gitter ist in der Abbildung 3.2 gezeigt. Die Yee-Zellen sind so ausgelegt, dass sich im Zentrum bei der TE-Mode die H_z-Komponente und bei der TM-Mode die E_z-Komponente befinden. Zusätzlich wird die zentrale Position eines Diskretisierungspunktes entlang an einer Kante deutlich. Durch die räumliche Anordnung der Komponenten zueinander ist die Umsetzung des Induktionsgesetzes aus der Gleichung (2.3) und des Durchflutungsgesetzes aus der Gleichung (2.4) gewährleistet, indem jede $\vec{H}$-Feldkomponente von vier $\vec{E}$-Feldkomponenten umwirbelt wird und umgekehrt [70].

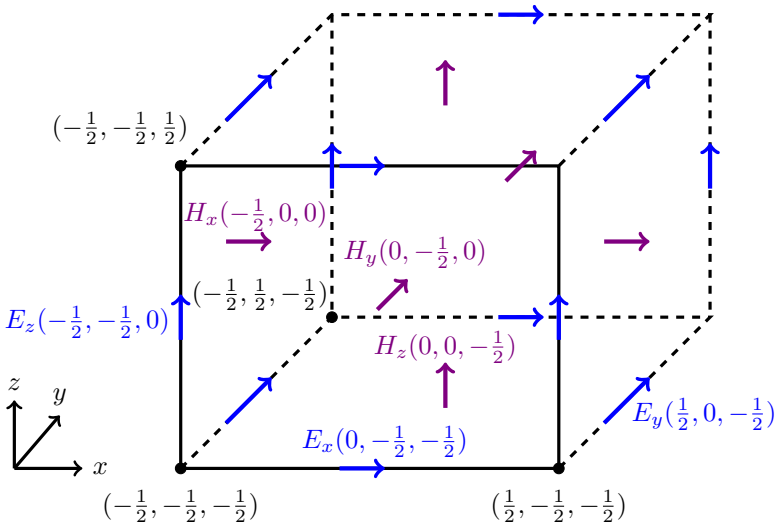


Abbildung 3.3: 3D Yee-Gitter für die TEM-Mode.

Im Gegensatz zur vorherigen Betrachtung erfolgt die Kopplung in den Raumrichtungen x und y. In Übereinstimmung mit der Gleichung (2.14) und Gleichung (2.15) lassen sich die folgenden Relationen für die TE- und TM-Mode schlussfolgern:

$$\left.\begin{aligned}
\frac{\partial E_x(0,-\frac{1}{2})}{\partial t} &\approx \frac{1}{\epsilon}\frac{H_z(0,0)-H_z(0,-1)}{\Delta y}\\
\frac{\partial E_y(-\frac{1}{2},0)}{\partial t} &\approx -\frac{1}{\epsilon}\frac{H_z(0,0)-H_z(-1,0)}{\Delta x}\\
\frac{\partial H_z(0,0)}{\partial t} &\approx -\frac{1}{\mu}\left(\frac{E_y(\frac{1}{2},0)-E_y(-\frac{1}{2},0)}{\Delta x}-\frac{E_x(0,\frac{1}{2})-E_x(0,-\frac{1}{2})}{\Delta y}\right)
\end{aligned}\right\}\ \text{TE}_z\text{-Mode}, \tag{3.3}$$

$$\left.\begin{aligned}
\frac{\partial E_z(0,0)}{\partial t} &\approx \frac{1}{\epsilon}\left(\frac{H_y(\frac{1}{2},0)-H_y(-\frac{1}{2},0)}{\Delta x}-\frac{H_x(0,\frac{1}{2})-H_x(0,-\frac{1}{2})}{\Delta y}\right)\\
\frac{\partial H_x(0,-\frac{1}{2})}{\partial t} &\approx -\frac{1}{\mu}\frac{E_z(0,0)-E_z(0,-1)}{\Delta y}\\
\frac{\partial H_y(-\frac{1}{2},0)}{\partial t} &\approx \frac{1}{\mu}\frac{E_z(0,0)-E_z(-1,0)}{\Delta x}
\end{aligned}\right\}\ \text{TM}_z\text{-Mode}. \tag{3.4}$$

Wie bereits erwähnt, werden standardmäßig quaderförmige Yee-Zellen angenommen. Für den Sonderfall, dass äquidistante Abstände mit $\Delta x = \Delta y = \Delta z$ für die Abtastung entlang jeder Raumrichtung gewählt werden, entspricht das Yee-Gitter im 3D-Fall einem Würfel, wie die Abbildung 3.3 verdeutlicht. Darüber hinaus sind nun sechs Feldkomponenten miteinander gekoppelt, was insbesondere hinsichtlich des Rechenaufwands einen erheblichen Einfluss hat. Das Wirbelfeld umschließt jede Feldkomponente mit den zwei Nachbarpunkten in x-, y- und z-Richtung. Weiter gilt es zu beachten, dass die Zellen in Bezug auf die $\vec{H}$-Feldkomponenten

flächen- und nicht raumzentriert ausgelegt sind. Äquivalent zum 2D-Yee-Gitter sind die $\vec{E}$-Feldkomponenten an den Kanten vorzufinden. Ausgehend vom kontinuierlichen PDGL-System im Abschnitt 2.1 ergibt sich das diskrete Differentialgleichung (DGL)-System, was für eine 3D-Geometrie folgende Gestalt annimmt:

$$\frac{\partial E_x(0,-\frac{1}{2},-\frac{1}{2})}{\partial t} \approx \frac{1}{\epsilon}\left(\frac{H_z(0,0,-\frac{1}{2}) - H_z(0,-1,-\frac{1}{2})}{\Delta y} - \frac{H_y(0,-\frac{1}{2},0) - H_y(0,-\frac{1}{2},-1)}{\Delta z}\right), \tag{3.5}$$

$$\frac{\partial E_y(\frac{1}{2},0,-\frac{1}{2})}{\partial t} \approx \frac{1}{\epsilon}\left(\frac{H_x(\frac{1}{2},0,0) - H_x(\frac{1}{2},0,-1)}{\Delta z} - \frac{H_z(1,0,-\frac{1}{2}) - H_z(0,0,-\frac{1}{2})}{\Delta x}\right), \tag{3.6}$$

$$\frac{\partial E_z(-\frac{1}{2},-\frac{1}{2},0)}{\partial t} \approx \frac{1}{\epsilon}\left(\frac{H_y(0,-\frac{1}{2},0) - H_y(-1,-\frac{1}{2},0)}{\Delta x} - \frac{H_x(-\frac{1}{2},0,0) - H_x(-\frac{1}{2},-1,0)}{\Delta y}\right), \tag{3.7}$$

$$\frac{\partial H_x(-\frac{1}{2},0,0)}{\partial t} \approx -\frac{1}{\mu}\left(\frac{E_z(-\frac{1}{2},\frac{1}{2},0) - E_z(-\frac{1}{2},-\frac{1}{2},0)}{\Delta y} - \frac{E_y(-\frac{1}{2},0,\frac{1}{2}) - E_y(-\frac{1}{2},0,-\frac{1}{2})}{\Delta z}\right), \tag{3.8}$$

$$\frac{\partial H_y(0,-\frac{1}{2},0)}{\partial t} \approx -\frac{1}{\mu}\left(\frac{E_x(0,-\frac{1}{2},\frac{1}{2}) - E_x(0,-\frac{1}{2},-\frac{1}{2})}{\Delta z} - \frac{E_z(\frac{1}{2},-\frac{1}{2},0) - E_z(-\frac{1}{2},-\frac{1}{2},0)}{\Delta x}\right), \tag{3.9}$$

$$\frac{\partial H_z(0,0,-\frac{1}{2})}{\partial t} \approx -\frac{1}{\mu}\left(\frac{E_y(\frac{1}{2},0,-\frac{1}{2}) - E_y(-\frac{1}{2},0,-\frac{1}{2})}{\Delta x} - \frac{E_x(0,\frac{1}{2},-\frac{1}{2}) - E_x(0,-\frac{1}{2},-\frac{1}{2})}{\Delta y}\right). \tag{3.10}$$

3.1.1 Eigenschaften der Yee-Diskretisierung

In diesem Abschnitt werden die positiven sowie negativen Eigenschaften der Ortsdiskretisierung nach Yee erörtert. So kann im Vorfeld abgeschätzt werden, ob sich die FDTD-Methode zur Simulation des physikalischen Prozesses eignet.

Vorteile der Yee-Diskretisierung

Der vermeintlich wichtigste Vorteil der FDTD-Methode ist ihre vergleichsweise einfache Implementierung, da lediglich ein Verständnis der Maxwell-Gleichungen und des zentralen Differenzenquotienten erforderlich ist. Des Weiteren kann die Propagation unmittelbar nachvollzogen werden, da dies über die Zeit geschieht [9]. Diesbezüglich sind rechenintensive Operationen, wie das Lösen von Gleichungssystemen, die Berechnung der Inversen einer Matrix oder die Bestimmung der Eigenwerte, aufgrund der expliziten Natur der FDTD-Methode nicht notwendig [8]. Stattdessen genügen reine MVMs, die wesentlich schneller auszuwerten sind. Hinzu kommt, dass die PCs heutzutage erheblich leistungsstärker sind als noch vor Jahrzehnten, wodurch komplexe Systeme mit sehr feiner Diskretisierung fast keine Herausforderung mehr darstellen. Insbesondere kann der Zugriff auf HPCs in Kombination mit einer simplen, realisierbaren Parallelisierung eine deutliche Effizienzsteigerung bei der Berechnung ermöglichen [71]. Hierbei profitiert der Prozessablauf von den üblicherweise dünnbesetzten Matrizen, welche die Speicherverwaltung optimieren [72].

Weiterhin wird die FDTD-Methode in vielen Einsatzgebieten angewandt. Dazu zählen neben der Elektrodynamik beispielsweise die Biomedizin [73], Geophysik [74], Kommunikationstechnik [75], Akustik [76], Optik [77] oder Quantenphysik [78]. Als Maßstab für den Nutzen der FDTD-Methode kann die enorme Anzahl an Publikationen in renommierten Journalen und Fachbücher genommen werden, die kontinuierlich steigt. Ebenso werden *Software-Tools* kontinuierlich entwickelt und optimiert, sowohl auf *open-source* als auch auf kommerzieller Basis [79]. Darüber hinaus können FDTD-Modelle auf diverse Systeme angepasst werden, wodurch keine Neuformulierung vonnöten ist. So können beliebige Arten von Medien, seien es inhomogene, verlustbehaftete, nichtlineare, anisotrope oder frequenzabhängig dispersive mit der FDTD-Approximation der Maxwell-Gleichungen modelliert werden [38].

Nachteile der Yee-Diskretisierung

Obwohl die Yee-Methode etliche Vorteile zu bieten hat, stehen dem essentielle Nachteile gegenüber. Trotz des Ausnutzens von dünnbesetzten Matrizen ist der benötigte Speicherbedarf für die Ortsdiskretisierung nach Yee nicht zu unterschätzen. Wird eine sehr hohe Auflösung angestrebt und zusätzlich mit Materialmodellen sowie mit Randbedingungen simuliert, führt dies zu einer starken Restriktion der Implementierung. Ferner ist die Dimension der Geometrie von wesentlicher Bedeutung, weil pro Dimension jeweils zwei, vier oder sechs Koppelterme pro Diskretisierungspunkt erforderlich sind.

Prinzipiell gilt die FDTD-Methode als stabil, sofern die beteiligten Parameter sorgfältig aufeinander abgestimmt sind. In diesem Kontext besagt die CFL-Bedingung [11], dass die elektromagnetische Welle pro Zeitschritt Δt höchstens den Abstand einer Gitterzelle zurücklegen darf. Somit gilt

$$\Delta t \leq \Delta t_{CFL} = \begin{cases} \left(c \cdot \sqrt{\frac{1}{\Delta x^2}} \right)^{-1} & \text{, 1D-Fall} \\ \left(c \cdot \sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}} \right)^{-1} & \text{, 2D-Fall} \\ \left(c \cdot \sqrt{\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2} + \frac{1}{\Delta z^2}} \right)^{-1} & \text{, 3D-Fall,} \end{cases} \tag{3.11}$$

wobei $c = \frac{1}{\sqrt{\epsilon\mu}}$ die Lichtgeschwindigkeit im zugrunde liegenden homogenen Medium angibt [9]. Durch die Gleichung (3.11) ist die Kopplung zwischen der örtlichen und zeitlichen Auflösung ersichtlich. Je höher die örtliche Auflösung gewählt wird, desto kleiner fällt die maximale Zeitschrittweite Δt_{CFL} aus, ohne eine Instabilität des Systems zu verursachen. In der Praxis wird oft beim 1D-Fall die Zeitschrittweite $\Delta t = \Delta t_{CFL}$ gewählt [9]. Dagegen treten beim 2D- und 3D-Fall unvermeidbare numerische Fehler durch die Diskretisierung auf, weshalb dort aus Stabilitätsgründen maximal $\Delta t = 0.99\Delta t_{CFL}$ vorgeschlagen wird [38, 80].

Ein weiterer Nachteil der FDTD-Methode ist die numerische Dispersion. In diesem Zusammenhang kann es vorkommen, dass sich die elektromagnetische Welle in den verschiedenen Raumrichtungen mit unterschiedlichen Geschwindigkeiten ausbreitet, was zu numerischen Verzerrungen führen kann. Die Ursache liegt in der Frequenzabhängigkeit der numerischen Phasengeschwindigkeit, wodurch die elektromagnetische Welle dispersiv wird. Die Abweichung zwischen der numerischen und analytischen Phasengeschwindigkeit hängt besonders von der örtlichen Abtastung ab. Im schlimmsten Fall kann eine Veränderung der Wellenform bedingt durch

das Ausschmieren der Lösung festgestellt werden [81]. Um dem entgegenzuwirken, wird in der Literatur empfohlen die Diskretisierungsweite auf $\Delta \in [\frac{\lambda}{10}, \frac{\lambda}{20}]$ zu begrenzen [9, 70]. Weitere Maßnahmen zur Minderung der numerischen Dispersion sind die Erhöhung der Zeitschrittweite [38] oder der örtlichen Auflösung [9], was jedoch zu Lasten der Genauigkeit bzw. des Rechenaufwands geht.

Konventionell wird eine reale Struktur bei der FDTD-Methode durch ein homogen diskretisiertes Rechengebiet abgebildet. Zwar wird die Implementierung hierdurch bedeutend vereinfacht, jedoch wirkt sich dies insbesondere bei kurvenförmigen Geometrien und schrägen Kanten nachteilig aus. Dadurch ergeben sich an den Grenzschichten *stair-cased* Artefakte, welche die geforderte lokale örtliche Auflösung negativ beeinflussen. Für derartige Systeme müssen Erweiterungen der FDTD-Methode oder alternative Verfahren in Betracht gezogen werden [8], die im weiteren Verlauf näher erläutert werden.

3.1.2 Alternativen zur Ortsdiskretisierung nach Yee

Neben der FDTD-Methode existieren noch weitere Verfahren, die im Zeitbereich zur Approximation der Maxwell-Gleichungen eingesetzt werden könnten. Zu den bekanntesten Alternativen zählen die Finite-Element-Method (FEM) und die Finite-Volume-Method (FVM). Dass im Rahmen dieser Arbeit dennoch der FDTD-Methode der Vorzug gegeben wird, ergibt sich aus den Einschränkungen der genannten Alternativen.

Finite-Element-Method

Die FEM hat ihren Ursprung in den 1940er-Jahren und entstammt den kontinuierlichen Galerkin-Methoden [82]. In den darauffolgenden Jahrzehnten wurde sie jedoch kontinuierlich weiterentwickelt und nahm schließlich die heute gebräuchliche Form an [83]. Wie bei der FDTD-Methode wird das Rechengebiet in finite Elemente unterteilt. Diese sind jedoch hinsichtlich ihrer Form deutlich flexibler als die klassischen orthogonalen Yee-Zellen, auch wenn für die FDTD-Methode alternative Zellformen existieren [9]. In Abhängigkeit der nachzubildenden Geometrie können Linien-, Flächen- oder Volumenelemente für die Ortsdiskretisierung genutzt werden. Generell erfolgt dies über Polygone. Bei den Flächenelementen kann auf Drei- oder Vierecke zurückgegriffen werden, bei den Volumenelementen auf Tetraeder, Pentaeder oder Hexaeder [70]. In Anbetracht der flexiblen Elemente bietet sich die FEM besonders für komplexe Geometrien an.

Im Gegensatz zur FDTD-Methode eignet sich die FEM bei der Approximation elektromagnetischer Probleme ausschließlich für die Ortsdiskretisierung, sodass eine separate Methode zur Zeitintegration erforderlich ist [8]. Je nach Anwendungsfall kommen dabei explizite oder implizite Verfahren zum Einsatz, die jeweils unterschiedliche Stärken und Schwächen aufweisen. Implizite Verfahren gelten als bedingungslos stabil und ermöglichen vergleichsweise große Zeitschrittweiten, sind aber sehr speicherintensiv und reagieren empfindlich auf starke Nichtlinearitäten im System. Explizite Verfahren zeichnen sich durch eine vergleichsweise geringe Komplexität aus, die in etwa linear mit der Systemgröße skaliert. Zudem sind sie vorteilhaft im Hinblick auf die Fehlersuche bei der Implementierung, allerdings sind sie nur bedingt stabil und erlauben nur kleine Zeitschrittweiten [84]. Darüber hinaus ist die Einbindung der Perfectly Matched Layer (PML) in die FEM bei der Modellierung dispersiver und anisotroper Medien im Zeitbereich mit hohen Speicheranforderungen verbunden [85]. Um die PML effizienter mit der FEM zu

kombinieren, wurden im Laufe der Zeit u.a. der Nutzen sphärischer Koordinaten [86] und die *displacement-based* FEM [87] untersucht.

Finite-Volume-Method

Die FVM gehört mit der FEM und der FDTD-Methode zu den bekanntesten numerischen Verfahren zur Lösung PDGLs in den Ingenieur- und Naturwissenschaften. Die Entwicklung der FVM geht auf die 1950er-Jahre zurück [88], wobei erst einige Jahrzehnte später die Approximation der Maxwell-Gleichungen in den Vordergrund rückte [89, 90]. Mit ihr lässt sich die inhärente Starrheit der FDTD-Methode überwinden, indem anstelle fester Gitterzellen finite Volumina zur Diskretisierung des Rechengebiets verwendet werden und demnach zu einer nichtuniformen Struktur führen [91]. Dies in Kombination mit der expliziten Evaluierung im Zeitbereich macht die FVM besonders attraktiv für die Untersuchung elektromagnetischer Phänomene [92]. Um die Komplexität moderat zu halten, werden standardmäßig 2D-Geometrien mit Dreiecken und 3D-Geometrien mit Tetraedern diskretisiert, jedoch ist die Wahl der Volumina frei [8].

Im Vergleich zu den anderen betrachteten numerischen Verfahren erfolgt bei der FVM die Anordnung der $\vec{E}$- und $\vec{H}$-Feldkomponenten in der Regel nebeneinander anstatt versetzt. Dies ermöglicht eine simultane Berechnung der gekoppelten Feldkomponenten [9]. Weil die FVM auf dem Prinzip der Erhaltung [93] beruht, basiert ihre Anwendung auf der Integralform der Maxwell-Gleichungen [94]. Unter der Annahme, dass die nicht orthogonalen Gitter infolge der Anordnung der finiten Volumina über ein lokales, krummliniges Koordinatensystem beschrieben werden können, lässt sich das nicht orthogonale Gitter mittels einer lokalen Abbildung in ein kartesisches Koordinatensystem überführen [9]. Durch diese Maßnahme kann die Berechnung vereinfacht sowie die numerische Stabilität und Effizienz erhöht werden.

Bedingt durch die Flexibilität der finiten Volumina stellen die Grenzen des Rechengebiets keine Herausforderung für die FVM dar. Diesbezüglich kann die Begrenzung je nach Randbedingung unterschiedlich definiert werden. Im Falle von Absorbing Boundary Conditions (ABC) kann der Rand so festgelegt werden, dass die elektromagnetische Welle nahezu senkrecht auftrifft [92]. Werden dagegen beispielsweise die PML als Grenzschichten implementiert, kann die Terminierung des Rechengebiets sphärisch oder ellipsoidal durchgeführt werden im Hinblick auf die Reduktion der Komplexität [95].

Trotz der beschriebenen Vorteile kann die nichtuniforme Struktur die Recheneffizienz und die räumliche Konvergenz der FVM ungünstig beeinträchtigen [92]. Im Verhältnis zur FDTD-Methode ist bei der FVM eine höhere Komplexität pro Zeitschritt zu erwarten aufgrund der unterschiedlichen Größen der finiten Volumina und der erhöhten Anzahl an Rechenoperationen, obwohl eine explizite Auswertung vorliegt und nur die Verrechnung mit einer Diagonalmatrix notwendig ist [9]. Bezogen auf die räumliche Konvergenz müssen nach [92] die *Gitterfeinheit*, *Gitterqualität* und *Gitterinhomogenität* aufeinander abgestimmt sein, um die angestrebte Genauigkeit zu erzielen. Bei der *Gitterfeinheit* muss die Diskretisierung für die höchste relevante Frequenz deutlich kleiner als die Wellenlänge gewählt werden, da eine zu grobe Auflösung besonders bei resonanten Strukturen zu numerischer Dissipation führen kann. Die *Gitterqualität* lässt sich z.B. über Kanten- oder Volumen-Oberflächen-Verhältnisse bewerten, wobei idealerweise regelmäßige Tetraeder erwünscht sind, da flache oder schräge Elemente numerisches Rauschen und im Extremfall Instabilitäten verursachen können. Im Kontext der *Gitterinhomogenität* kann in einem ausschließlich aus Tetraedern bestehenden Gitter der direkte Übergang von fein zu

grob diskretisierten Bereichen zwar die Anzahl der finiten Volumina reduzieren, allerdings wird bei einem zu abrupten Übergang die Genauigkeit der Approximation signifikant beeinflusst.

3.2 Operatornotation, Zustandsvektor und Systemmatrix

Wie bereits in Kapitel 2 erläutert, kann die Propagation einer elektromagnetischen Welle über die Maxwell-Gleichungen beschrieben werden. Soll dies mittels eines Materialmodells geschehen und es sollen zusätzlich Randbedingungen eingebunden werden, steigt die Anzahl der notwendigen Parametern signifikant an. Um dennoch eine übersichtliche Darstellung zu gewährleisten, wird die Operatornotation eingeführt. Vor diesem Hintergrund werden die $\vec{E}$- und $\vec{H}$-Feldkomponenten in Abhängigkeit des Ortsvektors $\vec{r}$ sowie der Zeit t im Zustandsvektor $\vec{\Psi}$ zusammengefasst:

$$\vec{\Psi}(\vec{r},t) = [E_x(t), E_y(t), E_z(t), H_x(t), H_y(t), H_z(t)]^T = [\vec{E}(\vec{r},t), \vec{H}(\vec{r},t)]^T. \tag{3.12}$$

Mit Hilfe einer örtlichen Diskretisierung kann der Zustandsvektor weiter angepasst werden, sodass er nur noch von der Zeit abhängt, im Sinne von $\vec{\Psi}(\vec{r},t) \rightarrow \vec{\Psi}(t)$. Die räumliche Abhängigkeit entfällt somit durch die Diskretisierung. Generell besteht der Vorteil der Notation aus der Gleichung (3.12) darin, dass alle relevanten Parameter bezüglich der Maxwell-Gleichungen, Materialmodelle und Randbedingungen direkt in $\vec{\Psi}$ inkludiert werden können. Dazu werden diese an das Ende des Zustandsvektors angehängt. Ferner gilt es, die lineare DGL

$$\frac{\partial \vec{\Psi}(t)}{\partial t} = f(\mathcal{H}) \cdot \vec{\Psi}(t). \tag{3.13}$$

zur Bestimmung der zeitabhängigen Ausbreitung der elektromagnetischen Welle zu lösen. In der Gleichung (3.13) kennzeichnet $f(\mathcal{H})$ den linearen Matrixoperator, der von der Systemmatrix $\mathcal{H}$ abhängt. Wird die Ortsdiskretisierung nach Yee gemäß dem Abschnitt 3.1 durchgeführt, so ergibt sich entsprechend der Definition in der Gleichung (3.13) die Beziehung $f(\mathcal{H}) = \mathcal{H}$ [15]:

$$\frac{\partial \vec{\Psi}(t)}{\partial t} = \mathcal{H} \cdot \vec{\Psi}(t). \tag{3.14}$$

Es ist evident, dass die Auswertung der Gleichung (3.14) nur noch auf einer MVM basiert. Prinzipiell ist die Berechnung dessen nicht herausfordernd, jedoch ist die Anzahl der Diskretisierungspunkte im System ausschlaggebend für den Rechenaufwand. Außerdem setzt sich die Systemmatrix $\mathcal{H}$ aus mehreren Untermatrizen zusammen, was einen exponentiell ansteigenden Speicherbedarf mit sich führt.

Aus Gründen der Nachvollziehbarkeit wird die örtliche Abtastung in Matrixform anhand eines 1D-Systems demonstriert. Angenommen, es soll die Wellenausbreitung der x-gerichteten, y-polarisierten TEM-Mode mit $\vec{\Psi} = [E_y, H_z]^T$ berechnet werden, dann ergibt sich die Systemmatrix

$$\mathcal{H} = \begin{bmatrix} 0 & -\frac{1}{\epsilon} D_x^H \\ \frac{1}{\mu} D_x^E & 0 \end{bmatrix}. \tag{3.15}$$

Äquivalent zu $\vec{\Psi}$ aus der Gleichung (3.12) wird $\mathcal{H}$ in der Gleichung (3.15) mit den korrespondierenden Elementen des Materialmodells und der Randbedingungen aufgefüllt bzw. erweitert.

Mittels der Untermatrizen D_x^E und D_x^H wird die diskrete Rotation der E_y- und H_z-Komponente in x-Richtung durchgeführt. Diese wiederum sind wie folgt definiert:

$$D_x^E = \frac{1}{\Delta x}\begin{bmatrix} -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & -1 & 1 \\ 0 & 0 & 0 & 0 & -1 \end{bmatrix}, \tag{3.16}$$

$$D_x^H = -(D_x^E)^T = \frac{1}{\Delta x}\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & \ddots & \ddots & 0 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}. \tag{3.17}$$

Somit sind D_x^E aus der Gleichung (3.16) und D_x^H aus der Gleichung (3.17) stets quadratische Matrizen der Ordnung $N_x \times N_x$, wobei $N_x \in \mathbb{N}$ die Anzahl der Diskretisierungspunkte in x-Richtung widerspiegelt. Des Weiteren sind in $\mathcal{H}^{2N_x \times 2N_x}$ aus der Gleichung (3.15) insgesamt $\frac{2N_x - 1}{{N_x}^2}$ Nicht-Null-Elemente vorhanden, die sich ausschließlich auf der Haupt- und den direkten Nebendiagonalen der Untermatrizen D_x^E und D_x^H befinden. Beispielsweise kann bei einem 1D-System mit $N_x = 1000$ Diskretisierungspunkten nur durch das Ausblenden der Nulleinträge eine relative Kompressionsrate von 99,8001 % erzielt werden. In absoluten Zahlen bedeutet dies, dass anstatt 8 MBit bereits 15,992 kBit zum Abspeichern der Systemmatrix $\mathcal{H}$ ausreichen, wenn der Datentyp *double* mit 8 Byte pro Element verwendet wird. Mit jeder Erhöhung der Dimension des Systems wird dieser Effekt zusätzlich verstärkt.

Eine weitere Möglichkeit zur Optimierung des Speicherbedarfs ist durch das Verhältnis zwischen den Untermatrizen D_x^E und D_x^H gegeben. Anstelle von zwei Untermatrizen reicht es aus, nur eine zu speichern, da die andere durch Transponieren und Vorzeichenwechsel erzeugt werden kann. Diese Erkenntnis kann auf jede Kopplung zwischen der $\vec{E}$- und $\vec{H}$-Feldkomponente unabhängig der Dimension der Struktur übertragen werden, wodurch beim 2D-Fall drei Untermatrizen sowie beim 3D-Fall sechs Untermatrizen nicht initialisiert werden müssen.

3.3 Randbedingungen

Bei der Approximation des zeitabhängigen Propagators in realitätsnahen Konfigurationen ist die Implementierung geeigneter Randbedingungen unerlässlich, um das Rechengebiet einzuschränken und eine numerische Lösung prinzipiell zu ermöglichen [96]. Je nach Geometrie können die Randbedingungen die Komplexität der Berechnung signifikant beeinträchtigen. Aus diesem Grund werden im Folgenden die grundlegenden Ansätze zur Umsetzung der verschiedenen Randbedingungen vorgestellt.

3.3.1 Reflektierende Randbedingung

Mit Hilfe von reflektierenden Randbedingungen können ideal leitende Spiegel bzw. Oberflächen für jegliche Anwendung simuliert werden, sei es für den Resonator eines Lasers [97] oder zur gezielten Führung von Wellen in Leitern [98]. Obwohl die nachfolgenden Randbedingungen in der konventionellen FDTD-Methode bereits implizit berücksichtigt werden, sei darauf hingewiesen, dass orthogonale Strukturen für eine fehlerfreie Berechnung vorausgesetzt werden [9].

Perfect Electric Conductor

Wird zur Modellierung eines ideal leitenden Materials ein Perfect Electric Conductor (PEC) als Randbedingung verwendet, so verschwindet das tangentiale $\vec{E}$-Feld an der Oberfläche und es folgt

$$\vec{E}_{tan} = 0. \tag{3.18}$$

Dies lässt sich äquivalent in Abhängigkeit vom Oberflächennormalenvektor $\vec{n}$ ausdrücken als

$$\vec{n} \times \vec{E} = \vec{0}. \tag{3.19}$$

Im Gegensatz dazu kann sich das magnetische Feld nur tangential entlang der Oberfläche eines PECs ausbreiten, jedoch kann es die Oberfläche nicht senkrecht durchdringen:

$$\vec{n} \cdot \vec{H} = \vec{0}. \tag{3.20}$$

Dennoch führt die tangentiale magnetische Feldkomponente an der Oberfläche eines PECs zu einem induzierten Oberflächenstrom $\vec{K}$, welcher durch die folgende Randbedingung beschrieben wird:

$$\vec{n} \times \vec{H} = \vec{K}. \tag{3.21}$$

Jener Strom resultiert aufgrund der Wechselwirkung des äußeren Magnetfeldes mit dem ideal leitenden Material. Dadurch wird sichergestellt, dass die Bedingung aus der Gleichung (3.18) stets erfüllt wird. Ohne diesen kompensierenden Strom könnte sich ein tangentiales elektrisches Feld an der Oberfläche ausbilden, was dem idealisierten Verhalten eines PECs widerspricht. In der Abbildung 3.4 ist ein PEC mit den relevanten Feldern schematisch dargestellt.

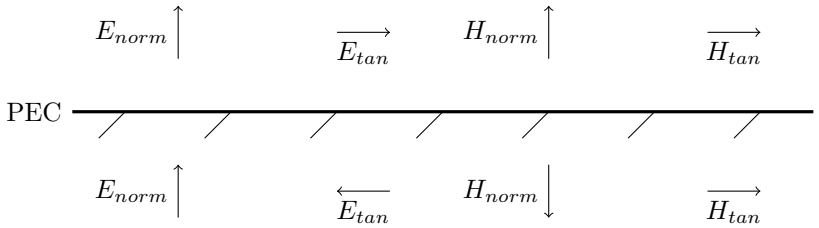


Abbildung 3.4: Illustration eines PECs.

Perfect Magnetic Conductor

Der Perfect Magnetic Conductor (PMC) bildet das Gegenstück zum PMC. Hierbei ist besonders erwähnenswert, dass dieser ein rein theoretisches Konstrukt darstellt und so nicht in der Natur vorkommt. Deshalb muss ein PMC synthetisch durch Metamaterialien oder künstliche magnetische Oberflächen hergestellt werden. Tatsächlich konnten in der jüngeren Vergangenheit bereits Oberflächen mit den Eigenschaften eines PMCs für bestimmte Frequenzbereiche synthetisch erzeugt werden. Der PEC ist speziell für die Antennentechnik von großem Interesse, da er durch seine ideal leitenden Eigenschaften die Miniaturisierung von Antennenbauteilen ermöglicht [99]. Im Kontrast zum PEC verschwinden beim PMC sowohl die Normalkomponente des $\vec{E}$-Feldes

$$\vec{n} \cdot \vec{E} = \vec{0} \tag{3.22}$$

als auch die Tangentialkomponente des $\vec{H}$-Feldes

$$\vec{n} \times \vec{H} = \vec{0}. \tag{3.23}$$

Dadurch befindet sich das elektrische Feld vollständig in der Tangentialebene, während das magnetische Feld keine tangentialen Anteile besitzt. Die Abbildung 3.5 veranschaulicht die Orientierung der Feldkomponenten bei einem PMC.

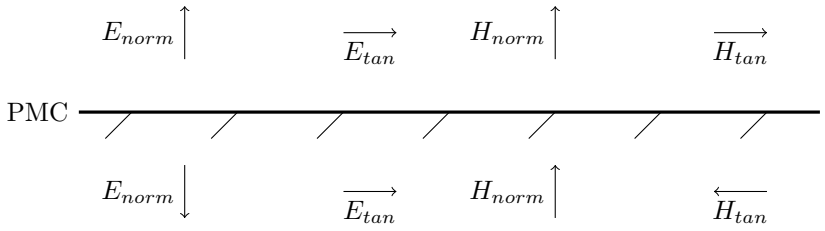


Abbildung 3.5: Illustration eines PMCs.

3.3.2 Dämpfende Randbedingungen

Zur Vermeidung von Reflexionen an den Rändern eines Rechengebiets wäre es grundsätzlich möglich, das Rechengebiet künstlich zu vergrößern, sodass die Welle den Rand nicht erreicht. Aus Gründen der Speicher- und Zeiteffizienz ist diese Methode jedoch in der Praxis nicht sinnvoll. Besonders bei der Simulation von Freiraumausbreitungen [100] und Streuprozessen [101] müssen unerwünschte Reflexionen numerisch unterdrückt werden, um das physikalisch korrekte Verhalten der elektromagnetischen Felder nicht zu verfälschen. Aus diesem Grund werden die dämpfenden Randbedingungen eingeführt, welche das Pendant zu den reflektierenden Randbedingungen darstellen. Trifft hier eine Welle auf den Rand des Rechengebiets, wird ihre Leistung sukzessive reduziert. Dies kann bis zur vollständigen Absorption modelliert werden.

Absorbing Boundary Conditions

Die ABC gehören zu den bekanntesten Vertretern der dämpfenden Randbedingungen, wobei insbesondere die Untersuchungen in [102] und [103] in der einschlägigen Literatur als fundamental gelten. Die Herausforderung bei der Anwendung von ABC besteht darin, dass der zentrale Differenzenquotient zur Approximation der Maxwell-Gleichungen nach Yee auch an den äußersten Rändern des Rechengebiets benötigt wird. Da das Rechengebiet jedoch per Definition endlich ist, stehen außerhalb seiner Grenzen keine Feldinformationen zur Verfügung, sodass der zentrale Differenzenquotient an diesen Stellen nicht anwendbar ist. Zwar könnte an den Rändern stattdessen auf Rückwärtsdifferenzen ausgewichen werden, dies würde aber zu einer geringeren numerischen Genauigkeit führen [9].

Angesichts dieser Tatsachen war die Intention in [102] eine Wellengleichung zu formulieren, die die Wellenausbreitung ausschließlich in eine Richtung erlaubt und numerisch austretende Wellen effektiv absorbiert. Abgesehen von der fehlenden Information außerhalb des Rechengebiets stellte auch die inhärente Nichtlokalität herkömmlicher ABC eine zusätzliche Herausforderung dar. Präziser erforderte die Implementierung der ABC nicht nur den vorherigen Zeitschritt und die benachbarten Gitterpunkte am Rand, sondern sämtliche berechnete Zeitschritte sowie die Feldwerte aller Diskretisierungspunkte. Das Ergebnis der Untersuchung war eine neu analytisch

hergeleitete, praxistaugliche Randbedingung mit einem lokalen Fehler zweiter Ordnung in Raum und Zeit. Darüber hinaus wurde eine numerisch stabile Approximation sichergestellt. Auf Grundlage von [102] wurde in [103] erstmals die Vereinbarkeit der ABC mit der FDTD-Methode erforscht. Es zeigte sich, dass das Resultat für damalige Ansprüche zufriedenstellend war. Folglich wurden jene ABC einschließlich ihrer Erweiterungen zu einer der weit verbreitetsten dämpfenden Randbedingungen im Kontext von FDTD-Simulationen [104].

Im Hinblick auf die Gegenwart ermöglichen die ABC die Modellierung der Wellenausbreitung mit geringer Reflexion, sowohl in dispersiven oder verlustbehafteten Materialien als auch für 2D- und 3D-Strukturen im freien Raum [9]. Allerdings kann die Implementierung der ABC in 3D-Systemen aufgrund der hohen Speicheranforderungen besonders herausfordernd sein [105]. Zudem bleiben die numerische Dispersion und die Genauigkeit bei der Modellierung der Strukturen weiterhin einschränkende Faktoren in FDTD-Simulationen [9]. Außerdem bieten die ABC aus [103] keine ausreichende Absorption bei Inhomogenitäten [80], evaneszenten Wellen [106] und für Einfallswinkel größer als 30° relativ zum Rand [107].

Perfectly Matched Layer

Um den genannten Nachteilen der ABC entgegenzusteuern, entwickelte Berenger in seiner wegweisenden Untersuchung [108] die PML. Zur Modellierung seiner Randbedingung wird ein absorbierendes Medium verwendet, bei dem idealerweise die Dicke des Randes nur aus einer geringen Anzahl an Zellen besteht. Die PML bietet wesentlich verbesserte numerische Eigenschaften verglichen mit den ABC, da sie die Absorption aller charakteristischen Wellen ermöglicht, unabhängig von Faktoren wie Frequenz, Einfallswinkel oder Polarisation. Des Weiteren ist die Terminierung heterogener, anisotroper, dispersiver und nichtlinearer Medien möglich [9]. Konzeptionell sieht die PML eine Erhöhung der Freiheitsgrade vor, indem jede Feldkomponente in jeweils zwei richtungsabhängige Unterkomponenten aufgeteilt wird. Für die Implementierung wird lediglich die Anpassung der FDTD-Methode an den gewünschten Rändern benötigt [72].

Obwohl der beschriebene *Split-Field*-Ansatz der PML viele Vorteile bietet, besteht der Hauptnachteil in einem gegenüber der ABC deutlich erhöhten Speicherbedarf, trotz der geringen Zellanzahl der PML. In diesem Zusammenhang sind besonders die Erweiterungen durch die *stretched-coordinates* [109, 110] sowie die uniaxiale PML (UPML) nach [111] hervorzuheben, die auf unterschiedliche Weise eine Reduktion des Speicherbedarfs bewirken. Die *stretched-coordinates* stellen eine mathematische Manipulation der PML dar. Sie ermöglichen es, die PML auf beliebige orthogonale oder krummlinige Koordinatensysteme zu übertragen. Darauf aufbauend wurde mit der UPML ein modifizierter Ansatz der *stretched-coordinates* entwickelt, der aufgrund seiner physikalischen Plausibilität zu den beliebtesten PML-Methoden zählt [72]. Auch wenn die UPML ein fiktives Konstrukt ist, kommt [112] dennoch zum Ergebnis, dass sich eine Absorption mit den Eigenschaften der UPML theoretisch durch physikalisch modellierbare Materialien realisieren lässt. Die erstmalige Integration der UPML in die FDTD-Methode erfolgte in [113].

Trotz der Fortschritte im Bereich der PML war es mit herkömmlichen Ansätzen nur unzureichend möglich, evaneszente Wellen zu absorbieren. Dies lag daran, dass deren Evaneszenz senkrecht zur Grenzfläche zwischen dem Vakuum und der PML verläuft, während sich die Ausbreitung parallel zur PML erstreckt [114]. Zudem sind evaneszente Wellen stark frequenzabhängig, weshalb in [115] die Complex-Frequency-Shifted (CFS)-PML hergeleitet wurde. Die Idee der CFS-PML

besteht darin, im Frequenzbereich eine komplexe Streckung der Koordinaten innerhalb der PML vorzunehmen. Nachfolgend wird zur Veranschaulichung ausschließlich die Dämpfung der CFS-PML in x-Richtung betrachtet, die äquivalent auf die y- und z-Richtung übertragbar ist. Dazu wird die komplexe Streckungsfunktion

$$s_x = \kappa_x + \frac{\sigma_x}{\alpha_x + j\omega\epsilon_0} \tag{3.24}$$

mit dem Absorptionsfaktor $\kappa_x \geq 1$, dem Abklingfaktor $\sigma_x \geq 0$ und dem Frequenzverschiebungsfaktor $\alpha_x \geq 0$ definiert. Mit Hilfe von s_x aus der Gleichung (3.24) ergibt sich für die komplexe Streckung der örtlichen Ableitung der Maxwell-Gleichung innerhalb der PML im Frequenzbereich die folgende Darstellung:

$$\frac{\partial}{\partial x'} = \frac{1}{s_x}\frac{\partial}{\partial x}. \tag{3.25}$$

Damit die Reflexionen infolge der endlichen örtlichen Abtastung an der Oberfläche der PML minimiert werden, ist die Implementierung der in der Gleichung (3.24) eingeführten PML-Parameter κ_x, σ_x und α_x als konstante Skalare nicht empfehlenswert. Stattdessen wird in [108] ein monoton zu- bzw. abnehmender Verlauf dieser Parameter über die gesamte PML hinweg vorgeschlagen, der die grundlegenden Absorptionseigenschaften der PML nicht beeinträchtigt. Im Kontext der PML hat sich ein geometrischer Verlauf für σ_x etabliert [116]:

$$\sigma_x(x) = \left(\frac{x}{d_x}\right)^m \sigma_{x,max}. \tag{3.26}$$

In der Gleichung (3.26) kennzeichnet d_x die Dicke der PML. Als Startwert wird $\sigma_x(0) = 0$ innerhalb der PML vorgegeben. Anschließend wird σ_x sukzessive bis zur äußeren Grenze des PECs auf $\sigma_x(d_x) = \sigma_{x,max}$ erhöht. Die Steuerung des Abklingfaktors σ_x erfolgt über den Parameter m, wobei sich für FDTD-Simulationen Werte im Bereich von $m \in [3, 4]$ als nahezu ideal erwiesen haben [113]. Sofern der Reflexionsfaktor

$$R(\theta) = \exp\left(-2\cos(\theta)\sqrt{\frac{\mu}{\epsilon}}\int_0^{d_x}\sigma_x(x)\mathrm{d}x\right) \tag{3.27}$$

sowie die PML-Parameter m und d_x bekannt sind, kann bei einem geometrischen Verlauf für eine angestrebte Reflexion $R(0)$ der maximale Abklingfaktor bestimmt werden [9]:

$$\sigma_{x,max} = -(m+1)\frac{\ln(R(0))}{2\sqrt{\mu}d_x}\sqrt{\epsilon}. \tag{3.28}$$

Bei der UPML hat sich ein leicht modifizierter geometrischer Verlauf für den Absorptionsfaktor bewährt [9]:

$$\kappa_x(x) = 1 + (\kappa_{x,max} - 1)\left(\frac{x}{d_x}\right)^m \tag{3.29}$$

Im Vergleich zur Gleichung (3.26) wird in der Gleichung (3.29) der Anfangswert mit $\kappa_x(0) = 1$ vorgegeben. Für die maximale Absorption am Rand des PECs gilt $\kappa_x(d_x) = \kappa_{x,max}$. Zwar bewirkt die CFS-PML bei niedrigen Frequenzen eine signifikante Reduktion des Reflexionsfehlers, allerdings gilt dies nicht uneingeschränkt für das Innere der PML. Aus diesem Grund wurde in der Gleichung (3.24) der Frequenzverschiebungsfaktor α_x eingeführt. Wie bereits bei σ_x in der Gleichung (3.26) und κ_x in der Gleichung (3.29) wird auch hier ein geometrischer Verlauf

gewählt. Diesmal hat die Funktion ihr Maximum am Anfang der PML und fällt innerhalb der PML kontinuierlich bis auf Null ab [9]:

$$\alpha_x = \alpha_{x,max}\left(\frac{d_x - x}{d_x}\right)^{m_a} \tag{3.30}$$

Die Gleichung (3.30) kann über die Skalierungsordnung m_a angepasst werden, wobei in [117] ein Wert von $m_a \approx 1$ empfohlen wird.

Kapitel 4

Faberpolynome

Nachdem die Diskretisierung des Ortes nach Yee in Kapitel 3 ausführlich behandelt wurde, soll nun die zeitliche Komponente des Propagators in den Vordergrund rücken. Insbesondere gilt die Aufmerksam den Faberpolynomen, welche vom Mathematiker Georg Faber [14] entwickelt worden sind und hinsichtlich der Lösung der Maxwell-Gleichungen im Zeitbereich vielversprechende Vorteile gegenüber konventionellen Verfahren mit sich bringen. Dennoch ist die Anzahl der Untersuchungen zu den Faberpolynomen in diesem Kontext derzeit überschaubar. Nach der anfänglichen Behandlung der theoretischen Grundlagen der Faberpolynome wird die Skalierung des Eigenwertspektrums im Detail untersucht. Da die Verteilung der Eigenwerte als unbekannt gilt, werden Methoden zur Abschätzung des Eigenwertspektrums vorgestellt. Darauf basierend wird die Implementierung der Faber-Methode diskutiert und potenzielle Optimierungsmöglichkeiten zur Erhöhung der Effizienz präsentiert. Darüber hinaus wird ein Überblick über den aktuellen Stand der Forschung zur Anwendung der Faberpolynome auf die Maxwell-Gleichungen gegeben.

4.1 Einführung in die Faberpolynome

Um die Maxwell-Gleichungen im Zeitbereich numerisch anzunähern, wird zu Beginn die Ortsdiskretisierung nach Yee angewandt. Das Hauptziel ist die Implementierung einer Approximation, die eine deutlich größere Zeitschrittweite als jene infolge der restriktiven CFL-Bedingung aus der Gleichung (3.11) erlaubt, zugleich Stabilität gewährleistet, gute Konvergenzeigenschaften aufweist und eine hohe Genauigkeit bietet. In Übereinstimmung mit [15] wird entsprechend den formulierten Anforderungen ein Matrixexponential-Ansatz verfolgt. Mit der Gleichung (3.13) kann die Mastergleichung für ein isotropes, verlustfreies Medium aufgestellt werden [9]:

$$\frac{\partial \vec{\Psi}(t)}{\partial t} = f(\mathcal{H}) \cdot \vec{\Psi}(t) \rightarrow \vec{\Psi}(t + \Delta t) = \exp(\Delta t \mathcal{H}) \cdot \vec{\Psi}(t). \tag{4.1}$$

Damit die Feldverteilung, ausgehend vom Zeitpunkt t, eine Zeitschrittweite Δt später betrachtet werden kann, muss das Matrixexponential $\exp(\Delta t \mathcal{H})$ in der Gleichung (4.1) ausgewertet werden. Auf den ersten Blick scheint eine direkte Berechnung von $\exp(\Delta t \mathcal{H})$ sinnvoll, sobald Δt festgelegt wurde, weil alle relevanten Parameter bekannt sind und diese nur in die Exponentialfunktion eingesetzt werden müssen. Allerdings muss die Ordnung der Systemmatrix $\mathcal{H}$ berücksichtigt werden, die eine Vorberechnung signifikant erschwert.

In Spezialfällen kann dennoch die Berechnung von $\exp(\Delta t \mathcal{H})$ vereinfacht werden. Ein wesentliches Merkmal ist, dass $\mathcal{H}$ idealerweise bereits als Diagonalmatrix vorliegt, sodass Matrixinversionen oder Eigenwertprobleme aufgrund der hohen Matrixordnung vermieden werden können [118].

Sofern $\mathcal{H}$ die Struktur einer Blockmatrix, einer zyklischen Matrix oder einer Toeplitz-Matrix aufweist, können die Verfahren aus [119] angewandt werden. Eine weitere Möglichkeit zur Approximation von $\exp(\Delta t\mathcal{H})$ bieten die Tschebyscheff-Polynome, die durch hervorragende Genauigkeit und Konvergenzeigenschaften gekennzeichnet sind [120]. Außerdem erlauben sie eine Approximation mit einer wesentlich größeren Zeitschrittweiten als Δt_{CFL}, allerdings ist ihre Anwendung auf Systeme mit schiefsymmetrischen Matrizen mit rein imaginären Eigenwerten beschränkt. Das Inkludieren von Materialmodellen oder einer PML ist nicht realisierbar.

Im Kontext der Approximation von $\exp(\Delta t\mathcal{H})$ ist ebenso das Krylov-Unterraum-Verfahren hervorzuheben [121], welches den Arnoldi- [122] und den Lanczos-Algorithmus [123] zur Erzeugung der Krylov-Unterräume verwendet. Der Arnoldi-Algorithmus gestaltet sich besonders effizient, wenn $\mathcal{H}$ eine hermetische Struktur besitzt und vorzugsweise dünnbesetzt ist. Dann kann eine Rekursionsbeziehung mit einer festen Länge von drei Ausdrücken pro Iteration ausgenutzt sowie von einem konstanten Speicherbedarf ausgegangen werden. Bei einer allgemeinen Matrix ist dies mit dem Arnoldi-Algorithmus nicht zu erreichen. Eine mögliche Lösung bietet der Lanczos-Algorithmus [121]. Als nachteilig stellt sich für das Krylov-Unterraum-Verfahren die Approximation mit Zeitschrittweiten $\Delta t \gg \Delta t_{CFL}$ heraus. In diesem Fall expandiert der zu generierende Unterraum und folglich auch der Speicher [124]. Für weitere Methoden zur Approximation von $\exp(\Delta t\mathcal{H})$, einschließlich der Taylorreihe [125] und der Padé-Approximation [126], wird auf [13] verwiesen.

Im Rahmen dieser Arbeit bildet die Approximation des zeitabhängigen Propagators $\exp(\Delta t\mathcal{H})$ mittels der Faberpolynome das Kernstück. Abgesehen von der Möglichkeit Δt_{CFL} deutlich überschreiten zu können, ermöglichen sie eine bessere Effizienz bezüglich des Speicherbedarfs als die Krylov-Unterraum-Methode [127]. Zusätzlich ist die Approximation mit den Faberpolynomen von keiner speziellen Struktur der Systemmatrix abhängig. Zu diesem Zweck wird mit Hilfe der Faberreihe die Gleichung (4.1) in die folgende Form überführt [128]:

$$\vec{\Psi}(t+\Delta t) = \exp(\Delta t\mathcal{H}) \cdot \vec{\Psi}(t) = \left(\sum_{m=0}^{\infty} c_m(\Delta t) F_m(\mathcal{H}) \right) \cdot \vec{\Psi}(t). \tag{4.2}$$

In der Gleichung (4.2) werden die Faberpolynome F_m sowie die korrespondierenden Entwicklungskoeffizienten c_m der Ordnung m eingeführt. Dabei entfällt die Zeitkomponente auf c_m, während die Ortskomponente in F_m enthalten ist. Des Weiteren kann die Faberreihe als Verallgemeinerung der Taylorreihe angesehen werden [129]. Anstatt um einen einzelnen Punkt zu entwickeln, erfolgt die Entwicklung über das Komplement des Kontinuums K, welches in diesem Fall ein einfach zusammenhängendes Gebiet G in der komplexen $\mathfrak{z}$-Ebene repräsentiert. Weiter wird angenommen, dass eine Jordankurve Γ [130] das Gebiet G mit $0 \notin G$ umhüllt [36, 131]. Beliebte Formen hierfür sind Kreise, Ellipsen oder Polygone, wie im weiteren Verlauf noch erläutert wird. Bezogen auf die Faber-Methode bedeutet dies, dass die Approximation im Eigenwertraum von $\mathcal{H}$ stattfindet, wobei jene Eigenwerte sich in der $\mathfrak{z}$-Ebene befinden [128]. Unter dieser Prämisse muss nach dem Riemannschen Abbildungssatz [132] eine konforme Abbildung mit $\phi(\mathfrak{z}) = \mathfrak{w}$ existieren, bei der sich das Komplement von G in der $\mathfrak{z}$-Ebene biholomorph auf das Komplement des Einheitskreises in der $\mathfrak{w}$-Ebene abbilden lässt. Dabei sollen $\phi(\infty) = \infty$, $|\phi(0)| > 1$ und für die Ableitung dessen $\phi'(\infty) > 0$ gelten, sodass $\phi(\mathfrak{z})$ mit einer Laurentreihe [133] um ∞ entwickelt werden kann [134]:

$$\phi(\mathfrak{z}) = \mathfrak{a}\mathfrak{z} + \mathfrak{a}_0 + \frac{\mathfrak{a}_1}{\mathfrak{z}} + \frac{\mathfrak{a}_2}{\mathfrak{z}^2} + \cdots = \mathfrak{a}\mathfrak{z} + \sum_{m=0}^{\infty} \frac{\mathfrak{a}_m}{\mathfrak{z}^m}. \tag{4.3}$$

In der Gleichung (4.3) gibt $\mathfrak{a}_m$ den Koeffizienten der Laurentreihe der Ordnung m an. Nach [135] können die Faberpolynome aufgestellt werden, indem nur der Nebenteil der Laurentreihe für $\phi(\mathfrak{z})^m$ ab $m \geq 1$ mit der Bedingung $F_0(\mathfrak{z}) := 1$ ausgewertet wird:

$$\phi(\mathfrak{z})^m := F_m(\mathfrak{z}) + \sum_{i=1}^{\infty} \frac{\mathfrak{a}_{i,m}}{\mathfrak{z}^i}. \tag{4.4}$$

Um schließlich das Komplement des Einheitskreises in der $\mathfrak{w}$-Ebene auf das Komplement des Gebietes G in der $\mathfrak{z}$-Ebene abbilden zu können, wird die inverse konforme Abbildung $\psi(\mathfrak{w})$ wieder als eine Laurentreihe ausgedrückt:

$$\psi(\mathfrak{w}) = \phi(\mathfrak{z})^{-1} = \gamma\mathfrak{w} + \gamma_0 + \frac{\gamma_1}{\mathfrak{w}} + \frac{\gamma_2}{\mathfrak{w}^2} + \cdots = \gamma\mathfrak{w} + \sum_{m=0}^{\infty} \frac{\gamma_m}{\mathfrak{w}^m}. \tag{4.5}$$

Mittels der γ-Koeffizienten aus der Gleichung (4.5) lassen sich die Faberpolynome formulieren [134]:

$$F_{m+1}(\mathfrak{z}) = \begin{cases} 1 & , m = -1 \\ (\mathfrak{z} - \gamma_0)\varrho^{-1} & , m = 0 \\ \left(\mathfrak{z}F_m(\mathfrak{z}) - \sum_{i=0}^{m} \gamma_i F_{m-i}(\mathfrak{z}) - m\gamma_m\right)\varrho^{-1} & , m \geq 1. \end{cases} \tag{4.6}$$

Damit die konforme Abbildung $\phi(\mathfrak{z})$ mit der inversen konformen Abbildung $\psi(\mathfrak{w})$ in Beziehung gesetzt werden kann, wird die logarithmische Kapazität $\varrho = \gamma := 1/\mathfrak{a} > 0$ definiert. Die logarithmische Kapazität ist insbesondere für die Stabilitätsanalyse der Faberreihe von großer Bedeutung. Dafür werden zunächst die Entwicklungskoeffizienten c_m bestimmt [136]:

$$c_m(\Delta t) = \frac{1}{2\pi j} \int\limits_{|\mathfrak{w}|=\varrho} \frac{f(\psi(\mathfrak{w}))}{\mathfrak{w}^{m+1}} d\mathfrak{w} = \frac{1}{2\pi j} \int\limits_{|\mathfrak{w}|=\varrho} \frac{\exp(\Delta t \psi(\mathfrak{w}))}{\mathfrak{w}^{m+1}} d\mathfrak{w}. \tag{4.7}$$

Gemäß der Gleichung (4.1) wird in der Gleichung (4.7) der Matrixexponential-Ansatz ausgenutzt. Zur Berechnung des komplexen Kurvenintegrals in der Gleichung (4.7) eignet sich die Cauchy-Integralformel [137]. Hierbei symbolisiert ϱ den Radius der Jordankurve Γ respektive des zuvor genannten Kreises in der komplexen $\mathfrak{w}$-Ebene. Wird die logarithmische Kapazität auf $\varrho = 1$ gesetzt, erfolgt die Integration über den Einheitskreis. Nach [128] wird dieser Sonderfall als skaliertes Gebiet bezeichnet. Demnach lassen sich die Faberpolynome F_m auf die gesamte Systemmatrix $\mathcal{H}$ anwenden, sodass in Anlehnung an die Gleichung (4.6) die folgende Rekursionsgleichung resultiert:

$$F_{m+1}(\mathcal{H}) = \begin{cases} \mathbb{1} & , m = -1 \\ \mathcal{H} - \gamma_0 \mathbb{1} & , m = 0 \\ \mathcal{H}F_m(\mathcal{H}) - \sum_{i=0}^{m} \gamma_i F_{m-i}(\mathcal{H}) - m\gamma_m \mathbb{1} & , m \geq 1. \end{cases} \tag{4.8}$$

Aufgrund der Matrixnotation $\mathfrak{z} \to \mathcal{H}$ in der Gleichung (4.8) wird die Einheitsmatrix $\mathbb{1}$ verwendet. Solange das Eigenwertspektrum von $\mathcal{H}$ innerhalb des skalierten Gebietes liegt, gilt die rekursive Beziehung der Faberpolynome in der Gleichung (4.8) als stabil [128].

4.2 Skalierung des Eigenwertspektrums

Ausgehend vom Abschnitt 4.1 soll nun die Skalierung des Eigenwertspektrums $\sigma_s(\mathcal{H})$ der Systemmatrix $\mathcal{H}$ ausführlich behandelt werden. Dementsprechend wird fortan für die Entwicklung der Faberpolynome aus Stabilitätsgründen von einer logarithmischen Kapazität $\varrho = 1$ ausgegangen. Des Weiteren soll die tatsächliche Verteilung der Eigenwerte vorerst als unbekannt angesehen werden, dennoch können allgemeine Zusammenhänge zur Einordnung ausgenutzt werden. Eine bekannte Tatsache ist, dass in einem System ohne Dämpfung und Verstärkung die Systemmatrix $\mathcal{H}$ reell sowie schiefsymmetrisch ist. Als Konsequenz sind die Eigenwerte rein imaginär. Zusätzlich treten die Eigenwerte als komplex konjugierte Paare auf, wodurch sie symmetrisch zur reellen Achse sind [118]. Ist das System durch das Einbinden von Dämpfung verlustbehaftet, treten Eigenwerte in der linken Halbebene auf. Je größer die Dämpfung ist, desto weiter ragen die Eigenwerte in die linke Halbebene hinein. Unter diesen Umständen gilt das System als stabil. Äquivalent dazu bewirkt eine Verstärkung im System, dass Eigenwerte in der rechten Halbebene auftreten und das System instabil wird [138]. Sofern nicht anders angegeben, ist die Verteilung der Eigenwerte auch bei diesen Phänomenen symmetrisch zur reellen Achse.

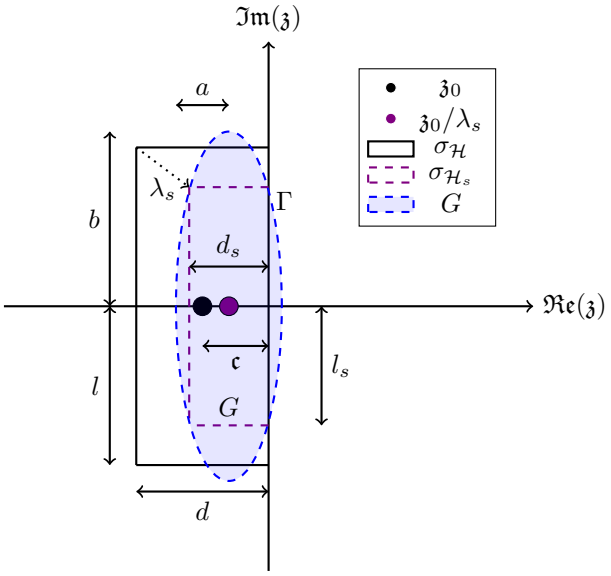


Abbildung 4.1: Das Rechteck mit durchgezogenen Linien um $\mathfrak{z}_0$ veranschaulicht die abgeschätzten Grenzen des ursprünglichen Eigenwertspektrums $\sigma_s(\mathcal{H})$. Nach der Skalierung mit λ_s ergibt sich ein neues skaliertes Eigenwertspektrum $\sigma_s(\mathcal{H}_s)$, welches vom gestrichelten Rechteck mit dem Mittelpunkt bei $\mathfrak{z}_0/\lambda_s$ umschlossen wird. Weiterhin wird das Gebiet G durch die Fläche in der gestrichelten Ellipse dargestellt und von der Jordankurve Γ eingegrenzt. Ferner berührt Γ das skalierte Rechteck an den Ecken mit der Intention das skalierte Rechteck mit dem kleinsten Flächeninhalt zu umschließen.

Auf Grundlage dieser Informationen wird die Skalierung sowohl mathematisch als auch grafisch in der Abbildung 4.1 veranschaulicht. Diesbezüglich wird in [20] vorgeschlagen das Eigenwertspektrum $\sigma_s(\mathcal{H})$ in erster Instanz über ein Rechteck abzuschätzen. Weil der Fokus in dieser Arbeit im späteren Verlauf auf das Einbinden von Dämpfungen, Materialmodellen und Nichtlinearitäten liegt, werden Eigenwerte in der linken Halbebene erwartet. Aufgrund der zuvor

erwähnten Symmetrieeigenschaft genügt es das Rechteck über die halbe Länge l hinsichtlich des Imaginärteils zu charakterisieren. Im Gegensatz dazu kann für die Breite des Rechtecks $d = 2\mathfrak{c}$ bezüglich des Realteils der Umstand ausgenutzt werden, dass für die Eigenwerte $\mathfrak{Re}(z \leq 0)$ gilt. Auf diese Weise kann eine Kante des Rechtecks exakt auf die imaginäre Achse gelegt werden. Der Mittelpunkt des Rechtecks ist in x-Richtung mit x_0 und in y-Richtung mit y_0 definiert sowie mit $\mathfrak{z}_0 = x_0 + jy_0$ zusammengefasst.

Um der Anforderung nach Stabilität gerecht zu werden, muss das Rechteck mit der logarithmischen Kapazität ϱ verknüpft werden, jedoch stößt das bis hierhin vereinfachte Gedankenmodell zum Umschließen der Eigenwerte mittels eines Rechtecks auf seine Grenzen. Dies ist der Tatsache geschuldet, dass der Einheitskreis in der $\mathfrak{w}$-Ebene durchaus über eine inverse konforme Abbildung $\psi(\mathfrak{w})$ in das besagte Rechteck in der $\mathfrak{z}$-Ebene abgebildet werden kann, dafür aber erst $\psi(\mathfrak{w})$ für das zugrunde liegenden System ermittelt werden muss. Im Falle eines unbekannten $\psi(\mathfrak{w})$ wäre ein Rückgriff auf die Schwarz-Christoffel-Transformation [139] erforderlich, welche jedoch erst in Kapitel 5 im Detail behandelt wird.

Demzufolge ist es unter diesen Gesichtspunkten wünschenswert eine bereits bekannte inverse Abbildung für die Jordankurve Γ ausnutzen zu können. In [20] fällt die Wahl auf eine Ellipse. Die Gründe hierfür sind einerseits, dass eine wohlbekannte inverse Abbildung $\psi(\mathfrak{w})$ existiert und andererseits die Anzahl der γ-Koeffizienten endlich respektive gering ist. Für die Ellipse heißt dies, dass die Laurentreihe aus der Gleichung (4.5) bei $m = 1$ terminiert:

$$\psi(\mathfrak{w}) = \mathfrak{w} + \gamma_0 + \frac{\gamma_1}{\mathfrak{w}}. \tag{4.9}$$

Ferner ist die Ellipse über ihre kleine Halbachse

$$a = \varrho + \frac{\gamma_1}{\varrho} \xrightarrow{\varrho=1} a = 1 + \gamma_1 \tag{4.10}$$

und ihre große Halbachse

$$b = \varrho - \frac{\gamma_1}{\varrho} \xrightarrow{\varrho=1} b = 1 - \gamma_1 \tag{4.11}$$

festgelegt, die im Verhältnis

$$\frac{(x - x_0)^2}{a^2} + \frac{(y - y_0)^2}{b^2} = 1 \tag{4.12}$$

zueinander stehen. Um nun die Ellipse mit dem Rechteck zusammenzuführen, muss zuvor die Skalierung des Eigenwertspektrums $\sigma_s(\mathcal{H}) \rightarrow \sigma_s(\mathcal{H}_s)$ vollzogen werden. Für diesen Zweck wird der Skalierungsfaktor λ_s eingeführt. Für das skalierte Rechteck, welches nun $\sigma_s(\mathcal{H}_s)$ umschließt, ergeben sich die angepassten Längenangaben $d \rightarrow d_s$, $l \rightarrow l_s$ und $\mathfrak{c} \rightarrow \mathfrak{c}_s$. Die Halbachsen mit den skalieren Seiten korrelieren über die Gleichung

$$\frac{b^2}{a^2} = \frac{b^2 - {l_s}^2}{{\mathfrak{c}_s}^2}, \tag{4.13}$$

wenn die Ellipse die Eckpunkte des skalierten Rechtecks berührt. Zudem sind beide Gebiete um $\frac{\mathfrak{z}_0}{\lambda_s} = \frac{x_0 + y_0}{\lambda_s}$ zentriert. Weitere Auswirkungen der Skalierung über λ_s zeigen sich in der Systemmatrix $\mathcal{H}$ sowie in der Zeitschrittweite Δt. Die skalierte Systemmatrix ergibt sich aus

$$\mathcal{H}_s = \mathcal{H}/\lambda_s, \tag{4.14}$$

während sich die skalierte Zeitschrittweite

$$\Delta t_s = \Delta t \lambda_s. \tag{4.15}$$

antiproportional dazu verändert. Wie aus der Umstellung von der Gleichung (4.15) hervorgeht, führt ein kleines λ_s zu einer entsprechend großen Zeitschrittweite Δt. In Anbetracht dessen wird die Minimierung von λ_s angestrebt, indem die Ellipse mit dem kleinsten Flächeninhalt entwickelt wird, welche gerade noch das skalierte Rechteck umschließt. Für $\varrho = 1$ kann das minimale λ_s berechnet werden, wenn sich

$$a = 2 - b \tag{4.16}$$

über das Einsetzen der Gleichung (4.11) in die Gleichung (4.10) ergibt sowie b auf

$$b = \sqrt{{l_s}^2 + ({\mathfrak{c}_s {l_s}^2)^{2/3}}} \tag{4.17}$$

festgelegt wird. Der resultierende Skalierungsfaktor λ_s lässt sich nach [20] über

$$\lambda_s = \frac{(l^{2/3} + \mathfrak{c}^{2/3})^{3/2}}{2}. \tag{4.18}$$

bestimmen.

Es ist trivial, dass die skalierten Parameter $\mathcal{H}_s$ und Δt_s als Argument an die Approximation von der Gleichung (4.2) übergeben werden:

$$\vec{\Psi}(t + \Delta t) = \exp(\Delta t_s \mathcal{H}_s) \cdot \vec{\Psi}(t) = \left(\sum_{m=0}^{\infty} c_m(\Delta t_s) F_m(\mathcal{H}_s) \right) \cdot \vec{\Psi}(t). \tag{4.19}$$

Im Normalfall müsste zunächst das komplexe Kurvenintegral in der Gleichung (4.7) zur Bestimmung der Entwicklungskoeffizienten c_m ausgewertet werden, was insbesondere bei Polygonen herausfordernd ist. Stattdessen lässt sich folgender analytischer Ausdruck für die entsprechende Ordnung m ausnutzen, welcher die Berechnung deutlich vereinfacht [18]:

$$c_m(\Delta t_s) = \exp(\Delta t_s \gamma_0) \left(\frac{1}{j\sqrt{\gamma_1}} \right)^m J_m(2j \Delta t_s \sqrt{\gamma_1}). \tag{4.20}$$

In der Gleichung (4.20) beschreibt J_m die Besselfunktion der Ordnung m. Die γ-Koeffizienten können der inversen Abbildungsfunktion $\psi(\mathfrak{w})$ von der Gleichung (4.5) entnommen werden und haben gemäß [128] direkten Einfluss auf die Rekursionsbeziehung der Faberpolynome F_m:

$$F_{m+1}(\mathcal{H}) = \begin{cases} \mathbb{1} & , m = -1 \\ \mathcal{H} - \gamma_0 \mathbb{1} & , m = 0 \\ (\mathcal{H} - \gamma_0) F_1(\mathcal{H}) - 2\gamma_1 \mathbb{1} & , m = 1 \\ (\mathcal{H} - \gamma_0) F_m(\mathcal{H}) - \gamma_1 F_{m-1}(\mathcal{H}) & , m \geq 2. \end{cases} \tag{4.21}$$

Wie in der Gleichung (4.21) zu sehen, ist ein Rekursionsanfang bei $m = 1$ im Vergleich zu jener Rekursionsbeziehung aus der Gleichung (4.8) dazu gekommen. Der weitaus wichtigere Teil ist im Rekursionsschritt bei $m \geq 2$ zu erkennen. Anstelle einer ständigen Aufsummierung der Faberpolynome in jedem Rekursionsschritt im Intervall $i \in [0, m]$ werden nur die Faberpolynome der Ordnung m und $m - 1$ benötigt.

4.3 Eingrenzung des Eigenwertspektrums

Wie aus dem Abschnitt 4.2 hervorgeht, ist eine zu große Abschätzung im Sinne des Skalierungsfaktors λ_s von der Gleichung (4.18) kontraproduktiv. In der Regel werden die Gershgorin-Kreise [140] zur schnellen Abschätzung der Eigenwerte verwendet, wobei sie meist nur eine grobe Einschätzung der tatsächlichen Eigenwerte liefern. Aus diesem Grund befasste sich die Untersuchung in [19] mit der möglichst genauen Abschätzung der Seiten d und l des ursprünglichen Rechtecks in Bezug auf den Real- und Imaginärteil. Hierbei wurde der Ansatz gewählt nach ebenen Wellen zu entwickeln und in den k-Raum des Wellenvektors k zu transformieren, während nur die Extremwerte der Materialparameter berücksichtigt werden. In diesem Fall werden die Grenzen des Eigenwertspektrums durch das Intervall des Wellenvektors $k \in [-k_{max}, k_{max}]$ festgelegt, da die Extrema in dem definierten Intervall im Normalfall mit den Eigenwerten übereinstimmen [19]. Der Parameter k_{max} gibt den Maximalbetrag des Wellenvektors k wieder und wird mit der örtlichen Diskretisierungsweite Δ gemäß $|k_{max}| = 2/\Delta$ ermittelt. Unter bestimmten Voraussetzungen kann sogar mit jenem Ansatz eine analytische Lösung für die Grenzen des Eigenwertspektrums ermittelt werden. Dies gilt z.B. für den Fall von verlustlosen Systemen, wo alle Eigenwerte ausschließlich imaginär sind. Diesbezüglich kann die Breite des Rechtecks auf $d = 0$ gesetzt werden. Die hergeleitete analytische Lösung für die Länge l ist für die jeweilige betrachtete Dimension wie folgt gegeben, nachdem $\mathcal{H}$ gemäß [141] zuvor normalisiert wurde [19]:

$$l = \begin{cases} 2c \cdot \sqrt{\dfrac{1}{\Delta x^2}} & \text{, 1D-Fall} \\ 2c \cdot \sqrt{\dfrac{1}{\Delta x^2} + \dfrac{1}{\Delta y^2}} & \text{, 2D-Fall} \\ 2c \cdot \sqrt{\dfrac{1}{\Delta x^2} + \dfrac{1}{\Delta y^2} + \dfrac{1}{\Delta z^2}} & \text{, 3D-Fall.} \end{cases} \tag{4.22}$$

Die Bestimmung einer analytischen Lösung wäre sogar noch für 1D-Systeme mit einer PML praktikabel. Im Gegensatz dazu ist die Bestimmung einer analytischen Lösung für komplexere Systeme mit sehr hohem Rechenaufwand verbunden oder generell nicht mehr möglich. Unter diesen Umständen werden die diskreten Werte von k aus dem Intervall $k \in [-k_{max}, k_{max}]$ zur Annäherung der Eigenwerte verwendet, wobei jede Komponente von k gleichförmig diskretisiert wird [15].

4.4 Praktische Umsetzung der Faber-Methode

Um die Faber-Methode ordnungsgemäß anwenden zu können, müssen entscheidende Aspekte bei der Implementierung berücksichtigt werden. Die in der Gleichung (4.2) vorgestellte Faberreihe ist zunächst als unendliche Reihe formuliert. Es stellt sich die Frage welche Entwicklungsordnung eine ideale Balance zwischen der Rechenzeit und Genauigkeit der Approximation darstellt. Eine zu hohe Ordnung mag zwar die Genauigkeit der Approximation des zeitabhängigen Propagators verbessern, jedoch geht dies mit einer höheren Rechenzeit einher. Umgekehrt würde ein zu frühes Terminieren der Faberreihe die Rechenzeit deutlich verringern, dafür müsste aber eine

höhere Fehlerrate toleriert werden. Zu diesem Zweck wird der Parameter N_{pol} definiert, der eine endliche Terminierung der Faberreihe erlaubt:

$$\vec{\Psi}(t+\Delta t) = \exp(\Delta t\mathcal{H}) \cdot \vec{\Psi}(t) := \left(\sum_{m=0}^{N_{pol}} c_m(\Delta t) F_m(\mathcal{H}) \right) \cdot \vec{\Psi}(t). \tag{4.23}$$

Generell kann N_{pol} in der Gleichung (4.23) beliebig gewählt werden kann. Tatsächlich liefert die Terminierung der Faberreihe ausschließlich über N_{pol} keine Erkenntnis über die Genauigkeit, da N_{pol} lediglich die Entwicklungsordnung der Faberreihe angibt. Dennoch kann der Faber-Ansatz zur Untersuchung der Komplexität über N_{pol} mit der FDTD-Methode in ein Verhältnis gesetzt werden [19]. Dies wird im weiteren Verlauf näher ausgeführt.

Anstelle der Entwicklungsordnung wird die Wertigkeit von c_m als Abbruchkriterium für die Terminierung der Faberreihe herangezogen. In [18] wird vorgeschlagen die Approximation zu terminieren, sobald der Schwellenwert von $c_{th} = 10^{-15} \geq |c_m|$ unterschritten wird. Da mit dem Faber-Ansatz deutlich größere Zeitschrittweiten als Δt_{CFL} gemäß der Gleichung (3.11) realisiert werden können, kommt der Analyse von c_m eine zentrale Bedeutung zu. Zu diesem Zweck wird der Zeitfaktor

$$q = \frac{\Delta t}{\Delta t_{CFL}} \tag{4.24}$$

eingeführt, der im Rahmen der jeweiligen Untersuchung das ganzzahlige Vielfache von Δt_{CFL} beschreibt. Des Weiteren wird angenommen, dass die skalierten Eigenwerte $\sigma_s(\mathcal{H}_s)$ wie im Abschnitt 4.2 von einer Ellipse eingegrenzt werden. Somit kann die Rechenvorschrift für c_m aus der Gleichung (4.20) benutzt werden. Einerseits sollte zur Kenntnis genommen werden, dass aufgrund der Besselfunktion J_m die Wertigkeit von c_m erst mit steigendem m rapide abfällt. Andererseits muss eine Erhöhung von q eine Erhöhung von N_{pol} nach sich ziehen, damit die Approximation über die Faberreihe stabil bleibt. Trotzdem kann unter dem Aspekt der Rechenzeit eine Erhöhung des Schwellenwerts sinnvoll sein.

4.5 Optimierung des Approximationsablaufs

Formal können die Entwicklungskoeffizienten c_m durch die Gleichung (4.7) sowie die Faberpolynome F_m durch die Gleichung (4.8) vor der eigentlichen Berechnung bestimmt und mit der Faberreihe in der Gleichung (4.2) miteinander verrechnet werden. Auf den ersten Blick scheint der Ablauf intuitiv. Auf den zweiten Blick wird allerdings deutlich, dass besonders der Zwischenspeicher zur Berechnung von F_m immens beansprucht wird. Für jeden Rekursionsschritt ab $m \geq 1$ werden insgesamt $m+3$ Matrizen mit der Matrixordnung von $\mathcal{H}$ benötigt. Bei einem 1D-System kann dies noch mit moderatem Aufwand bewältigt werden, spätestens ab einem 2D-System ist die Durchführung mit üblichen Speicherressourcen jedoch nicht mehr umsetzbar. Die Nutzung von dünnbesetzten Matrizen kann diese Hürde auch nicht überwinden.

In der Tat trägt die Verwendung der Ellipse zur Eingrenzung der skalierten Eigenwerte deutlich dazu bei, diesem Umstand entgegenzuwirken. Anstatt $m+3$ Matrizen genügen nun zwei Matrizen im Zwischenspeicher für den Rekursionsschritt ab $m \geq 2$ aus der Gleichung (4.21). Falls $\mathcal{H}$ trotzdem zu speicherintensiv ist, muss im schlimmsten Fall davon ausgegangen werden, dass nur eine Matrix in der Größe von $\mathcal{H}$ temporär vorliegen darf. Dennoch ist die Implementierung

in diesem Zusammenhang problemlos umsetzbar, sofern der Zustandsvektor $\vec{\Psi}$ direkt in die Approximation integriert wird:

$$\vec{\Psi}(t+\Delta t) = \exp(\Delta t_s \mathcal{H}_s) \cdot \vec{\Psi}(t) := \left(\sum_{m=0}^{N_{pol}} c_m(\Delta t_s) F_m(\mathcal{H}_s) \right) \cdot \vec{\Psi}(t) = \sum_{m=0}^{N_{pol}} \left(c_m(\Delta t_s) F_m(\mathcal{H}_s) \vec{\Psi}(t) \right). \tag{4.25}$$

Durch diese geringfügige Anpassung in der Gleichung (4.25) bleibt die Approximation im mathematischen Sinne unbeeinflusst, jedoch lassen sich aus programmiertechnischer Sicht erhebliche Vorteile erzielen. Einerseits liegen statt Matrizen zu jeder Ordnung m ausschließlich Vektoren im Zwischenspeicher vor. Andererseits wird die Rechenzeit signifikant reduziert, indem nach der Initialisierung von $\mathcal{H}$ durch effiziente Programmierung nur noch Vektoradditionen während der Approximation vonnöten sind. Die allgemeine Rekursionsbeziehung von F_m aus der Gleichung (4.8) wird in diesem Sinne modifiziert zu

$$F_{m+1}(\mathcal{H}) = \begin{cases} \vec{\Psi} & , m = -1 \\ (\mathcal{H} - \gamma_0) \cdot \vec{\Psi} & , m = 0 \\ \mathcal{H} F_m(\mathcal{H}) - \sum_{i=0}^{m} \gamma_i F_{m-i}(\mathcal{H}) - m\gamma_m \vec{\Psi} & , m \geq 1. \end{cases} \tag{4.26}$$

Zwar wird dadurch die potenzielle Vorberechnung von F_m zunichtegemacht, weil zuvor $\vec{\Psi}$ initialisiert werden muss. Jedoch ist der Weg über die Vorberechnung aus den genannten Gründen ohnehin nicht ratsam, sodass nur der hier präsentierte Ansatz praktikabel bleibt.

4.6 Stand der Forschung

Um einen Überblick über den aktuellen Stand der Forschung zu den Faberpolynomen im Kontext der Maxwell-Gleichungen zu erhalten, werden die bisherigen Untersuchungen grob zusammengefasst. Dadurch soll sowohl offenbart werden welche Errungenschaften mit den Faberpolynomen bisher erzielt wurden auch als welche potenziellen Untersuchungen noch in Betracht gezogen werden können.

Eigenwertabschätzung und Konvergenzeigenschaft

Im Abschnitt 4.3 wurde bereits gezeigt, dass die Abschätzung der Eigenwerte nach Gershgorin zu grob für eine effiziente Approximation mit der Faber-Methode ist. Die Untersuchung in [19] ist speziell für verlustlose Systeme jeder Dimension von Interesse, da sie die analytische Bestimmung der Ober- und Untergrenzen für die resultierenden imaginären Eigenwerte einer Systemmatrix ermöglicht, welche nach Yee örtlich diskretisiert wurden. Auch können die Eigenwerte für Materialmodelle oder für Systeme mit Dämpfungen numerisch abgeschätzt werden. Darüber hinaus erfolgte eine Effizienzanalyse, bei der die Faber-Methode unter Verwendung von Zeitfaktoren bis $q = 500$ mit der konventionellen FDTD-Methode verglichen wurde. Es stellte sich heraus, dass für eine niedrige Genauigkeit die FDTD-Methode zu bevorzugen ist, bedingt durch die grundlegende Konvergenzeigenschaft des Verfahrens. Umgekehrt übertraf der Faber-Ansatz die FDTD-Methode um mehrere Größenordnungen, sobald eine hohe numerische Genauigkeit gefordert war. Der Einfluss von q machte sich insbesondere bei der Konvergenz

bemerkbar. Mit zunehmendem q trat die Sättigung bei einer geringen Entwicklungsordnung N_{pol} auf. Auffällig war auch, dass eine Erhöhung von q keinen Einfluss auf die maximal erreichbare Genauigkeit hatte. Sobald die Konvergenz erreicht wurde, blieb die Fehlerrate konstant.

Quellterme, absorbierende Grenzschichten und komplexe Einhüllende

In [17] wurde das Einbinden zeitabhängiger Quellterme, wie etwa einer Punktquelle oder einer ebenen Welle an einer Grenzschicht, in die Faber-Methode behandelt. Außerdem ermöglichte die Verwendung der Auxiliary Differential Equations (ADE) die Integration von Materialmodellen und der PML. Eine wichtige Erkenntnis war, dass obwohl der Faber-Ansatz höhere Zeitschrittweiten als Δt_{CFL} ermöglicht, die Entwicklung von Quelltermen bei höheren Δt zu höheren Entwicklungsfehlern führt. Ähnlich verhielt es sich auch bei der Frequenz, bei der eine Erhöhung zu einem höheren Fehler führte. Positiv anzumerken ist, dass der Speicher nur minimal mehr belastet wurde. Dies liegt daran, dass bereits eine geringe Anzahl an Diskretisierungspunkten ausreicht, um praxisnahe Quellterme zu modellieren.

Die Analyse in [32] zielte darauf ab ein Complex Envelope (CE)-Verfahren mit der Faber-Methode zu verknüpfen, damit es im Falle von bandbegrenzten Feldern trotzdem möglich ist die CFL-Bedingung zu umgehen. Mit der CE-Faber-Methode konnte dies verwirklicht werden. Selbst als das Nyquist-Abtasttheorem [142] durch Unterabtastung und einer sehr hohen Zeitschrittweite verletzt wurde, konnte mit der komplexen Einhüllenden das ursprüngliche Signal um die Trägerfrequenz herum rekonstruiert werden. Dadurch konnte die Feldverteilung nur durch das Abtasten der komplexen Einhüllenden bestimmt werden.

Mit der Forschung in [33] wurde eine Verbindung zwischen [17] und [32] hergestellt. Die Idee war die Komplexität für die Einbindung von Quelltermen für praktische Anwendungen weiter zu reduzieren. Es wurden u.a. die Integration von Stromverteilungen sowie die Anregungen von Eigenmoden an der Grenzschicht von Wellenleitern behandelt. Ein Ansatz sah die Aufspaltung der Quellterme in einen beitragsfreien und einen quellenbehafteten Anteil vor. Der Teil mit Quellterm erforderte den Einsatz komplexer Algebra für die Auswertung und entsprechend viel Rechenaufwand, weshalb hierfür die CE-Faber-Methode ausgenutzt wurde. Dies konnte bis auf reine Vektoradditionen reduziert werden. Es wurde gezeigt, dass mit der CE-Faber-Methode eine höhere Genauigkeit gegenüber der konventionellen Faber-Methode erzielt werden kann. Zudem eignet sie sich besser zur Bestimmung der Wellenform bei einem höheren Zeitfaktor q.

Time-Domain Beam Propagation

In [34] lag der Fokus auf der Implementierung einer rein vektoriellen, expliziten Time-Domain Beam Propagation Method (TDBPM), die auf den Faberpolynomen basiert. Auf Grundlage der Wideband (WB)-TDBM wurde die Narrowband (NB)-TDBPM abgeleitet. Ähnlich wie bei der CE-Methode sollte eine sehr große Zeitschrittweite bei hoher Genauigkeit realisiert werden. Neu war die Betrachtung der Skalierung der Systemmatrix in Abhängigkeit der inhärenten Trägerfrequenz, womit die Approximation des zeitabhängigen Propagators weiter optimiert werden konnte. Es wurde festgestellt, dass eine höhere Trägerfrequenz eine stärkere Skalierung verursacht, sodass eine geringere Entwicklungsordnung ausreicht. Ferner wurde die Effizienz der NB-TDBPM untersucht. Dabei wurde als Schwachstelle die örtliche Auflösung der untersuchten Welle ausgemacht, die einen enormen Einfluss auf die Komplexität hat.

Mit der Folgeuntersuchung in [35] sollte dieser Umstand mittels einer neuartigen WB-TDBPM behoben werden. Bezüglich der Eigenwertskalierung traten dieselben Auffälligkeiten wie bei

der WB-TDBPM auf. Durch den Einsatz einer Padé-Approximation konnte die Steigung der Komplexität bei der WB-TDBPM im Vergleich zur NB-TDBPM wesentlich reduziert werden. Des Weiteren ließ sich der relative Fehler um mehrere Größenordnungen verringern.

Nichtlineare Integratoren

Abschließend wurde in [16] angestrebt, multiphysikalische Phänomene durch die Integration nichtlinearer Medien in die Faber-Methode approximieren zu können. Eine Hauptaussage war, dass die meisten nichtlinearen Methoden auf der Linearisierung einer Mastergleichung beruhen. Im ersten Schritt stand die Validierung numerischer Methoden im Kontext der Elektrodynamik im Vordergrund, gefolgt von der Implementierung einer Optimierung. Analysiert wurden die Exponential-Euler-, Lawson-Type- und Rosenbrock-Type-Integratoren. Im Rahmen der unterschiedlichen numerischen Evaluierungen konnten speziell die Rosenbrock Integratoren bezüglich der Fehlerrate überzeugen, besonders die EXPRB32-Methode [143]. Des Weiteren war eine höhere Fehlerrate bemerkt worden, je mehr Nichtlinearität in dem System vorhanden war. Diesbezüglich wurde noch der Ansatz einer adaptiven Schrittweite erfolgreich eingepflegt, der nach dem Erreichen des spezifischen Sättigungpunkts eine konstante Genauigkeit ermöglichte.

4.7 Zusammenfassung und Zwischenfazit

Die Faber-Methode wurde über die konforme Abbildung zwischen der $\mathfrak{w}$- und $\mathfrak{z}$-Ebene hergeleitet. Die namensgebenden Faberpolynome F_m und ihre zugehörigen Entwicklungskoeffizienten c_m werden in der Faberreihe miteinander verrechnet, sodass das Matrixexponential $\exp(\Delta t\mathcal{H})$ approximiert werden kann. Zur Gewährleistung der Stabilität ist eine logarithmische Kapazität $\varrho \leq 1$ zu wählen. Als Prämisse für das Prozedere müssen die Eigenwerte $\sigma_s(\mathcal{H})$ erst abgeschätzt und dann zu $\sigma_s(\mathcal{H}_s)$ skaliert werden, da keine exakte Kenntnis über ihre Verteilung vorhanden ist. Hierbei diente in den Grundlagen eine Ellipse als Jordankurve Γ zum Eingrenzen von $\sigma_s(\mathcal{H}_s)$. Weiterhin wurde erläutert wie die Ellipse zu bestimmen ist und welchen Einfluss sie auf die Evaluation hat. Ausgehend davon wurde eine Möglichkeit präsentiert die Grenzen von $\sigma_s(\mathcal{H})$ bestenfalls analytisch und im Normalfall numerisch zu ermitteln. Darüber hinaus ist das unabdingbare Einbinden des Zustandsvektors $\vec{\Psi}$ in die Approximation erörtert worden. Zum Schluss wurde Bezug auf die bisherigen Untersuchungen der Faberpolynome im Kontext der Approximation der Maxwell-Gleichungen im Zeitbereich genommen.

Nach wie vor bietet die Faber-Methode erhebliches Optimierungspotenzial. Im Bereich der Eigenwertskalierung wurde darauf hingewiesen, dass eine engere Umschließung des Eigenwertspektrums zu einer höheren Zeitschrittweite führt. Weil ausschließlich eine Ellipse verwendet wurde, ist die Untersuchung anderer Formen beachtenswert. Abgesehen von den bekannten Abbildungsfunktionen für einen Kreis und ein abgerundetes Quadrat stößt die Klasse der Polygone auf großes Interesse, welche mit der Schwarz-Christoffel-Transformation abgebildet werden können. Ein bisher völlig unbeachteter Aspekt ist die Aufteilung des Rechengebiets in mehrere kleinere Untergebiete, wodurch die Approximation des zeitabhängigen Operators auf mehrere lokale Operatoren verteilt werden kann. Weil nur die wenigsten Systeme von einer homogenen örtlichen Diskretisierung ausgehen, erhält dieser Ansatz besondere Aufmerksamkeit. Abgesehen von den genannten Vorschlägen erfolgte noch keine Untersuchung, welche den Einfluss der genutzten Soft- und Hardware prüfte. Bei der Software wäre ein Vergleich zwischen einer höheren und einer niederen Programmiersprache sinnvoll. Im Rahmen der Hardware könnte aufgrund der

expliziten Natur der Faber-Methode die Parallelisierung des Approximationsablaufs betrachtet werden. Ein weiterer Ansatz zur Verringerung der Komplexität der Approximation besteht darin, die Systemmatrix $\mathcal{H}$ mit überschaubarem Informationsverlust durch eine Projektion in einen geeigneten Unterraum zu komprimieren.

Kapitel 5

Konforme Abbildungen

In diesem Kapitel wird der potenzielle Mehrwert der konformen Abbildung mittels der Schwarz-Christoffel (SC)-Transformation zur Effizienzsteigerung der Faber-Methode untersucht. Das Ziel ist es, die Laufzeit zu verringern, ohne dabei die Genauigkeit negativ zu beeinträchtigen. Dabei wird speziell das Konvergenzverhalten der Faberreihe infolge der SC-Transformation analysiert. Zuvor wird die Theorie der SC-Transformation erläutert und auf die damit verbundenen Herausforderungen eingegangen.

5.1 Einführung in die SC-Transformation

Die SC-Transformation wurde von den Mathematikern Christoffel und Schwarz unabhängig voneinander innerhalb eines Zeitraums von zwei Jahren veröffentlicht, zunächst in [144] und später in [145]. Sie repräsentiert eine spezielle Art der konformen Abbildungen in der komplexen Analysis. Ein zentrales Merkmal der SC-Transformation ist die Winkeltreue der Kontur nach der Abbildung. Im Gegenzug geht die Information über die Distanzen der Ursprungsabbildung verloren. Ausgehend vom Riemannschen Abbildungssatz [132] ist die Existenz einer konformen Abbildung auf ein Polygon zwar garantiert, allerdings ist diese nicht in expliziter Form bekannt. Dahingegen liefert die SC-Transformation eine Integralformel, welche die Innenwinkel direkt berücksichtigt und damit besonders für die konforme Abbildung auf Polygone geeignet ist [146]. Typische Anwendungsfälle der SC-Transformation in der Elektrotechnik umfassen u.a. Mikrowellenleiter [147], integrierte Schaltkreise [148] sowie allgemeine elektromagnetische Applikationen [149].

Im Gegensatz zu Kapitel 4 wird der Einheitskreis in der $\mathfrak{w}$-Ebene nun nicht mehr auf eine Ellipse, sondern im Einklang mit [139] auf ein Polygon in der $\mathfrak{z}$-Ebene abgebildet. Wie bereits erwähnt, müssen gemäß der Definition der Faberpolynome die Komplemente der betrachteten Konturen durch eine konforme Abbildung aufeinander abgebildet werden. Die Abbildungsfunktion für diesen Prozess ist wie folgt definiert [150]:

$$\psi(\mathfrak{w}) = A + C \int_1^{\mathfrak{z}} \mathfrak{w}^{-2} \prod_{i=1}^{n} (\mathfrak{w} - \eta_i)^{\alpha_i - 1} d\mathfrak{w}. \tag{5.1}$$

Die betrachteten einfachen Polygone Γ_p werden über ihre n Eckpunkte $\omega \in [\omega_1, \omega_i, \ldots, \omega_n]$, Innenwinkel $\alpha \in [\alpha_1\pi, \alpha_i\pi, \ldots, \alpha_n\pi]$ und Außenwinkel $\beta \in [\beta_1\pi, \beta_i\pi, \ldots, \beta_n\pi]$ mit der Beziehung $\alpha + \beta = 2\pi$ charakterisiert. Der Parameter $\eta \in [\eta_1, \eta_i, \ldots, \eta_n]$ gibt den Urbildpunkt auf dem Einheitskreis an, der schließlich auf einen Eckpunkt ω von Γ_p abgebildet wird. Aufgrund der mathematischen Konvention erfolgt die Aufzählung aller relevanten Größen gegen den Uhrzeigersinn. In der Gleichung (5.1) ist die komplexe Konstante A für die Translation verantwortlich,

während die komplexe Konstante C die Skalierung und Drehung ermöglicht [151]. Zwar wird mit der Gleichung (5.1) eine kontinuierliche Abbildung durchgeführt, jedoch gilt für die Ableitung von $\psi(\mathfrak{w})$ genau an den Urbildpunkten $\psi'(\mathfrak{w}) = 0$. Infolgedessen wird die Abbildung angepasst [150]:

$$\psi(\mathfrak{w}) = A + C \int_1^{\mathfrak{z}} \prod_{i=1}^{n} \left(1 - \frac{\eta_i}{\mathfrak{w}}\right)^{\alpha_i - 1} d\mathfrak{w}. \tag{5.2}$$

Die Bestimmung der komplexen Konstanten in der Gleichung (5.2) geht mit der Lösung eines nichtlinearen Systems einher. In [152] wird deshalb der Ansatz gewählt für A einen beliebigen Punkt in Γ_p zu nehmen und C über den folgenden Zusammenhang zu ermitteln:

$$C = \frac{\omega_n - A}{\int_1^{\mathfrak{z}} \prod_{i=1}^{n} \left(1 - \frac{\eta_i}{\mathfrak{w}}\right)^{\alpha_i - 1} d\mathfrak{w}}. \tag{5.3}$$

Für die korrekte Abbildung muss die Summe der Drehwinkel der konkaven und konvexen Polygonseiten 2π ergeben [139]. Unter Berücksichtigung der gegebenen Außenwinkel resultiert

$$\sum_{i=1}^{n} (\beta_i - 1) = 2 \tag{5.4}$$

bzw. für Polygone mit n Ecken

$$\sum_{i=1}^{n} (\beta_i - 1)\eta_i = 0. \tag{5.5}$$

Die Abbildung 5.1 zeigt schematisch die Anwendung der SC-Transformation. Je nach Anwendungsfall kann eine analytische oder numerische Auswertung der Abbildungsfunktion erfolgen. Während im analytischen Fall der Einheitskreis exemplarisch auf eine Ellipse abgebildet wird, erfolgt im numerischen Beispiel die Abbildung auf ein einfach zusammenhängendes Gebiet in Form eines konkaven Dekagons. Dabei lassen sich die Eckpunkte der Kontur in der $\mathfrak{z}$-Ebene sowohl abgerundet als auch zuspitzen [139]. Für eine vertiefte Auseinandersetzung mit dem Thema der konformen Abbildungen wird auf [139, 146, 150, 153, 154] verwiesen.

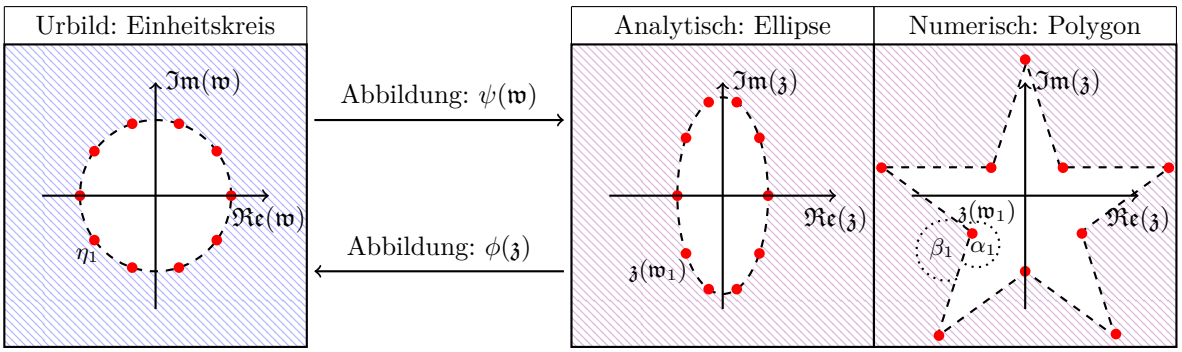


Abbildung 5.1: Auf der linken Seite dient der Einheitskreis in der $\mathfrak{w}$-Ebene als Urbildmenge. Mit der Abbildungsfunktion $\psi(\mathfrak{w})$ wird in die $\mathfrak{z}$-Ebene abgebildet. Bei einfachen Konturen wie einer Ellipse ist $\psi(\mathfrak{w})$ bekannt und analytisch bestimmbar. Für komplexere Konturen wie Polygone ohne Überschneidungen muss die SC-Transformation numerisch ausgewertet werden. Das Abbilden in die $\mathfrak{w}$-Ebene geschieht über die inverse Abbildungsfunktion $\phi(\mathfrak{z})$. Zu beachten ist, dass jeweils die Komplemente der Gebiete aufeinander abgebildet werden.

5.1.1 Parameterproblem

Die Anwendung der SC-Transformation zur Abbildung des Einheitskreises auf ein Polygon gemäß der Gleichung (5.2) erscheint zunächst unkompliziert. Dennoch gilt es mit Bedacht die korrekten Urbildpunkte η_i am Rande des Einheitskreises auszusuchen, sodass diese auf die gewünschten Eckpunkte des Polygons Γ_p abgebildet werden. Wird diese Tatsache außer Acht gelassen und es werden stattdessen willkürliche Urbildpunkte gewählt, hat dies eine Verzerrung der Abbildung zur Folge. In diesem Kontext bedeutet Verzerrung, dass die ursprünglichen Winkelverhältnisse beibehalten, allerdings die Seitenlängen des Polygons Γ_p unvorhersehbar skaliert werden [155]. Somit muss zuvor das sogenannte SC-Parameterproblem gelöst werden, um die gewünschte Abbildung zu erhalten. Aufgrund der nichtlinearen Abhängigkeit zwischen η und den Seiten von Γ_p existiert derzeit nur für die wenigsten praktischen Anwendungen eine analytische Lösung des Parameterproblems. Aufgrund der situativ hohen Komplexität der SC-Transformation ist die numerische Berechnung mit einem PC in den meisten Fällen unvermeidlich [139]. In diesem Zusammenhang sind besonders zwei Toolboxen hervorzuheben. Zum einen gibt es SCPACK [156], welches eine Ansammlung mehrerer FORTRAN-Programme beinhaltet. Zum anderen steht mit der in MATLAB entwickelten SC-Toolbox [157] eine *open-source Software* zur Verfügung, die in *MathWorks* bereitgestellt und in unregelmäßigen Abständen aktualisiert wird.

Eine häufig verwendete Technik zur Lösung des Parameterproblems beruht auf den Verhältnissen zwischen den Seitenlängen von Γ_p [152]. Dabei wird die Tatsache ausgenutzt, dass für Polygone mit $n \leq 3$ Eckpunkten die SC-Transformation bereits explizit ist. In diesen Fällen ist es nicht notwendig das Parameterproblem zu lösen, da die SC-Transformation über drei Freiheitsgrade verfügt, weshalb drei beliebige Urbildpunkte festgelegt werden können. Dagegen gilt es ab $n > 3$ das rechenintensive System mit $n - 3$ Bedingungen für jeden Eckpunkt ω von Γ_p zu lösen [158]:

$$\frac{\left| \int_{\eta_m}^{\eta_{m+1}} \prod_{i=1}^{n} (1 - \frac{\eta_i}{\mathfrak{w}})^{\beta-1} d\mathfrak{w} \right|}{\left| \int_{\eta_1}^{\eta_2} \prod_{i=1}^{n} (1 - \frac{\eta_i}{\mathfrak{w}})^{\beta-1} d\mathfrak{w} \right|} = \frac{|\omega_{m+1} - \omega_i|}{|\omega_2 - \omega_1|}, \qquad m \in [2, 3, \ldots, n-2]. \tag{5.6}$$

In einigen Untersuchungen werden sogar die Konstanten A und C als Unbekannte in das Parameterproblem inkludiert, trotzdem können diese bei vorhandener Kenntnis der Urbildpunkte mit geringem Aufwand ermittelt werden [139]. Einen möglichen Ansatz zeigt die Gleichung (5.3).

5.1.2 Crowding-Phänomen

Ein unerwünschter Effekt, welcher bei der SC-Transformation auftreten kann, ist das *Crowding-Phänomen* [159]. Generell kann das *Crowding* als eine Form der schlechten Konditionierung aufgefasst werden, da es für alle numerischen Methoden hinsichtlich der konformen Abbildungen Schwierigkeiten bereitet [139]. Es tritt besonders dann in Erscheinung, wenn die zu abbildende Kontur in der $\mathfrak{z}$-Ebene relativ lange und schmale Bereiche besitzt. In der Abbildung 5.1 würde dies auf die scharfen Spitzen des Polygons zutreffen. Aufgrund dessen würden sich zahlreiche Urbildpunkte in einem kleinen Bereich des Einheitskreises in der $\mathfrak{w}$-Ebene konzentrieren. Je schärfer die Spitze des Polygons ist, desto kompakter rücken die Urbildpunkte zusammen. Hierbei nimmt der Abstand zwischen den Urbildpunkten exponentiell ab. Im schlimmsten Fall sind die Urbildpunkte nicht mehr als einzelne Punkte zu identifizieren, sodass Rundungsfehler die

Konsequenz sind. Die Folge wäre neben der Verzerrung der Abbildung auch das Auftreten numerischer Instabilität. In MATLAB verwendet der Datentyp *double* bei doppelter Genauigkeit gemäß der aktuellen Revision der IEEE-Norm 754 [160] die Zahl $\pm 2{,}22507 \cdot 10^{-308}$ für den betragsmäßig kleinsten darstellbaren Wert [161].

Ein Verfahren zur Minderung von *Crowding* wäre eine andere Urbildmenge als den Einheitskreis zu wählen [139]. Ebenso könnte durch den Einsatz von Algorithmen wie Cross-Ratios of the Delaunay Triangulation (CRDT) [162], welches die Doppelverhältnisse der Urbildpunkte ausnutzt, sowohl *Crowding* als auch das Parameterproblem gleichzeitig abgefangen werden. Dennoch ist zu berücksichtigen, dass die genannten Optimierungsmöglichkeiten eine höhere Komplexität erfordern und nur der Vollständigkeit halber erwähnt werden.

5.2 Numerische Evaluierung

Um einen Erkenntnisgewinn bezüglich der möglichen Effizienzsteigerung durch die Anpassung der konformen Abbildung zu erhalten, wird das Eigenwertspektrum eines stark verlustbehafteten Drude-Modells wie aus dem Abschnitt 2.2.1 als Referenz herangezogen. Als Konturen dienen sowohl bereits bekannte analytische als auch numerisch bestimmbare Jordankurven Γ. Zur ersten Kategorie gehören neben der zuvor thematisierten Ellipse [20] auch der Einheitskreis [163] und ein Quadrat mit abgerundeten Kanten [128]. In der zweiten Kategorie werden jene Polygone erfasst, die eine Auswertung der SC-Transformation aus der Gleichung (5.2) bedürfen. Die Anzahl der Ecken der Polygone wird auf drei bis sieben begrenzt. Somit wird die Abbildungsfunktion $\psi(\mathfrak{w})$ zur Abbildung des Komplements des Einheitskreises in der $\mathfrak{w}$-Ebene auf das Komplement der Kontur in der $\mathfrak{z}$-Ebene im Rahmen dieser Arbeit auf die folgenden Fälle beschränkt:

$$\psi(\mathfrak{w}) = \begin{cases} \mathfrak{w} + \gamma_0 + \dfrac{\gamma_1}{\mathfrak{w}} & \text{, Ellipse} \\ \mathfrak{w} + \gamma_0 & \text{, Einheitskreis} \\ \mathfrak{w} + \gamma_0 - \dfrac{1}{8\mathfrak{w}^3} & \text{, Abgerundetes Quadrat} \\ A + C \displaystyle\int \prod_{i=1}^{n} \left(1 - \frac{\eta_i}{\mathfrak{w}}\right)^{\alpha_i - 1} d\mathfrak{w} & \text{, Polygon.} \end{cases} \tag{5.7}$$

Ferner ergeben sich in Abhängigkeit von der Anzahl der γ-Koeffizienten in $\psi(\mathfrak{w})$ unterschiedliche Rechenvorschriften für die Entwicklungskoeffizienten

$$c_m(\Delta t_s) = \frac{1}{2\pi j} \oint\limits_{|\mathfrak{w}|=\varrho} \frac{f(\psi(\mathfrak{w}))}{\mathfrak{w}^{m+1}} d\mathfrak{w} = \frac{1}{2\pi j} \oint\limits_{|\mathfrak{w}|=\varrho} \frac{\exp(\Delta t_s \psi(\mathfrak{w}))}{\mathfrak{w}^{m+1}} d\mathfrak{w}. \tag{5.8}$$

Weil nur für die verhältnismäßig einfachen Konturen, wie die Ellipse und den Einheitskreis, analytische Ausdrücke für c_m unter Verwendung der Besselfunktion J_m hergeleitet werden können, muss c_m in den anderen Fällen numerisch approximiert werden. Das entsprechende Vorgehen mit Zuhilfenahme der Cauchy-Produktformel [164] wird umfangreich im Abschnitt A.1 hergeleitet. Bei den Faberpolynomen F_m ist die Rekursionsgleichung aus dem Abschnitt 4.1

Tabelle 5.1: Hardwarespezifikationen zur Untersuchung der konformen Abbildungen.

CPU	Taktrate	Cache	Kerne	RAM
Intel i5-8600K	3,6 GHz	L3 mit 9 MB	6	32 GB

allgemeingültig:

$$F_{m+1}(\mathfrak{z}) = \begin{cases} 1 & ,m = -1 \\ \left(\mathfrak{z}F_m(\mathfrak{z}) - \sum_{i=0}^{m} \gamma_i F_{m-i}(\mathfrak{z}) - m\gamma_m\right) \cdot \varrho^{-1} & ,m \geq 0. \end{cases} \tag{5.9}$$

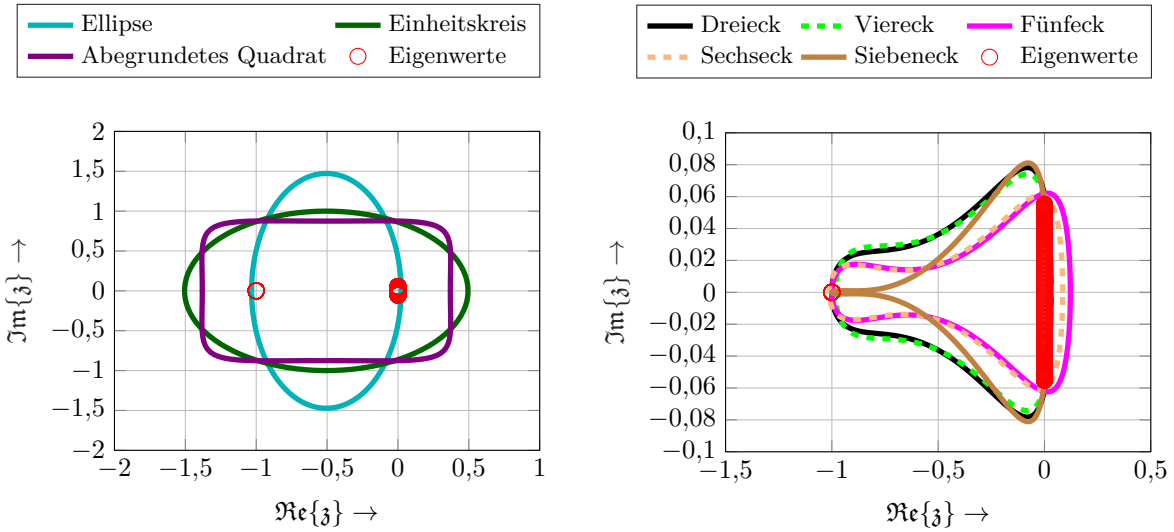
Wie in Kapitel 4 erörtert, muss zur Gewährleistung der Stabilität der Approximation die logarithmische Kapazität nach der Skalierung der Eigenwerte der Systemmatrix $\mathcal{H}$ auf $\varrho \leq 1$ fixiert werden. Die infolge der SC-Transformation generierten Polygone sind so konzipiert, dass sie das skalierte Eigenwertspektrum möglichst eng umschließen. Daher wird ein Wert von $\varrho < 1$ erwartet. Dies ist besonders bei dem Rekursionsschritt ab $m > 0$ in der Gleichung (5.9) zu berücksichtigen. Der angestrebte Rechenzeitgewinn bei der Verwendung der numerischen Konturen soll durch die Reduktion der Entwicklungsordnung N_{pol} im Vergleich zu den analytischen Konturen realisiert werden. Dies ist auf die engere Umschließung des Eigenwertspektrums durch die numerischen Konturen und der schnelleren Konvergenz der Entwicklungskoeffizienten c_m gegen den Schwellenwert c_{th} zurückzuführen. Die verwendeten Hardwarespezifikationen sind in der Tabelle 5.1 zusammengefasst.

5.2.1 Implementierung

In [PS1] wurden erste erfolgreiche Untersuchungen durchgeführt, um die Anwendung der SC-Transformation im Kontext der Approximation des zeitabhängigen Propagators mittels der Faberpolynome zu beweisen. Dabei wurde analog zum Drude-Modell das Debye-Modell aus dem Abschnitt 2.2.1 verwendet. Es zeigte sich, dass der Realteil der Eigenwerte der Systemmatrix um mehrere Größenordnungen kleiner war als der Imaginärteil, was zu einem stark gestauchten Eigenwertspektrum führte. Aus diesem Grund lassen sich die gewonnenen Erkenntnisse aus [PS1] nicht unmittelbar auf andere verlustbehaftete Systeme transferieren. Damit dennoch eine möglichst allgemeingültige und aussagekräftige Analyse für Systeme mit dämpfenden Grenzschichten wie der ABC oder PML aus dem Abschnitt 3.3.2 erzielt werden kann, wurde in [PS2] stattdessen das Drude-Modell eingesetzt. Die Systemmatrix $\mathcal{H}$ und der Zustandsvektor $\vec{\Psi}$ wurden gemäß [19] gewählt und orientieren sich an [124]:

$$\mathcal{H} = \begin{bmatrix} 0 & \frac{1}{\epsilon_0\epsilon_\infty}\vec{\nabla}\times & -\frac{1}{\epsilon_0\epsilon_\infty} \\ -\frac{1}{\mu}\vec{\nabla}\times & 0 & 0 \\ \epsilon_0{\omega_D}^2 & 0 & -\gamma_D \end{bmatrix} \qquad \vec{\Psi}(t) = [\vec{E}, \vec{H}, \vec{J}_D]^T. \tag{5.10}$$

Für die Untersuchung werden die Parameter entsprechend zu [19] mit $\gamma_D = 3{,}23 \cdot 10^{13}\,\mathrm{s}^{-1}$, ${\omega_D}^2 = 1{,}39 \cdot 10^{15}\,\mathrm{s}^{-1}$ und $\epsilon_\infty = 1\,\mathrm{F/m}$ festgelegt. Speziell durch die Festlegung von γ_D lässt sich die Dämpfung des Systems einstellen, wobei der Grenzfall $-\gamma_D \to -\infty$ zu $\mathfrak{Im}(\mathfrak{z}) \to -\infty$ in $\mathcal{H}$



(a) Analytisch bestimmbare Konturen.

(b) Numerisch bestimmbare Konturen.

Abbildung 5.2: Das skalierte Eigenwertspektrum des untersuchten Drude-Modells wird mit analytisch bestimmbaren Konturen in der Abbildung 5.2a eingegrenzt, während es in der Abbildung 5.2b durch numerisch ermittelte Konturen mittels der SC-Transformation beschränkt wird. In beiden Fällen handelt es sich um dasselbe Eigenwertspektrum, allerdings kann dieses mittels der Polygone enger eingegrenzt werden.

in der Gleichung (5.10) führt. Des Weiteren ist die tatsächliche Verteilung der Eigenwerte von $\mathcal{H}$ unbekannt. Bedingt durch die Dämpfung kann die Approximation von der Gleichung (4.21) nicht angewandt werden, da $\mathcal{H}$ nicht mehr schiefsymmetrisch ist und keine rein imaginären Eigenwerte mehr vorliegen. Dennoch gilt aufgrund der erheblichen Größenunterschiede der Einträge in der dünnbesetzten Systemmatrix $\mathcal{H}$ die Beziehung $|\mathfrak{Re}(\mathfrak{z})| \gg |\mathfrak{Im}(\mathfrak{z})|$ gemäß der Gleichung (5.10). Daraus lässt sich unmittelbar schließen, dass $\min\{\mathfrak{Re}(\mathfrak{z})\} = -3{,}23 \cdot 10^{13}$, was eindeutig mit γ_D korreliert. Zusätzlich führt das gezielte Weglassen einer Verstärkung im System zu $\max\{\mathfrak{Re}(\mathfrak{z})\} = 0$.

Zur ungefähren Bestimmung von $|\max\{\mathfrak{Im}(\mathfrak{z})\}|$ bieten sich die Gershgorin-Kreise an, was mit der k-Raum-Methode aus [19] verfeinert werden kann. Da jedoch der Fokus auf der möglichen Reduktion der Rechenzeit liegt und nicht auf einem neuen Verfahren zur präzisen Abschätzung der Eigenwerte, wurde trotz der genannten Approximation auf analytisch bestimmte Eigenwerte zurückgegriffen. Vor diesem Hintergrund haben sich unter der Voraussetzung von $\varrho = 1$ für die skalierte Ellipse $l_s = 0{,}7$ und $d_s = 0{,}3$ sowie eine Verschiebung des Mittelpunkts in die linke Halbebene entlang der x-Achse über $x_0/\lambda_s = -0{,}506$ mit $\lambda_s = 0{,}5539\ \mathrm{s}^{-1}$ als gute Annäherung bewahrheitet. Weiterhin wurde derselbe Mittelpunkt sowohl für den Einheitskreis als auch für das abgerundete Quadrat verwendet. In der Abbildung 5.2a sind die Verteilung der skalierten Eigenwerte in der $\mathfrak{z}$-Ebene und die Eingrenzung dieser über die analytisch bestimmbaren Konturen illustriert.

Im Gegensatz dazu zeigt die Abbildung 5.2b die Umschließung der Eigenwerte durch Polygone mit drei bis sieben Ecken. Zunächst entsprechen die Polygone nicht der erwarteten Form, was auf die Berechnung ihrer inversen Abbildungsfunktion $\psi(\mathfrak{w})$ zurückzuführen ist. Dafür gibt es zwei Gründe. Der erste Grund ist, dass bewusst auf fertige *Toolboxen* wie [157] verzichtet

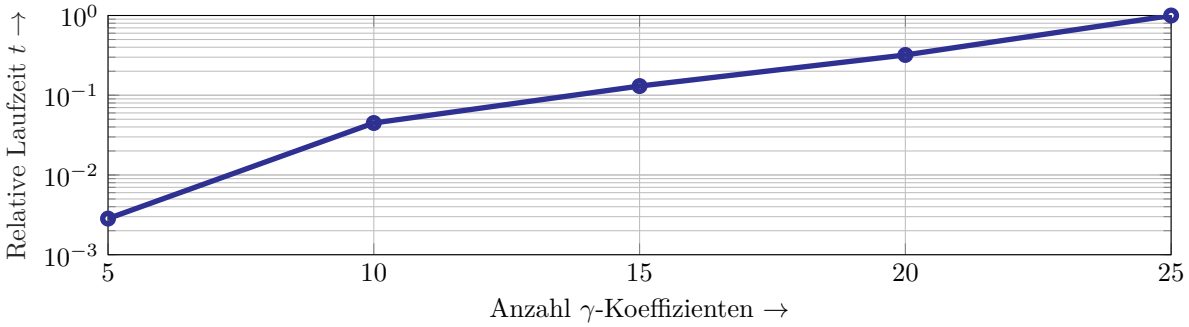


Abbildung 5.3: Einfluss der Anzahl der γ-Koeffizienten auf die Laufzeit beim Dreieck.

wurde, um sich den gegebenen Anforderungen dynamisch anpassen zu können. Der zweite Grund ist, dass ein möglichst objektiver Laufzeitvergleich des zeitabhängigen Propagators unter Verwendung der verschiedenen $\psi(\mathfrak{w})$ ermöglicht werden soll. Je mehr γ-Koeffizienten in $\psi(\mathfrak{w})$ von der Gleichung (5.7) vorhanden sind, desto höher wird der Aufwand für die Auswertung der Faberpolynome F_m in der Gleichung (5.9). Weil die Laurentreihe unter den bekannten $\psi(\mathfrak{w})$ die höchste Ordnung bei dem abgerundeten Quadrat besitzt und bei γ_3 terminiert, wird dies auch für die Polygone gefordert. Die Sinnhaftigkeit dieser Maßnahme zur Verwirklichung eines objektiven Vergleichs wird mit der Abbildung 5.3 anhand des Dreiecks verdeutlicht. Bereits ein Anstieg um fünf γ-Koeffizienten führt zu einer deutlichen Laufzeiterhöhung, bedingt durch die zunehmende Komplexität innerhalb der Summe in der Gleichung (5.9).

Tabelle 5.2: Überblick der Koeffizienten der inversen Abbildung $\psi(\mathfrak{w})$ für die jeweilig Kontur.

	Koeffizient				
Kontur	$\gamma = \varrho$	γ_0	γ_1	γ_2	γ_3
Ellipse	1	-0,506	-0,473	0	0
Einheitskreis	1	-0,506	0	0	0
Abgerundetes Quadrat	1	-0,506	0	0	-0,125
Dreieck	0,294	-0,458	0,237	-0,027	-0,016
Viereck	0,290	-0,454	0,233	-0,029	-0,016
Fünfeck	0,309	-0,417	0,269	-0,023	-0,017
Sechseck	0,302	-0,442	0,263	-0,022	-0,016
Siebeneck	0,284	-0,451	0,239	-0,040	-0,013

Die Berechnung von $\psi(\mathfrak{w})$ für die Abbildung auf die Polygone mit Hilfe der SC-Transformation erfolgt analog zu [PS1]. Da das Integral in der Gleichung (5.2) nur in den seltensten Fällen analytisch gelöst werden kann, wird es durch eine Taylorreihe um $\mathfrak{w} \rightarrow \infty$ approximiert. Dabei resultieren Monome der Ordnung $\mathfrak{w}^{-i}$. Für eine handhabbare Implementierung wird die Substitution von $\mathfrak{w}^{-i}$ durch eine Hilfsvariable ausgenutzt, um die Taylorreihe um $\mathfrak{w} = 0$ entwickeln zu können. Anschließend wird im approximierten Ausdruck die Resubstitution vor der Integration durchgeführt. Die Herausforderung hierbei besteht darin, dass die Integration von $\mathfrak{w}^{-1}$ zu einer logarithmischen Funktion führt, deren Entwicklung um den Punkt $\mathfrak{w} = 0$ divergiert. Die vermeintlich einfachste Lösung ist das Monom $\mathfrak{w}^{-1}$ unter Einhaltung der Bedingungen aus der

Gleichung (5.4) und Gleichung (5.5) zu umgehen. Aufgrund der Tatsache, dass die vorliegenden Eigenwerte in komplex konjugierten Paaren auftreten, können ebenfalls komplex konjugierte Urbildpunkte für die Abbildung ausgesucht werden. Die SC-Transformation wahrt die Winkel der gegenüberliegenden Eckpunkte eines Polygons bezüglich der reellen Achse. Durch diese Tatsache kann die Anzahl der Freiheitsgerade in der Gleichung (5.5) halbiert werden. Wird auf ein Polygon mit einer ungeraden Anzahl an Ecken abgebildet, muss der Urbildpunkt $\eta_{(n+1)/2}$ zuvor definiert werden. Exemplarisch resultiert bei einem Polygon mit $n = 7$ Ecken und $\eta_{(n+1)/2} = \eta_4 = -1$ folgender Ausdruck [PS1]:

$$-\beta_4\eta_4 = 2\beta_1\mathfrak{Re}\{\eta_1\} + 2\beta_2\mathfrak{Re}\{\eta_2\} + 2\beta_3\mathfrak{Re}\{\eta_3\} \tag{5.11}$$

Zur Verdeutlichung sind in der Tabelle 5.2 die γ-Koeffizienten der betrachteten Konturen aufgeführt. Es sei nochmals hervorgehoben, dass die logarithmische Kapazität ϱ dem Koeffizienten vor dem Monom γ der Laurentreihe entspricht. Außerdem sei erwähnt, dass die ideale Abbildungsfunktion zur engsten Eingrenzung der Eigenwerte durch die konvexe Hülle [165] ermittelt werden kann. Hierfür ist jedoch eine vollständige Kenntnis über die Verteilung der betragsmäßig größten Eigenwerte erforderlich.

5.2.2 Ergebnisse

Bevor die Laufzeiten zur Approximation des zeitabhängigen Propagators mit den unterschiedlichen Konturen untersucht werden, wird das Konvergenzverhalten der Entwicklungskoeffizienten c_m studiert. Die Berechnung von c_m kann nach der Gleichung (5.8) unabhängig vom System vorweg mit den γ-Koeffizienten aus der Tabelle 5.2 durchgeführt werden. Die Verläufe für die jeweiligen numerisch und analytisch bestimmten c_m können in der Abbildung 5.4 eingesehen werden. Als Kriterium zum Terminieren der Entwicklung dient die Schwelle beim Absolutbetrag von $c_{th} = 10^{-15}$. Weil ausschließlich die Faber-Methode untersucht wird und die Fehlerrate der Methode primär durch c_m beeinflusst wird, ist ein Vergleich der Genauigkeit nicht vorgesehen. Außerdem wird als Zeitfaktor $q = 1$ angesetzt.

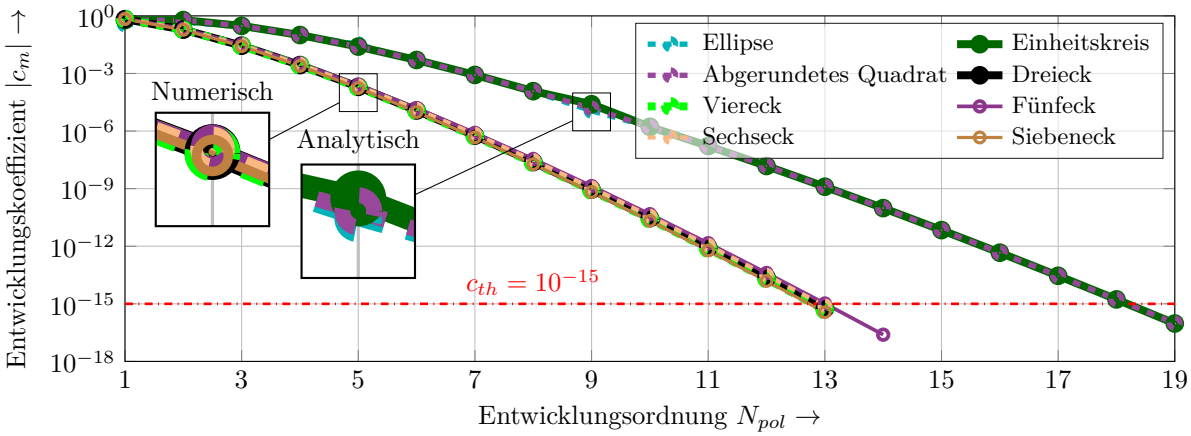


Abbildung 5.4: Die exponentiell abfallenden Verläufe für die Entwicklungskoeffizienten c_m sind für die betrachteten Konturen sowie für eine steigende Abbruchordnung der Faberreihe N_{pol} dargestellt, wobei $c_{th} = 10^{-15}$ gilt.

Es fällt auf, dass die numerisch sowie die analytisch bestimmten Entwicklungskoeffizienten c_m jeweils nahezu identisch verlaufen, abgesehen von minimalen Abweichungen. Es ist nicht auszuschließen, dass jene Abweichungen durch die Maschinengenauigkeit bei der Rundung von Gleitkommazahlen verursacht werden. Die wesentliche Erkenntnis bleibt jedoch die deutlich geringere benötigte Entwicklungsordnung für die Auswertung der Polygone mit der SC-Transformation. Anstelle von $N_{pol} = 19$ reicht bei den Polygonen $N_{pol} = 13$ zur Unterschreitung des Schwellwerts aus. Einzig das Fünfeck erfordert eine Ordnung von $N_{pol} = 14$, was auf den erwähnten Rundungsfehler und auf den größten auftretenden Wert von ϱ aller Polygone zurückgeführt werden kann. Auf Grundlage der vorliegenden Ergebnisse wurde die These bestätigt, dass eine engere Kontur um die Eigenwerte zu einer schnellen Konvergenz führt.

Damit die Auswertung des Propagators im System aus der Gleichung (5.10) rechentechnisch ausgereizt wird, werden $N_x \in [10^6,10^7]$ Diskretisierungspunkte vorgegeben. Ferner ist die relative Rechenzeit in Abhängigkeit von N_x aufgetragen, wobei die maximal auftretende Rechenzeit im gesamten Intervall als Referenz für die Normierung verwendet wird. Die Resultate sind in der Abbildung 5.5 dargestellt. Offensichtlich ist die Evaluation der Faberreihe am schnellsten bei Verwendung des Einheitskreises, wenn auch mit marginalem Unterschied. Es folgen die Laufzeiten der Ellipse und des abgerundeten Quadrats, gefolgt von denen der Polygone. Die Rekursionsbeziehung aus der Gleichung (5.9) liefert Aufschluss über die beobachteten Vorkommnisse.

Wird die Anzahl der Rechenoperationen als Maßstab für die Berechnung eines Zeitschritts herangezogen, fällt auf, dass nicht die Wertigkeit der γ-Koeffizienten, sondern primär ihre Anzahl einen Einfluss in diesem Kontext hat. Sofern die γ-Koeffizienten von der Tabelle 5.2 übernommen werden und die Verrechnung mit $\gamma_i = 0$ keine Gewichtung erhält, ergeben sich unterschiedliche Mengen an Rechenoperationen pro Rekursionsschritt. Bei dem Einheitskreis sind es zwei Multiplikationen und eine Subtraktion. Bei der Ellipse und dem abgerundeten Quadrat sind es für $m = 1$ bzw. $m = 3$ vier Multiplikationen, zwei Additionen und zwei Subtraktionen. Für jeden anderen Rekursionsschritt sind eine Multiplikation und eine Subtraktion weniger notwendig. Im Falle der Polygone steigt die Anzahl an erforderlichen Rechenoperationen zwischen $m = 1$ und $m = 3$ stetig um zwei Multiplikationen und eine Addition. Darüber hinaus ist die Produktbildung mit ϱ nicht zu vernachlässigen, während sie bei den analytischen Abbildungen mit $\varrho = 1$ ignoriert werden kann. Den schlimmsten Fall stellt der Rekursionsschritt bei $m = 4$ dar, wo in Summe 13 Operationen ausgewertet werden müssen. Ab $m = 5$ sind es wiederum eine Multiplikation sowie Subtraktion weniger. Diese Umstände erklären, wieso die Auswertung mit dem Fünfeck die meiste Zeit benötigt, da hier ein höheres N_{pol} vorliegt und mehr Rechenoperationen durchgeführt werden, wie in der Abbildung 5.4 illustriert.

In Anbetracht der Situation stellt sich die Frage in wie vielen Taktzyklen eine Rechenoperation bearbeitet wird und ob dies signifikanten Einfluss auf die Laufzeit hat. Zwar wird in bestimmten Anwendungsbereichen, wie der digitalen Signalverarbeitung, die Leistungsfähigkeit eines Systems anhand von Multiply-Accumulate (MAC)-Rechenoperationen beurteilt [166], jedoch ist im Allgemeinen von einer hardwarespezifischen Bearbeitungszeit auszugehen. Daher lässt sich keine pauschale Aussage treffen. Eine weitere Frage ist, inwieweit die numerischen Abbildungen der Polygone so angepasst werden können, dass sie eine schnellere Laufzeit für die Approximation ermöglichen als mit dem Einheitskreis. Augenscheinlich darf $\psi(\mathfrak{w})$ bei der Abbildung auf ein Polygon nur einen γ-Koeffizienten besitzen, um objektiv mit dem Einheitskreis konkurrieren zu können. Gleichzeitig bedarf die Taylorreihe zur Annäherung des SC-Integrals mindestens eine

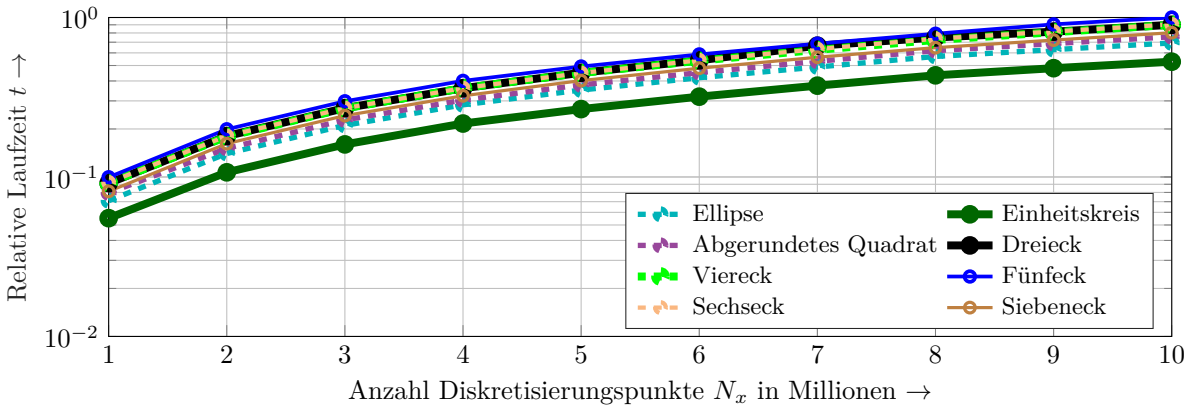


Abbildung 5.5: In Abhängigkeit der Anzahl an γ-Koeffizienten sind geringe Abweichungen der durchschnittlichen Rechenzeit pro Zeitschritt zu vermerken.

Ordnung von $m = 4$, damit die vorliegenden Eigenwerte ordnungsgemäß eingehüllt werden. Da beide Bedingungen nicht gleichzeitig erfüllt werden können, ist dieser Widerspruch unlösbar. Im Hinblick auf die Rechenzeit ist damit der Einheitskreis zu bevorzugen, auch wenn der Rechenzeitgewinn nur minimal ausfällt.

5.3 Zusammenfassung und Zwischenfazit

In diesem Kapitel wurden die Herausforderungen der SC-Transformation adressiert sowie Verfahren zur Bewältigung dieser erörtert. Des Weiteren wurde die SC-Transformation erfolgreich angewandt, um das Eigenwertspektrum der Systemmatrix $\mathcal{H}$ in der $\mathfrak{z}$-Ebene durch numerisch ermittelte Konturen einzugrenzen. Ferner wurden Methoden zur Approximation des Eigenwertspektrums thematisiert, etwa über die Dämpfung, die Gershgorin-Kreise oder die k-Raum-Methode. Für die Faber-Methode war insbesondere interessant, ob die Form der abgebildeten Kontur eine Auswirkung auf die Laufzeit der Approximation des zeitabhängigen Propagators hat. Es wurde bewiesen, dass die Konvergenz der Entwicklungskoeffizienten mit einer reduzierten Entwicklungsordnung erzielt werden kann, je enger die Kontur die Eigenwerte umschließt. Dagegen wurde die Laufzeit weniger von der Form der Kontur beeinflusst, sondern primär durch die Anzahl der γ-Koeffizienten in der jeweiligen Abbildungsfunktion. Aus diesem Grund war die Ausführung bei Verwendung des Einheitskreises am schnellsten, während der Einsatz von Polygonen zu einer leicht erhöhten Laufzeit führte. Die Ursache lag an dem Verhältnis zwischen der Entwicklungsordnung N_{pol} und der Anzahl der Rechenoperation pro Rekursionsschritt zur Evaluation eines Faberpolynoms F_m. Wurde die eine Größe reduziert, nahm die andere zu. Generell bietet die Anwendung der SC-Transformation in verlustfreien wie auch in verlustbehafteten Systemen entgegen der ursprünglichen Annahme keinen zusätzlichen Vorteil.

Kapitel 6

Lokale Operatoren

Um die Komplexität der Faber-Methode zu reduzieren und gleichzeitig eine Verringerung der Laufzeit zu erreichen, wird das neuartige Konzept der lokalen Operatoren erforscht. Dabei werden bekannte Dekompositions- und *Splittingverfahren* sowie Gitterstrukturen behandelt, die auf unterschiedliche Weise zur Reduktion der Komplexität beitragen. Damit der Ansatz der lokalen Operatoren eingeordnet werden kann, wird die theoretisch hergeleitete Komplexität mit praktisch ermittelten Laufzeiten gegenübergestellt. Außerdem wird der Einfluss der Implementierung berücksichtigt und es werden weitere Optimierungsansätze analysiert.

6.1 Dekomposition und Splitting

Die bereits thematisierte Diskretisierung nach Yee aus dem Abschnitt 3.1 erlaubt es die Maxwell-Gleichungen numerisch zu approximieren. Bei der Auswertung des Matrixexponential-Ansatzes zur Beschreibung der elektromagnetischen Wellenausbreitung ist die Ordnung der Systemmatrix $\mathcal{H}$ ein zentraler Faktor für die Effizienz der Approximation. Obwohl $\mathcal{H}$ typischerweise stark dünnbesetzt ist, kann trotzdem nur eine endliche Anzahl an Diskretisierungspunkten zwischengespeichert werden. Im Hinblick auf die lokalen Operatoren werden Zerlegungs- und *Splittingansätze* zur Reduzierung der Komplexität diskutiert. Prinzipiell ist der Ablauf der Ansätze identisch. Zunächst wird ein großes Problem in mehrere kleinere Teilprobleme unterteilt, sodass der Speicheraufwand auf mehrere unabhängige Teilevaluierungen verteilt werden kann. Im Anschluss findet eine Kopplung statt, bei der die Teilergebnisse zu einem Gesamtergebnis vereint werden. Die gesamte Vorgehensweise stellt sich als herausfordernd dar, da die Aufteilung des Gesamtproblems mit einer Fehlerzunahme verbunden sein kann und entsprechend akkurat zusammengesetzt werden muss [80].

6.1.1 Dekomposition des Raumes

Bei der Dekomposition des Raumes wird das Rechengebiet in mehrere kleinere Subgebiete unterteilt. Bereits die Wahl der Unterteilung hat einen erheblichen Einfluss auf die Berechnung. Für den Fall, dass eine schnelle Konvergenz erforderlich ist, sollte die Dekomposition in nicht überlappende Subgebiete vorgenommen werden. Wird hingegen ein parallelisierter Ablauf angestrebt, ist eine Dekomposition mit überlappenden Bereichen vorteilhaft [167]. Die Abbildung 6.1 illustriert die Dekomposition des Raumes ohne und mit Überlappung.

Weiterhin wird zwischen einer iterativen und nicht iterativen Kopplung differenziert, welche unterschiedliche Charakteristiken aufweisen. Eine iterative Kopplung zeichnet sich durch eine

Abbildung 6.1: Illustration der Dekomposition des Raumes.

einfache Implementierung aus, muss aber für eine möglichst genaue Lösung mehrere Iterationszyklen durchlaufen. Als konventionelle Methode sei hier die Schwarz-Waveform-Relaxation (SWR)-Methode [168] genannt, die üblicherweise in iterativen Verfahren verwendet wird. Die SWR-Methode vereint die Schwarz-Iteration [169] zur Dekomposition des Gebiets und die Waveform-Relaxation [170] zur Fehlerkorrektur. Zu den Vertretern der nicht iterativen Kopplungsverfahren zählen u.a. die Mortar-FEM-Methode [171] und das Finite Element Tearing and Interconnecting (FETI) [172]. Bei diesen nicht iterativen Kopplungsverfahren werden an den Rändern der Subgebiete Zusatzbedingungen definiert, sodass die Teilergebnisse unmittelbar konform sind. Für die Kopplung zu einer Gesamtgleichung können beispielsweise Lagrange-Multiplikatoren [173] eingesetzt werden. Im Gegensatz zu den iterativen Kopplungsverfahren ist die resultierende Lösung bereits nach einmaliger Berechnung für das gesamte Rechengebiet gültig.

6.1.2 Dekomposition der Zeit

Neben der Dekomposition des Raumes lässt sich auch die Zeit in separate Teilbereiche zerlegen. Besonders bei zeitintensiven Simulationen kann eine parallele Verarbeitung der Teilintervalle die Rechenressourcen entlasten und zu einer erheblichen Beschleunigung der Laufzeit führen [167]. So erlaubt der Parareal-Algorithmus [174] die Verknüpfung verschiedener *Solver*, wobei deren jeweilige Stärken gezielt ausgenutzt werden.

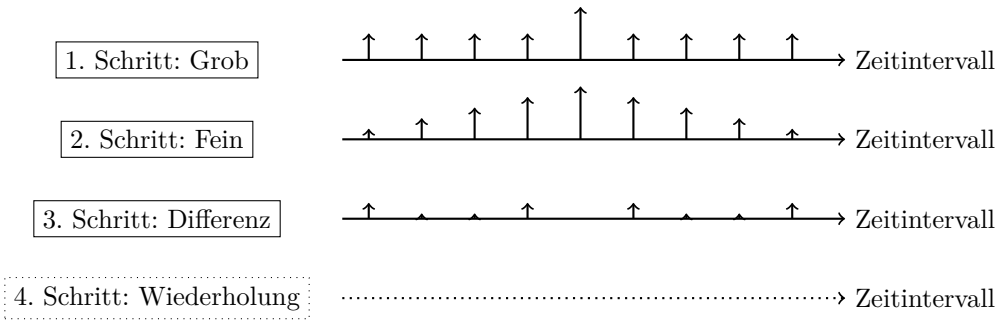


Abbildung 6.2: Illustration der Dekomposition der Zeit.

Im ersten Schritt besteht das Ziel darin, die Anfangsbedingungen aller Zeitintervalle schnell und grob anzunähern. Dies geschieht zunächst seriell. Dazu eignen sich *Solver* wie das explizite Euler-Verfahren [175] oder das Runge-Kutta (RK)-Verfahren niedriger Ordnung [176], die mit großen Zeitschrittweiten arbeiten und eine geringe Komplexität besitzen. Im zweiten Schritt wird die Genauigkeit der Berechnung in jedem Teilintervall erhöht, wobei eine längere Rechenzeit mit deutlich kleinerer Zeitschrittweite in Kauf genommen wird. Zu diesem Zweck ist eine Parallelisierung der Teilauswertungen in den voneinander unabhängigen Teilintervallen erstrebenswert.

Auch hier kann das RK-Verfahren eingesetzt werden, aber mit einer höheren Ordnung als im ersten Schritt. Der gesamte Vorgang wird so lange iterativ wiederholt, bis die Differenz zwischen den groben und feinen Teilergebnissen in den Teilintervallen über zwei aufeinanderfolgende Iterationen hinreichend klein ist [177]. Die Dekomposition der Zeit ist exemplarisch in der Abbildung 6.2 gezeigt.

6.1.3 Splitting des Operators

Mit Hilfe des *Splittings* kann der Operator zur Lösung einer PDGL in mehrere einfachere Teiloperatoren zerlegt werden, was zu einer Reduktion und effizienteren Verteilung der Rechenlast führt. Innerhalb eines festen Zeitintervalls werden die Teiloperatoren separat ausgewertet und anschließend über gemeinsame Anfangsbedingungen wieder gekoppelt. Die Teiloperatoren können unterschiedliche physikalische Eigenschaften aufweisen. Im Kontext der Maxwell-Gleichungen ist die Separierung der $\vec{E}$- und $\vec{H}$-Feldkomponente denkbar. Das *Splitting* des Operators lässt sich in iterative, additive und multiplikative Ansätze kategorisieren [80]. Im Rahmen dieser Arbeit stehen insbesondere die multiplikativen *Splittingverfahren* im Fokus, da der Matrixexponential-Ansatz zur Lösung der PDGLs angewandt wird.

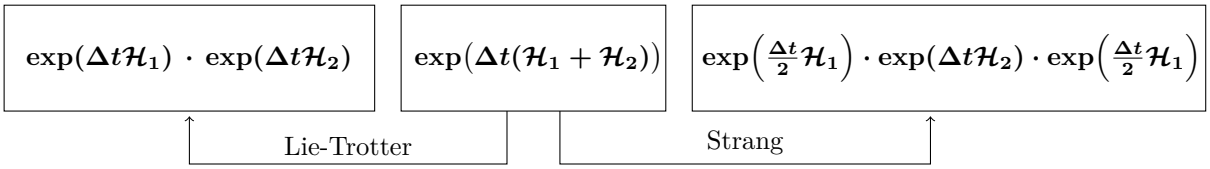


Abbildung 6.3: Illustration des *Splittings* des Operators.

In diesem Zusammenhang haben sich das Lie-Trotter- [178, 179] und das Strang-Verfahren [180] als bewährte Vertreter etabliert, im Wesentlichen aufgrund ihrer einfachen Implementierung und zuverlässigen Stabilität [181]. In [80] werden diese Verfahren als exponentielle Verfahren klassifiziert, da sie auf der Anwendung von Exponentialoperatoren basieren. Generell nutzt das Lie-Trotter-Verfahren die Struktur der Exponentialfunktion, indem das Matrixexponential eines zusammengesetzten Operators gemäß der Additionsregel formal in das Produkt der Exponentialfunktionen seiner Teiloperatoren zerlegt wird. Das Matrixexponential wird durch die sequentielle Anwendung der Exponentialoperatoren der Teiloperatoren approximiert. Dahingegen gestaltet sich die Implementierung des Strang-Verfahrens komplexer, weil zusätzlich halbe Zeitschrittweiten verwendet werden, jedoch ermöglicht dies im Vergleich zum Lie-Trotter-Verfahren eine höhere Fehlerordnung. In der Abbildung 6.3 ist das Konzept des *Operatorsplittings* veranschaulicht.

6.2 Gitterstrukturen

Die bisherigen Untersuchungen im Zuge der Approximation des Matrixexponentials mittels der Faber-Methode wurden ausschließlich unter der Annahme eines räumlich homogen diskretisierten Rechengebiets gemäß Yee durchgeführt. Dies trägt dazu bei, die Implementierung erheblich zu vereinfachen. Die Kehrseite ist allerdings, dass die Approximation im ganzen Rechengebiet

mit demselben Rechenaufwand durchgeführt wird. Oft sind es nur Subbereiche, welche diesen Aufwand tatsächlich erfordern. Trotzdem werden die verbleibenden Bereiche aufgrund der global festgelegten Auflösung mit dem gleichen Rechenaufwand behandelt [182]. Vor diesem Hintergrund werden im Folgenden zwei orthogonale Diskretisierungsansätze erläutert, die als Erweiterung zur Diskretisierung mit regulären Yee-Zellen mit äquidistantem Zellabstand entwickelt wurden und darauf abzielen, die Komplexität entsprechend der Auflösung anzupassen.

6.2.1 Nichtuniforme Struktur

Mittels der nichtuniformen Diskretisierung können Propagatoren in komplexen Geometrien effizienter und flexibler evaluiert werden, insbesondere bei gekrümmten Rändern oder Materialübergängen [79]. Die Umsetzung der nichtuniformen Diskretisierung ist verhältnismäßig simpel, da lediglich eine globale und davon abweichende lokale Diskretisierungsweiten definiert werden müssen. Dadurch kann die Anzahl der benötigten Yee-Zellen und Rechenoperationen deutlich gesenkt werden. Allerdings kann aufgrund der nicht äquidistanten Abstände zwischen den Yee-Zellen nicht mehr der zentrale Differenzenquotient zur Berechnung der örtlichen Ableitung der Maxwell-Gleichungen verwendet werden, was zu einer unvermeidlichen Zunahme des Approximationsfehlers führt. Obwohl nun ein lokaler Fehler der ersten Ordnung auftritt, bleibt der globale Fehler der zweiten Ordnung erhalten, wie in [183] bewiesen wurde. Die Abbildung 6.4 demonstriert die nichtuniforme Diskretisierung.

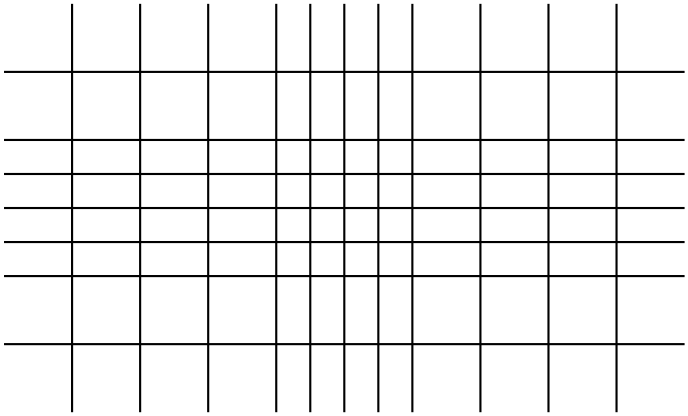


Abbildung 6.4: Illustration einer nichtuniformen Gitterstruktur.

Durch die Verwendung eines nichtuniformen Gitters ergibt sich neben einer globalen auch eine lokale Zeitschrittweite. Gemäß der CFL-Bedingung verringert sich die maximal zulässige Zeitschrittweite, um numerische Instabilitäten zu vermeiden. Was auf den ersten Blick wie ein Nachteil wirkt, wird im Rahmen von Local Time-Stepping (LTS)-Verfahren gezielt ausgenutzt. Die Grundidee besteht darin, einen zeitlichen Integrationsablauf mit unterschiedlich vielen Auswertungen in den Subgebieten zu realisieren. Angenommen, die grobe Diskretisierungsweite sei doppelt so groß gewählt wie die feine, so müsste der Propagator im feinen Subgebiet bei gleicher globaler Zeitschrittweite doppelt so häufig ausgewertet werden wie im groben Subgebiet. Weil sich die Anzahl der Yee-Zellen im feinen Subgebiet nur auf einen Teil des gesamten Gitters beschränkt, führt dies dennoch zu einer insgesamt reduzierten Komplexität. Damit die

Teilergebnisse mit der globalen und lokalen Zeitschrittweite korrekt gekoppelt werden, ist eine kontinuierliche zeitliche Synchronisation erforderlich [184]. Die an der Schnittstelle auftretende numerische Dispersion infolge unterschiedlicher Diskretisierungsweiten kann durch etablierte Verfahren wie Finite-Difference (FD) höherer Ordnung [9], Adaptive Mesh Refinement (AMR) [185] oder Interpolation Methods (IM) [186] gemindert werden.

6.2.2 Subgridding-Struktur

Ein alternativer Ansatz zur nichtuniformen Gitterstruktur, um gekrümmte Ränder oder Materialübergänge in komplexen Systemen lokal höher aufzulösen, ohne die Rechenzeit drastisch zu erhöhen, ist die *Subgriddingmethode*. Anstatt die Diskretisierungsweite über das gesamte Rechengebiet hinweg zu variieren, werden bei der *Subgriddingstruktur* die Yee-Zellen selbst in kleinere Subzellen unterteilt, um lokale Subgitter mit höherer Auflösung zu erzeugen. Der Informationsaustausch zwischen den Subzellen erfolgt in ähnlicher Weise wie der zwischen den ursprünglichen Yee-Zellen. Die Abbildung 6.5 illustriert die *Subgriddingstruktur*.

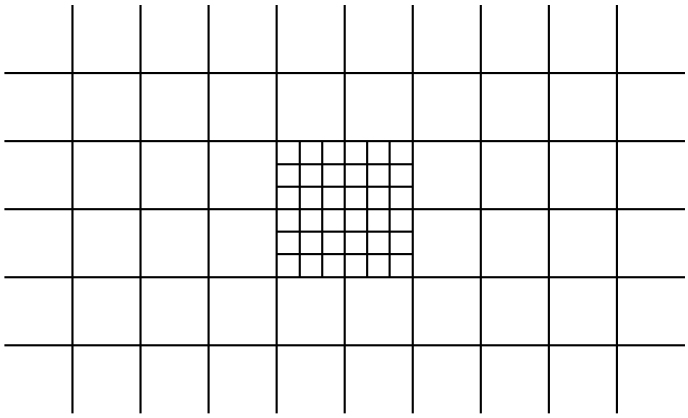


Abbildung 6.5: Illustration einer *Subgriddingstruktur*.

Aufgrund der unterschiedlichen Auflösungen der Gitter kann nicht mehr der zentrale Differenzenquotient zur örtlichen Ableitung der Maxwell-Gleichungen angewandt werden. Ferner ist aus Stabilitätsgründen die Modifizierung der CFL-Bedingung notwendig. Daher sind zusätzliche Methoden für die örtliche und zeitliche Kopplung der Gitter vonnöten. Dennoch kann im einfachsten Fall die globale CFL-Bedingung so angepasst werden, dass sie mit der kleinsten Diskretisierungsweite im Gitter übereinstimmt, was jedoch nicht im Sinne einer effizienten Rechenzeit ist [79]. Im Normalfall konzentrieren sich die anspruchsvolleren IM auf die Kopplung an den Schnittstellen zwischen den Gittern, die zudem zeitlich synchronisiert werden muss [71]. Ungeachtet der Fortschritte im Bereich der IM wurde in [187] gezeigt, dass keine der bekannten IM als vollständig stabil zu erachten ist. Als Ursache wird angeführt, dass evaneszente Wellen, die im groben Gitter abklingen sollten, infolge der ungleichmäßigen Diskretisierung im feineren Gitter weiter propagieren können [188]. In [186] wurden die Verfahren hinsichtlich ihrer Genauigkeit, Effizienz, Stabilität, allgemeinen Anwendbarkeit und Komplexität bewertet. Die Analyse machte deutlich, dass ein vollständiger Ausgleich dieser Eigenschaften nicht möglich ist und somit grundlegend abgewogen werden muss. Generell eignen sich in einem Rechengebiet mit einer *Subgriddingstruktur* sowohl das AMR- als auch LTS-Verfahren.

6.3 Einführung in die lokalen Operatoren

Im Kontext der Faberpolynome wird ein neuer Ansatz erforscht. Der Ausgangspunkt sei eine Evaluierung, bei der Ω unterschiedlich diskretisiert wird, sodass neue Subgebiete entstehen. Weiterhin sei angenommen, dass bislang ein einzelner globaler Operator mit dem Zustandsvektor $\vec{\Psi}$ für die Approximation verwendet wurde. Entsprechend der Subgebiete sollen nun separate Evaluierungen stattfinden. Anstelle eines globalen Operators liegt die Motivation in der Entwicklung mehrerer lokaler Operatoren, welche die Genauigkeit beibehalten, aber die Rechenzeit der Approximation reduzieren. Konzeptionell ist der Ansatz der lokalen Operatoren erstmals in [PS3] analysiert und publiziert worden. Zur Klarheit werden das Beispiel sowie Teilergebnisse aus [PS4] angeführt und in der Abbildung 6.6 vereinfacht illustriert, um die erwarteten Vorteile und Herausforderungen des Ansatzes zu verdeutlichen. Ferner werden die Publikationen [PS3] und [PS4] nachfolgend mit Darstellungen und Analysen ergänzt. Diesbezüglich soll Ω ein 1D-System mit N_x Diskretisierungspunkten repräsentieren.

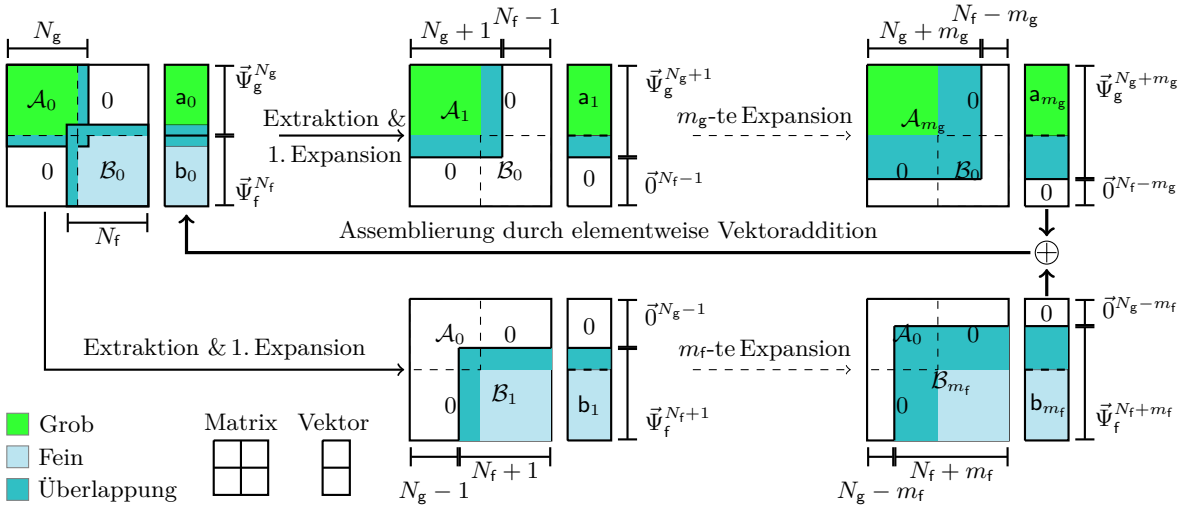


Abbildung 6.6: Das Konzept der lokalen Operatoren wird anhand der Kopplung zwischen zwei Feldkomponenten bei einem 1D-System visualisiert, welches auf ein 2D-System und 3D-System übertragen werden kann. Der Ausgangspunkt ist die Systemmatrix $\mathcal{H}$, die das Rechengebiet Ω beschreibt. Wenn Bereiche von Ω mit einer gröberen und einer feineren Diskretisierung abgetastet werden, resultieren die Submatrizen $\mathcal{A}_\mathsf{g}$ und $\mathcal{B}_\mathsf{f}$. Dementsprechend werden die Subzustandsvektoren $\vec{\Psi}_\mathsf{g}^{N_\mathsf{g}}$ und $\vec{\Psi}_\mathsf{f}^{N_\mathsf{f}}$ gebildet. Für die Auswertung mit den lokalen Operatoren werden jeweils ein Paar der Submatrizen und Subzustandsvektoren extrahiert und anschließend separat entwickelt. Der Entwicklungsprozess wird über eine Expansion realisiert, wobei zu beachten ist, dass die Elemente in den Submatrizen statisch bleiben. Aus Kompatibilitätsgründen werden beim Subzustandsvektor Nullen angehängt. Die Parameter a und b sind die mit den Submatrizen korrespondierenden Einträge des Zustandsvektors bei der Entwicklungsordnung m. Nachdem die Entwicklungen vollendet sind, wird der Zustandsvektor $\vec{\Psi}$ assembliert, indem die Subzustandsvektoren elementweise addiert werden. Diese Art der Entwicklung wird im Folgenden als dynamische Implementierung definiert.

Konventionell erfolgt die Approximation mit $\vec{\Psi}^{N_x}$, wobei die Notation mit dem Exponenten der Lesbarkeit dient. Zudem wird Ω in zwei Subbereiche unterteilt. Ein Subbereich weist eine grobe

Diskretisierung auf, der andere eine feine. Dementsprechend müssen sowohl die Systemmatrix $\mathcal{H}$ als auch der Zustandsvektor $\vec{\Psi}$ aufgeteilt werden. Für den groben Subbereich ergeben sich die Submatrix ${\mathcal{A}_\mathsf{g}}^{N_\mathsf{g}\times N_\mathsf{g}}$ und der Subzustandsvektor $\vec{\Psi}_\mathsf{g}^{N_\mathsf{g}}$, analog dazu ${\mathcal{B}_\mathsf{f}}^{N_\mathsf{f}\times N_\mathsf{f}}$ und $\vec{\Psi}_\mathsf{f}^{N_\mathsf{f}}$ für den feinen Subbereich. Die Parameter N_g und N_f geben die Anzahl der Diskretisierungspunkte im groben bzw. feinen Subgebiet an, die mit den Indizes g und f gekennzeichnet sind. Außerdem gilt $N_x = N_\mathsf{g} + N_\mathsf{f}$.

An dieser Stelle liegt die Vermutung nahe, zwei voneinander entkoppelte Evaluationen umzusetzen und die Teilergebnisse anschließend zu superponieren. Dieses Vorhaben würde jedoch zu fehlerhaften Resultaten führen aufgrund der bestehenden Kopplung zwischen den beiden Submatrizen $\mathcal{A}_\mathsf{g}$ und $\mathcal{B}_\mathsf{f}$. Präziser wird das Element in der letzten Spalte von $\mathcal{A}_\mathsf{g}$ demselben Diskretisierungspunkt zugeordnet wie dem Element in der ersten Spalte von $\mathcal{B}_\mathsf{f}$. Folglich kann das Superpositionsprinzip nicht wie erwartet angewandt werden. Um dennoch die Superposition theoretisch zu ermöglichen, müssten ${\mathcal{A}_\mathsf{g}}^{(N_\mathsf{g}+1)\times N_\mathsf{g}}$ und ${\mathcal{B}_\mathsf{f}}^{(N_\mathsf{f}-1)\times N_\mathsf{f}}$ initialisiert werden. Das bedeutet, dass $\mathcal{A}_\mathsf{g}$ und $\vec{\Psi}_\mathsf{g}^{N_\mathsf{g}}$ jeweils um den ersten Eintrag erweitert werden müssten, welcher dem feinen Gebiet zugeordnet ist. Jedoch ist die Kopplung für das Verfahren der lokalen Operatoren elementar, sodass stets eine quadratische Matrixordnung gegeben sein muss [PS3].

Beim Prozessablauf der lokalen Operatoren werden $\mathcal{A}_\mathsf{g}$ und $\mathcal{B}_\mathsf{f}$ nach der Initialisierung extrahiert und getrennt mit der Faberreihe entwickelt. Für den lokalen Operator f_g, der dem groben Subgebiet zugewiesen ist, folgt für $m = 1$ die Expansion ${\mathcal{A}_\mathsf{g}}^{N_\mathsf{g}\times N_\mathsf{g}} \rightarrow {\mathcal{A}_\mathsf{g}}^{(N_\mathsf{g}+1)\times(N_\mathsf{g}+1)}$. Hierbei stammen die hinzugefügten Zeilen- und Spalteneinträge aus den ursprünglich benachbarten Submatrizen. Insbesondere sind die Einträge von $\mathcal{B}_\mathsf{f}$ für die Expansion wesentlich, sodass ein Überlappbereich gebildet wird. Es wird betont, dass die Einträge der Submatrizen zu keinem Zeitpunkt durch die Expansion beeinflusst werden und konstant bleiben. Anders sieht dies bei $\vec{\Psi}^{N_\mathsf{g}}$ mit den Elementen a_0 bei $m = 0$ aus. Durch die MVM zwischen ${\mathcal{A}_\mathsf{g}}^{N_\mathsf{g}\times N_\mathsf{g}}$ und $\vec{\Psi}^{N_\mathsf{g}}$ wird im Rahmen der Faber-Methode das Ergebnis des Produkts mit a_1 festgehalten. Damit die Länge des expandierten Subzustandsvektors $\vec{\Psi}^{N_\mathsf{g}} \rightarrow \vec{\Psi}^{N_\mathsf{g}+1}$ mit der expandierten Submatrix ${\mathcal{A}_\mathsf{g}}^{(N_\mathsf{g}+1)\times(N_\mathsf{g}+1)}$ übereinstimmt, werden an das Vektorende entsprechend viele Nullen angehängt. Die Prozesse werden so lange wiederholt bis $m = N_{pol}$ gilt. Die Entwicklung des lokalen Operators f_f, der mit dem feinen Subgebiet verknüpft ist, erfolgt äquivalent dazu. Sobald beide Entwicklungen abgeschlossen sind, werden die Subzustandsvektoren elementweise zum Zustandsvektor des Zeitschritts durch eine elementweise Addition assembliert.

Der Hauptvorteil der Entwicklung mit lokalen Operatoren besteht darin, dass für die separaten Evaluationen unterschiedliche Entwicklungsordnungen gewählt werden können. Im Folgenden sei die Entwicklungsordnung in Bezug auf das grobe Subgebiet m_g und auf das feine Subgebiet m_f mit der Bedingung $m_\mathsf{f} \geq m_\mathsf{g}$. Basierend auf der Notation in [189] lässt sich mit dem erlangtem Wissen die korrekte Implementierung der Evaluation wie folgt erschließen [PS3]:

$$\vec{\Psi}_\mathsf{g}^{N_\mathsf{g}} \rightarrow \vec{\Psi}_\mathsf{g}^{(N_\mathsf{g}+m_\mathsf{g})} = \exp\left({\mathcal{A}_\mathsf{g}}^{(N_\mathsf{g}+m_\mathsf{g})\times(N_\mathsf{g}+m_\mathsf{g})}\right) \cdot \begin{bmatrix} \vec{\Psi}_\mathsf{g}^{N_\mathsf{g}} \\ \vec{0}^{m_\mathsf{g}} \end{bmatrix}, \tag{6.1}$$

$$\vec{\Psi}_\mathsf{f}^{N_\mathsf{f}} \rightarrow \vec{\Psi}_\mathsf{f}^{(N_\mathsf{f}+m_\mathsf{f})} = \exp\left({\mathcal{B}_\mathsf{f}}^{(N_\mathsf{f}+m_\mathsf{f})\times(N_\mathsf{f}+m_\mathsf{f})}\right) \cdot \begin{bmatrix} \vec{\Psi}_\mathsf{f}^{N_\mathsf{f}} \\ \vec{0}^{m_\mathsf{f}} \end{bmatrix}. \tag{6.2}$$

Einerseits ist die Entwicklungsordnung für die Laufzeit der Approximation entscheidend und andererseits für die Genauigkeit der zeitlichen Auflösung. Eine Verringerung der Entwicklungsordnung führt somit zu einer Reduktion der Laufzeit und der Auflösung. Mit den lokalen Operatoren

kann durch das Festlegen von $m_\mathsf{f} > m_\mathsf{g}$ dieser Umstand ausgenutzt werden. Während f_f weiterhin mit einer hohen Entwicklungsordnung ausgewertet wird, geschieht dies bei f_g mit einer reduzierten Ordnung. Dies kommt dadurch zustande, dass die skalierte Zeitschrittweite Δt_s proportional zum maximalen Eigenwert λ_s der skalierten Systemmatrix $\mathcal{H}_s$ ist, der wiederum antiproportional zur Diskretisierungsweite Δx ist. Dadurch konvergieren die Entwicklungskoeffizienten c_m mit einer verringerten Entwicklungsordnung N_{pol} gegen c_{th}. Für eine vereinfachte Darstellung der lokalen Operatoren wird auf den Abschnitt B.1 verwiesen.

Da die Entwicklungskoeffizienten c_m der lokalen Operatoren demselben Fehlerkriterium c_{th} unterliegen, hat die angepasste Entwicklungsordnung keinen Einfluss auf die Genauigkeit, sofern das Fehlerkriterium eingehalten wird. Zusätzlich ist keine Synchronisation der Zeitschrittweite vonnöten, wie es beispielsweise bei gängigen LTS-Verfahren der Fall ist. Trotzdem ist das Auftreten von numerischen Fehlern bei der Expansion von $\vec{\Psi}_\mathsf{g}^{N_\mathsf{g}}$ in das feine Subgebiet aufgrund der abweichenden Entwicklungsordnung nicht vollständig auszuschließen. In [190] wird deshalb vorgeschlagen die Expansion gezielt in einem Zwischenbereich zwischen den Subgebieten stattfinden zu lassen. Dabei soll die Expansion durch die Verschiebung der Entwicklungsordnung m_g primär über f_f durchgeführt werden, was zu den modifizierten Subzustandsvektoren $\vec{\Psi}_\mathsf{g}^{N_\mathsf{g}-m_\mathsf{g}} \rightarrow \vec{\Psi}_\mathsf{g}^{N_\mathsf{g}}$ und $\vec{\Psi}_\mathsf{f}^{N_\mathsf{f}+m_\mathsf{g}} \rightarrow \vec{\Psi}_\mathsf{f}^{N_\mathsf{f}+m_\mathsf{g}+m_\mathsf{f}}$ führt. Dadurch wird gewährleistet, dass $\vec{\Psi}_\mathsf{g}^{N_\mathsf{g}}$ nur noch in den definierten Zwischenbereich und nicht mehr in das feine Subgebiet expandiert. Die Abbildung 6.7 veranschaulicht die Situation ohne und mit Zwischenbereich.

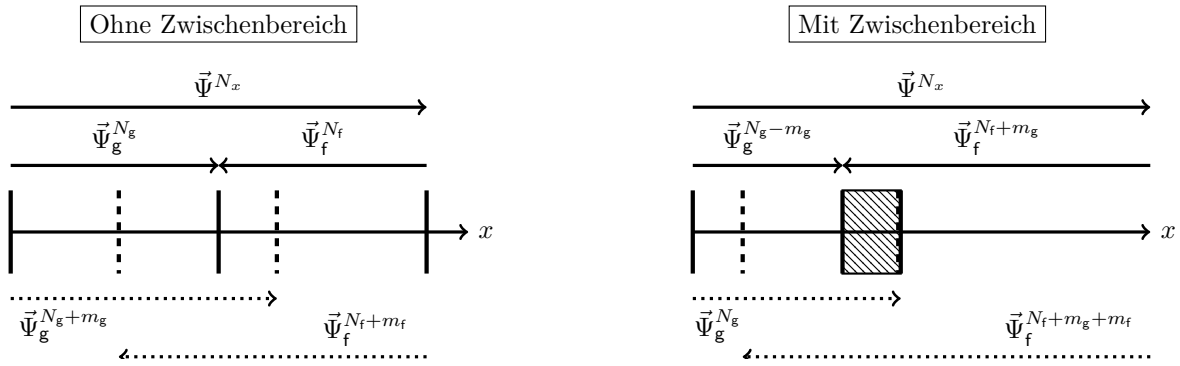


Abbildung 6.7: Expansion der Subzustandsvektoren ohne und mit Zwischenbereich.

In den Folgeuntersuchungen konnte trotz der Implementierung der Zwischenbereiche keine Veränderung der Genauigkeit festgestellt werden, dafür jedoch eine negative Beeinträchtigung der Laufzeit. Aus diesen Gründen wird dieser Optimierungsansatz nicht weiter berücksichtigt. Darüber hinaus bieten die lokalen Operatoren noch den Vorteil die Approximation mit der kompletten Systemmatrix $\mathcal{H}$ zu umgehen, da lediglich die extrahierten Submatrizen $\mathcal{A}_\mathsf{g}$ und $\mathcal{B}_\mathsf{f}$ für die Berechnung der rechenaufwändigen MVMs erforderlich sind. Zudem ist eine Parallelisierung des Prozessablaufs der lokalen Operatoren realisierbar, weil erst beim Assemblieren der Teilergebnisse die finale Kopplung vollzogen wird. Unter den diskutierten Dekompositions- und Splittingverfahren lassen sich die lokalen Operatoren am ehesten der in Abschnitt 6.1.1 beschriebenen Raumdekomposition zuordnen. Eine Dekomposition der Zeit erscheint in diesem Kontext nicht sinnvoll, da die zeitliche Entwicklung durch die Entwicklungsordnung der Faber-Methode gesteuert wird. Das *Operatorsplitting* wird im weiteren Verlauf separat behandelt.

6.3.1 Komplexität

In [PS3] wurde konzeptionell gezeigt, dass die Auswertung mit lokalen Operatoren potenziell schneller sein kann als mit einem globalen Operator, besonders für $\Delta t \gg \Delta t_{CFL}$. Aus diesem Grund wurde in der Publikation [PS4] die theoretische Komplexität für die Methode der lokalen Operatoren hergeleitet und mit praktischen Laufzeiten validiert, wie nachfolgend dargestellt. Als Rechengebiet Ω dient die Wellenleiterstruktur aus der Abbildung 6.8, welche für den 2D-Fall einen Filmwellenleiter mit drei *Layern* repräsentiert und bei der Erweiterung auf den 3D-Fall einen Rippenwellenleiter [191].

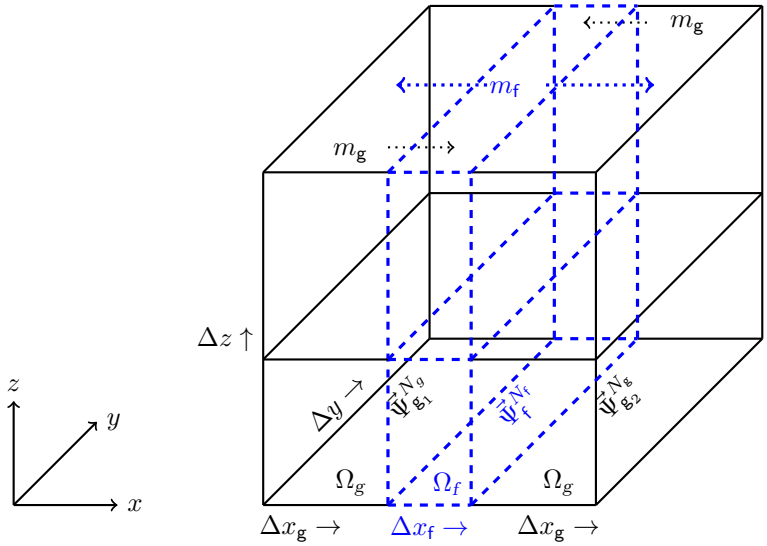


Abbildung 6.8: In dem Beispielrechengebiet wurde entlang der x-Achse mit Δx_g und Δx_f diskretisiert, sodass drei Subgebiete vorliegen. Der lokale Operator f_f für das feine Subgebiet Ω_f wird mit der Entwicklungsordnung m_f entwickelt, die lokalen Operatoren f_g der groben Subgebiete Ω_g mit m_g. Ferner sind die Subzustandsvektoren $\vec{\Psi}_{\mathsf{g}_1}^{N_\mathsf{g}}$, $\vec{\Psi}_{\mathsf{f}}^{N_\mathsf{f}}$ und $\vec{\Psi}_{\mathsf{g}_2}^{N_\mathsf{g}}$ abgebildet.

Um den potenziellen Vorteil der lokalen Operatoren zu demonstrieren, werden die zwei äußeren Subgebiete Ω_g entlang der x-Achse mit der groben Diskretisierungsweite Δx_g und das innere Subgebiet Ω_f mit der feinen Diskretisierungsweite Δx_f abgetastet sowie jeweils einem lokalen Operator bei der Approximation des Propagators zugeordnet. Die Konfiguration entspricht der nichtuniformen Gitterstruktur, die in dem Abschnitt 6.2.1 beschrieben wurde. Aus Gründen der Vergleichbarkeit zwischen den Dimensionen soll nur die Diskretisierung in x-Richtung variabel sein, weshalb $\Delta x_\mathsf{g} \geq \Delta x_\mathsf{f} = \Delta y = \Delta z$ angenommen wird. Diese Maßnahme mindert die aufkommenden numerischen Fehler in y- und z-Richtung. In diesem Zusammenhang wird auf [192] verwiesen, wo ein ähnliches System hinsichtlich des örtlichen Fehlers untersucht wurde. Die Quintessenz war, dass trotz einer lokal niedrigeren Auflösung im Vergleich zur globalen Diskretisierung die Genauigkeit der globalen Diskretisierung erhalten bleibt und eine Suprakonvergenz festgestellt werden konnte.

Gemäß [188] werden mit dem Refinementfaktor $R_{\Delta x} = \frac{\Delta x_\mathsf{g}}{\Delta x_\mathsf{f}}$ die relevanten Diskretisierungsweiten miteinander in Beziehung gesetzt. Weil nur Δx_g erhöht wird, reduziert sich der maximale Eigen-

wert λ_s in der entsprechenden Submatrix. Dadurch ist sichergestellt, dass das Eigenwertspektrum von der konformen Abbildungen stets umschlossen und die Stabilität nicht beeinträchtigt wird. Darüber hinaus führt eine Erhöhung von Δx_g zu einer Verringerung der benötigten Faberpolynome F_m für die Auswertung im grob diskretisierten Subgebiet, was zu einer Reduzierung der Rechenzeit im Vergleich zum fein diskretisierten Subgebiet führt, wie bereits im Abschnitt 6.3 erläutert. Damit der Vergleich trotz der Modifikation von Δx_g objektiv bleibt, wird die Länge des Rechengebiets in x-Richtung unabhängig von Δx_g als konstant vorausgesetzt. Zur Vereinfachung der Darstellung wird die Anzahl der Diskretisierungspunkte in den äußeren Subgebieten mit $N_{\mathsf{g}1} = N_{\mathsf{g}2} = N_\mathsf{g}$ gleichgestellt. Somit ergeben sich insgesamt $N = 2N_\mathsf{g} + N_\mathsf{f}$ Diskretisierungspunkte im System. Ergänzend dazu müssen die Abstände zwischen den einzelnen Abtastpunkten in den groben Subgebieten proportional zur Erhöhung von Δx über $R_{\Delta x} \in \mathbb{N}$ zunehmen, sodass die Länge von Ω unbeeinflusst bleibt. Entsprechen $\tilde{N}_\mathsf{g}$ und $\Delta\tilde{x}_\mathsf{g}$ den jeweiligen angepassten Größen, so folgt

$$N_\mathsf{g} \cdot \Delta x_\mathsf{g} = \left(\frac{N_\mathsf{g}}{R_{\Delta x}}\right) \cdot (R_{\Delta x} \cdot \Delta x_\mathsf{g}) = \tilde{N}_\mathsf{g} \cdot \Delta\tilde{x}_\mathsf{g}. \tag{6.3}$$

Das Verhältnis zwischen N_g und Δx_g in der Bedingung aus der Gleichung (6.3) bleibt auch unter dem Einfluss von $R_{\Delta x}$ unverändert. Zur Klarheit demonstriert die Abbildung 6.9 den Einfluss des Refinementfaktor auf die groben Subgebiete für $R_{\Delta x} \geq 1$.

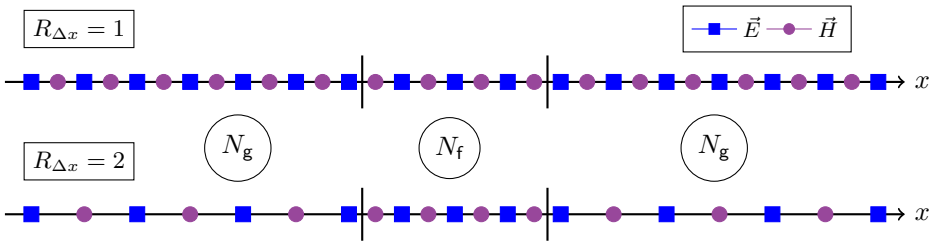


Abbildung 6.9: Bei $R_{\Delta x} = 1$ wird mit äquidistanten Abtastpunkten diskretisiert. Bei $R_{\Delta x} = 2$ wird N_g halbiert, dafür verdoppelt sich der Abstand zwischen den Punkten in den groben Subgebieten. Das feine Subgebiet bleibt unbeeinflusst von $R_{\Delta x}$ und die Länge des Rechengebiets bleibt konstant.

Um die Komplexitätsanalyse so objektiv wie möglich zu gestalten, wird für den globalen Operator $N_{pol} = m_\mathsf{f}$ vereinbart. Dadurch wird garantiert, dass die Evaluationen von f und die von f_f gleich behandelt werden aufgrund derselben Diskretisierungsweite Δx_f. Des Weiteren soll die Genauigkeit bei den lokalen Operatoren auch bei $R_{\Delta x} > 1$ in derselben Größenordnung liegen wie bei $R_{\Delta x} = 1$. Aus diesem Grund wird für die Entwicklungskoeffizienten der lokalen Operatoren das Fehlerkriterium $c_{th} = 10^{-15} > |c_m|$ definiert, wodurch f_g einen Fehler in derselben Größenordnung wie f_f aufweist.

In [19] wurde die Laufzeit der Faber-Methode über die Anzahl der MVMs sowie der Entwicklungsordnung N_{pol} abgeschätzt, wobei eine statische Matrixordnung vorausgesetzt wurde. Im Falle der lokalen Operatoren werden die Submatrizen extrahiert, die während der Entwicklung expandieren. Hier treten dynamische Größenordnungen der Submatrizen auf. Zu diesem Zweck wird zunächst die Anzahl der für die Evaluierung eines Zeitschritts mit der konventionellen Faber-Methode erforderlichen Multiplikationen als Referenzgröße herangezogen, anstelle der direkten Betrachtung von MVMs. Sofern die Systemmatrix $\mathcal{H}^{N\times N}$ als Bezugsgröße dient, sind

die absoluten Kosten für den Referenzfall proportional zur Matrixordnung N^2. Die resultierende Kostenfunktion für den globalen Operator C_{glob} ist einfach abzuleiten, da nur Matrizen der Ordnung N^2 im Verlauf der Approximation berücksichtigt werden müssen, bis $N_{pol} = m_\mathsf{f}$ erreicht ist. Mit der Normierung auf den Referenzfall folgt

$$C_{glob}(m_\mathsf{f}) = \frac{m_\mathsf{f} \cdot N^2 \cdot D}{N^2} = m_\mathsf{f} \cdot D. \tag{6.4}$$

In der aktuellen Analyse der Komplexität wird der Übersicht halber von einem 1D-System ausgegangen. Dennoch kann die Komplexität über den Skalierungsparameter D auch für höhere Dimensionen mühelos ermittelt werden. Bei den lokalen Operatoren sind aufgrund der extrahierten Submatrizen und der reduzierten Entwicklungsordnung $m_\mathsf{g} \leq m_\mathsf{f}$ der lokalen Operatoren der äußeren Subgebiete geringere Kosten zu erwarten. In Rahmen dieser Studie setzt sich die gesamte Kostenfunktion C_{lok} für die lokalen Operatoren aus der Summe der einzelnen Kostenfunktion zusammen:

$$C_{lok}(m_\mathsf{f}, m_\mathsf{g}) = 2 \cdot C_{grob}(m_\mathsf{g}) + C_{fein}(m_\mathsf{f}). \tag{6.5}$$

Bedingt durch die Vereinfachung, beide groben Subgebiete mit derselben Anzahl an Diskretisierungspunkten N_g zu versehen, tritt in der Gleichung (6.5) der Koeffizient *2* vor der Kostenfunktion der groben Subgebiete C_{grob} auf. Die Kostenfunktion C_{glob} wird wiederum durch eine Aufsummierung der dynamischen Ordnungen der expandierten Submatrizen aufgestellt:

$$C_{grob}(m_\mathsf{g}) = \frac{D}{N^2} \cdot \sum_{m=0}^{m_\mathsf{g}} (N_\mathsf{g} + m)^2. \tag{6.6}$$

Bereits für $m = 0$ liegt in der Gleichung (6.6) eine Kopplung zwischen den Subgebieten vor, wie die Abbildung 6.6 verdeutlichte, wobei mit zunehmendem m immer mehr Elemente der Submatrizen in die Kopplung mit einfließen. In gleicher Weise wird die Kostenfunktion C_{fein} aus der Gleichung (6.5) definiert:

$$C_{fein}(m_\mathsf{f}) = \frac{D}{N^2} \cdot \sum_{m=0}^{m_\mathsf{f}} (N_\mathsf{f} + 2m)^2. \tag{6.7}$$

Im Vergleich zu C_{grob} aus der Gleichung (6.6) ist in C_{fein} aus der Gleichung (6.7) der Koeffizient *2* vor der Entwicklungsordnung m zu finden. Die Ursache dafür liegt an der Expansion von f_f, welche in beide groben Subgebiete erfolgt wie die Abbildung 6.8 zeigt. Auch hier muss bereits bei $m = 0$ die Kopplung für die Methode der lokalen Operatoren präsent sein. Die bislang hergeleitete Komplexität für die Anwendung der lokalen Operatoren ist so definiert worden, dass die Kostenfunktion minimiert wird.

Obwohl dies theoretisch vorteilhaft erscheint, kann sich in der Praxis ein alternativer Ansatz als effizienter erweisen. Daher wird C_{lok} aus der Gleichung (6.5) künftig als unteres theoretisches Limit betrachtet, während nachfolgend ein oberes theoretisches Limit $\tilde{C}_{lok}$ bestimmt wird. Der alternative Ansatz sieht vor, die Expansion der Submatrizen vollständig zu vermeiden, sodass ausschließlich die Subzustandsvektoren dynamisch angepasst werden müssen. Die Idee besteht darin, bei jeder Entwicklungsordnung m die vollständig expandierten Submatrizen in der Evaluierung zu verwenden, die ursprünglich erst bei $m = m_\mathsf{f}$ bzw. $m = m_\mathsf{g}$ vorlagen. Das Konzept ist in der Abbildung 6.10 illustriert. Formal wird die Gleichung (6.5) wie folgt angepasst:

$$\tilde{C}_{lok}(m_\mathsf{f}, m_\mathsf{g}) = 2 \cdot \tilde{C}_{grob}(m_\mathsf{g}) + \tilde{C}_{fein}(m_\mathsf{f}). \tag{6.8}$$

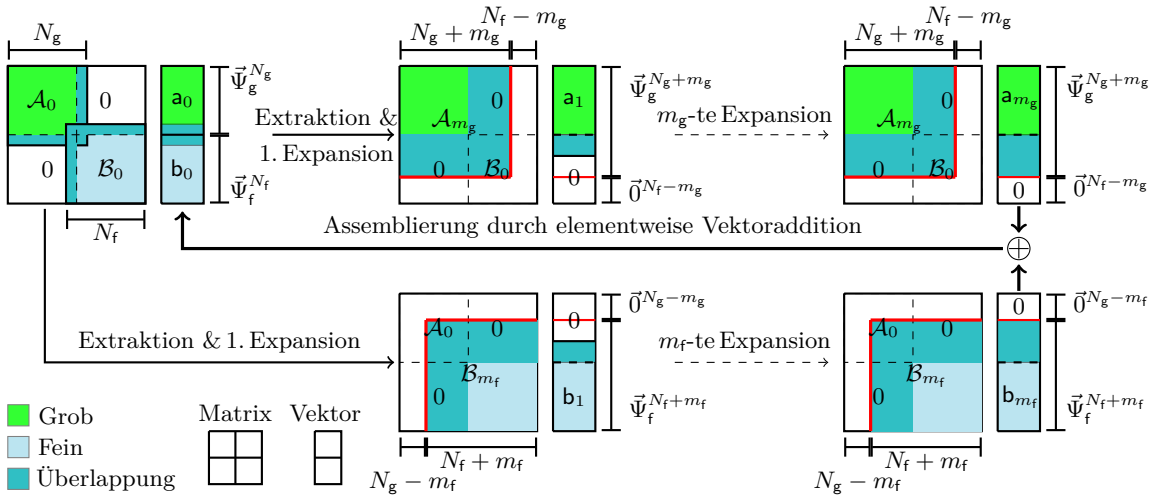


Abbildung 6.10: Illustration der statischen Implementierung der lokalen Operatoren. Der Unterschied zur dynamischen Implementierung aus der Abbildung 6.6 liegt in der Matrixordnung der extrahierten Submatrizen für jede Entwicklungsordnung, welche stets konstant bleibt.

Dementsprechend resultiert für die Gleichung (6.6)

$$\tilde{C}_{grob}(m_g) = \frac{D}{N^2} \cdot \sum_{m=0}^{m_g} (N_g + m_g)^2 = \frac{D}{N^2} \cdot m_g \cdot (N_g + m_g)^2 \tag{6.9}$$

sowie für die Gleichung (6.7)

$$\tilde{C}_{fein}(m_f) = \frac{D}{N^2} \cdot \sum_{m=0}^{m_f} (N_f + 2m_f)^2 = \frac{D}{N^2} \cdot m_f \cdot (N_f + 2m_f)^2. \tag{6.10}$$

Aus theoretischer Sicht erhöht dieser Ansatz die Komplexität erheblich, weil mit jeder Entwicklungsordnung m derselbe Aufwand betrieben werden muss, der beim vorherigen Ansatz erst bei der finalen Entwicklungsordnung $m_f = m_g = N_{pol}$ nötig ist. Aus praktischer Sicht kann durch diese Maßnahme die Rechenzeit reduziert werden, da die Submatrizen nur ein Mal initialisiert sowie im weiteren Verlauf nicht mehr angepasst werden müssen. Es gilt somit zu überprüfen, ob die Erweiterung der Submatrizen oder die Anzahl der Multiplikationen die Laufzeit dominiert.

6.3.2 Untersuchung der theoretischen Ergebnisse

Zur Analyse der zuvor definierten Komplexitäten werden drei Konstellationen k definiert, um den Einfluss des Zeitfaktors q und des Fehlerkriteriums c_{th} auf die Kostenfunktionen C_{glob} und C_{lok} beobachten zu können. Die Konstellation werden definiert mit $\mathsf{k} = 1 \rightarrow \{q = 1 \wedge c_{th} = 10^{-6}\}$, mit $\mathsf{k} = 2 \rightarrow \{q = 10 \wedge c_{th} = 10^{-6}\}$ und mit $\mathsf{k} = 3 \rightarrow \{q = 1 \wedge c_{th} = 10^{-15}\}$, die in der Tabelle 6.1 zusammengefasst sind. In dem Abschnitt 4.1 wurde als Fehlerkriterium $c_{th} = 10^{-15}$ angegeben, um die Genauigkeit der Approximation adäquat zu gewährleisten. In dieser Untersuchung liegt der Fokus speziell auf der Reduktion der Laufzeit, weshalb im Referenzfall $\mathsf{k} = 1$ die Schwelle auf $c_{th} = 10^{-6}$ erhöht wird. Prinzipiell kann bei $\mathsf{k} = 1$ von der minimalen Komplexität ausgegangen

Tabelle 6.1: Einfluss des Refinementfaktors auf die Entwicklungsordnung.

				$R_{\Delta x}$									
k	q	c_{th}	m	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1	**1**	10^{-6}	m_f					11					
			m_g	11	9	8	7	7	7	6	6	6	6
2	**10**	10^{-6}	m_f					36					
			m_g	36	23	19	16	14	13	13	12	11	11
3	**1**	10^{-15}	m_f					19					
			m_g	19	15	14	13	12	12	11	11	11	11

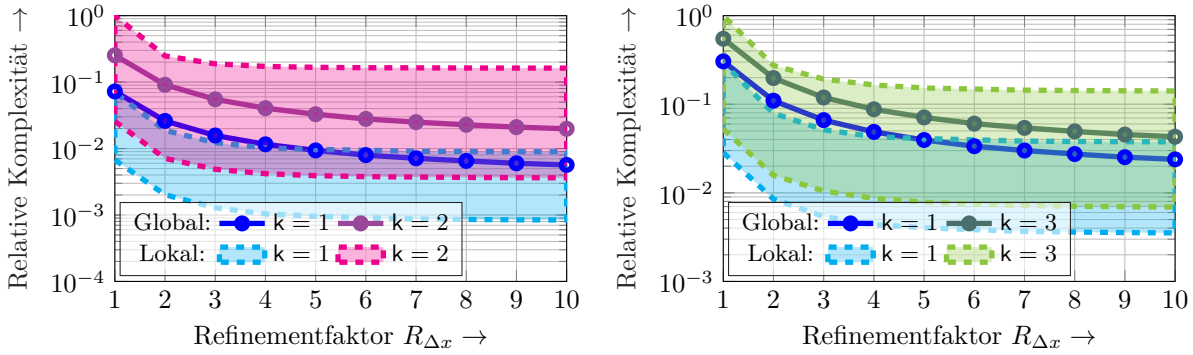
werden, da jede Erhöhung von q oder Absenkung von c_{th} eine höhere Polynomordnung N_{pol} nach sich zieht. Es sollte beachtet werden, dass c_{th} nur als Fehlerkriterium der Faberreihe genutzt wird und nicht die tatsächliche Genauigkeit der Approximation widerspiegelt. Des Weiteren wird bei $\mathsf{k} = 2$ nur ein Zeitfaktor von $q = 10$ festgelegt, obwohl die Faber-Methode bedeutend höhere Werte für q ermöglicht. Dies wird damit begründet, dass ein höherer q-Wert zu einer höheren Entwicklungsordnung N_{pol} führt.

Im Kontext der lokalen Operatoren muss berücksichtigt werden, dass die Entwicklungsordnung nicht beliebig hoch gewählt werden darf, weil andernfalls die Expansion über die Ränder von Ω in undefinierte Bereiche hinausgeht, wie im Abschnitt B.1 thematisiert wird. Präziser ergibt sich $m_\mathsf{g} \leq m_\mathsf{f} = N_{pol} \leq N_\mathsf{g}$ als Bedingung. Der Refinementfaktor wird in dem Intervall $R_{\Delta x} \in [1, 10]$ definiert. Für $R_{\Delta x} = 1$ ist die Diskretisierung in ganz Ω homogen. Für $R_{\Delta x} > 1$ bleibt sie dies in Ω_f weiterhin, während in Ω_g die Abtastpunkte proportional zu $R_{\Delta x}$ abnehmen und die Abstände untereinander proportional zu $R_{\Delta x}$ zunehmen. Die Operatoren f und f_f werden unabhängig von $R_{\Delta x}$ entwickelt, sodass nur die Anpassung in Ω_g Einfluss auf den Vergleich hat. Augenscheinlich wirkt sich die Zunahme von $R_{\Delta x}$ auch auf die numerische Dispersion aus. Bekannte Verfahren zur Mitigierung dessen sind in dem Abschnitt 6.2 thematisiert worden. Weil die Reduktion der Komplexität bzw. der Laufzeit die höchste Priorität hat, wird die numerische Dispersion jedoch künftig nicht mehr beachtet.

Im Einklang mit dem Abschnitt 6.3.1 wird die Komplexität zwischen dem globalen Operator und der lokalen Operatoren verglichen, die erforderlich ist, um einen Zeitschritt zu propagieren. Diesbezüglich werden die Ergebnisse der lokalen Operatoren mit einer unteren und einer oberen Grenze illustriert. Erstes repräsentiert den dynamischen Ansatz mit C_{lok} aus der Gleichung (6.5) und letzteres den statischen Ansatz mit $\tilde{C}_{lok}$ aus der Gleichung (6.8), wodurch die Komplexität nicht als einzelner Wert, sondern als beschränktes Intervall zu verstehen ist.

Als Anzahl der Diskretisierungspunkte wird $N_x = N_y = N_z = 1000$ mit einer Diskretisierungsweite von $\Delta x_\mathsf{g} \geq \Delta x_\mathsf{f} = \Delta y = \Delta z = 10\,\mathrm{nm}$ vorgegeben, um eine praktische Implementierung des Systems im Hinblick auf die Speicheranforderungen zu ermöglichen. Hierbei ist Δx_f statisch und Δx_g dynamisch. Weiterhin wird für die Verteilung der Diskretisierungspunkte $\frac{N_\mathsf{g}}{N_x} = 0{,}4$ in Ω_g sowie $\frac{N_\mathsf{f}}{N_x} = 0{,}2$ in Ω_f festgelegt. Die elektromagnetische Welle wird mit $\vec{\Psi} = [E_x, E_y, H_z]^T$

im Zentrum von Ω initialisiert. Für die Startbedingung von H_z wird ein Gaußpuls mit einer Varianz von $\sigma_g = 200\,\mathrm{nm}$ gewählt und für die verbleibenden Komponenten gilt $E_x = E_y = 0$. Gemäß Yee werden E_x und E_y über die Kopplung mit H_z bestimmt. Die Komplexität wird in jedem Vergleich durch die Normierung mit der maximal vorkommenden Komplexität in einer Teiluntersuchung relativiert. Bei der Konstellation $\mathsf{k} = 2$ wird zudem die Komplexität durch den Faktor *zehn* dividiert, um einen objektiven Vergleich zu den anderen Konstellationen mit $q = 1$ zu schaffen. Die Resultate sind in der Abbildung 6.11 zu sehen.



(a) Komplexität für $q \in \{1, 10\}$ und $c_{th} = 10^{-6}$.

(b) Komplexität für $q = 1$ und $c_{th} \in \{10^{-6}, 10^{-15}\}$.

Abbildung 6.11: In der Abbildung 6.11a wird die Komplexität des Referenzfalls $\mathsf{k} = 1$ mit derjenigen der Konstellation $\mathsf{k} = 2$ gegenübergestellt und in der Abbildung 6.11b mit der Komplexität der Konstellation $\mathsf{k} = 3$. Das Resultat des globalen Operators wird mittels einer durchgezogenen Linie veranschaulicht, dagegen sind für die lokalen Operatoren aufgrund der Variation der Implementierung Intervalle zu sehen. Die untere Grenze repräsentiert C_{lok} und die obere $\tilde{C}_{lok}$.

Der erste Vergleich zwischen den Konstellationen $\mathsf{k} = 1$ und $\mathsf{k} = 2$ mit unterschiedlichem q offenbart in der Abbildung 6.11a, dass die Komplexität des globalen Operators bei $q = 10$ konstant etwa eine halbe Größenordnung höher als bei $q = 1$ ist. Dies ist auf die höhere Entwicklungsordnung zurückzuführen, die für die Auswertung eines Zeitschritts benötigt wird. Der konstante Offset ergibt sich aus der Unabhängigkeit der Entwicklungsordnung m_f von dem Refinementfaktor $R_{\Delta x}$, wie die Tabelle 6.1 zeigt. Somit wird die Senkung der Komplexität des globalen Operators ausschließlich durch die Abnahme von N_g bestimmt. Im Falle der lokalen Operatoren ist der Einfluss des höheren Zeitfaktors q deutlich ausgeprägter als zuvor. Sowohl die obere als auch die untere Grenze weichen zueinander und untereinander um mehr als eine Größenordnung ab, bedingt durch die situativ erforderliche Entwicklungsordnung. Ferner ist die Wirkung des Refinementfaktors unmittelbar durch die inhomogene Diskretisierung bei $R_{\Delta x} = 2$ zu erkennen, wo die Komplexität rapide um mehr als eine halbe Größenordnung abnimmt. Ab $R_{\Delta x} > 4$ ist der Einfluss des Refinementfaktors bzw. von f_g nicht mehr zu vernehmen. Es ist hauptsächlich f_f für die Komplexität von Bedeutung. Eine weitere nennenswerte Besonderheit zeigt der Vergleich zwischen den Operatoren auf. Während bei der Konstellation $\mathsf{k} = 1$ der globale Operator erst bei $q = 5$ die obere Grenze des lokalen Operators unterschreitet, ist dies bei $\mathsf{k} = 2$ stets zutreffend. Die Erklärung dafür ist die verhältnismäßig hohe Entwicklungsordnung für die Auswertung mittels der lokalen Operatoren der groben Subgebiete, je höher q gewählt wird.

In der zweiten Untersuchung werden die Konstellationen $\mathsf{k} = 1$ und $\mathsf{k} = 3$ verglichen. Hierbei ist q fix und der Schwellenwert auf $c_{th} \in \{10^{-6}, 10^{-15}\}$ beschränkt. Gemäß der Tabelle 6.1 ist die Erwartung, dass die Abweichungen deutlich geringer ausfallen als zuvor. Dies deckt sich mit der Abbildung 6.11b, in der alle *Offsets* ungefähr halbiert wurden. Es können fast alle Erkenntnisse des vorherigen Vergleichs übernommen werden. Ein Unterschied liegt jedoch in der Komplexität der lokalen Operatoren, die bis $R_{\Delta x} = 3$ stark abnimmt und ab $R_{\Delta x} = 6$ eine Sättigung erreicht. Allgemein ist durch die zwei Vergleiche festzuhalten, dass besonders die Komplexität des dynamischen Ansatzes der lokalen Operatoren signifikanter zunimmt, wenn ein höherer Zeitfaktor q verwendet wird.

6.4 Numerische Evaluierung in einem linearen System

Die theoretische Analyse der lokalen Operatoren im Abschnitt 6.3.2 zeigte, dass die Implementierung einen nicht zu unterschätzenden Einfluss auf die theoretische Komplexität hat. Allerdings lassen sich aus der hergeleiteten Komplexität keine direkten Rückschlüsse auf die tatsächliche Laufzeit ziehen. Insbesondere können die Laufzeiten für Hintergrundprozesse, wie der Allokation temporärer Speicherressourcen, der Erweiterung von Datenstrukturen wie Matrizen und Vektoren, des Datentransfers zwischen den Funktionen oder der unvermeidbaren Schwankungen der Performance durch die Hardware, nur theoretisch bzw. stark vereinfacht abgeschätzt werden. Generell kann der Ablauf der lokalen Operatoren beschrieben werden durch

$$\begin{aligned} Komplexität &= \{Extraktion + Expansion + Assemblierung\} \\ &= \{Expansion + Overhead\}, \end{aligned} \tag{6.11}$$

ungeachtet der Implementierung [PS4, PS5]. Der Hauptteil der Laufzeit für die Evaluierung mit Hilfe des globalen Operators entsprechend der Faber-Methode entfällt auf die Expansion. Bei den lokalen Operatoren gilt dies ebenso in der Gleichung (6.11). Dennoch sind hier die Extraktion und Assemblierung unerlässliche Prozesse im Ablauf der lokalen Operatoren, wie bereits die Abbildung 6.6 veranschaulichte. Um die Aufmerksamkeit gezielt auf die Expansion zu lenken, werden die Extraktion und Assemblierung als *Overhead* zusammengefasst. Im Folgenden wird die Laufzeit der Approximation des Propagators sowohl durch den globalen Operator als auch durch die lokalen Operatoren in einem linearen sowie in einem nichtlinearen System untersucht. Des Weiteren werden die Studien durch das Einbinden von Processing Units (PU) wie einer CPU und GPU ergänzt. Außerdem werden C und MATLAB als Vertreter einer niederen und hohen Programmiersprache betrachtet.

Tabelle 6.2: Relative Laufzeiten des dynamischen und statischen Ansatzes bei den lokalen Operatoren.

		$R_{\Delta x}$									
Ansatz	**Limit**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
Dynamisch	C_{lok}	1,00	0,52	0,41	0,35	0,31	0,29	0,27	0,26	0,25	0,24
Statisch	$\tilde{C}_{lok}$	0,22	0,12	0,10	0,08	0,08	0,07	0,07	0,06	0,06	0,06

Bevor jedoch mit der Auswertung der praktischen Studien begonnen wird, muss abschließend entschieden werden, ob der dynamische Ansatz aus der Gleichung (6.5) oder der statische Ansatz aus der Gleichung (6.8) bei den lokalen Operatoren zukünftig für einen objektiven Vergleich berücksichtigt werden soll. Aus diesem Grund wird in der folgenden Untersuchung die Rechenzeiten für die Approximation des zeitabhängigen Propagators in dem System von der Abbildung 6.6 mit denselben Parametern und Annahmen aus dem Abschnitt 6.3.2 für ein 2D-System analysiert. Die Resultate sind in der Tabelle 6.2 ersichtlich.

Im Gegensatz zu den theoretisch gewonnen Erkenntnissen spiegeln die praktisch ermittelten Laufzeiten eine andere Realität wider. War die Komplexität von C_{lok} stets geringer als die von $\tilde{C}_{lok}$, trifft auf die Laufzeit das Gegenteil zu. Im Durchschnitt ist die Berechnung mit $\tilde{C}_{lok}$ viermal langsamer als jene mit C_{lok}. Obwohl der statische Ansatz die Komplexität des *Overheads* minimiert und gleichzeitig die Komplexität der Expansion erhöht, stellt der *Overhead* dennoch die Hauptursache für die beträchtliche Laufzeit von C_{lok} dar. Zur Validierung dieses Resultats wurden empirische Studien mit einer variierenden Anzahl an Diskretisierungspunkten, einem 2D- und 3D-System sowie mit C und MATLAB durchgeführt, die alle zum selben Ergebnis führten, sodass $\tilde{C}_{lok}$ zu präferieren ist [PS4]. Damit ist validiert worden, dass im Rahmen der Evaluation die Anpassung der Matrizen mehr Rechenzeit erfordert als die Berechnung der Multiplikationen. In Anbetracht der Tatsachen wird in den praktischen Untersuchungen ausschließlich der statische Ansatz angewandt.

Die Untersuchung der Laufzeiten zur Approximation des Propagators in einem linearen System erfolgt analog zu der Komplexitätsanalyse aus dem Abschnitt 6.3.1 unter Verwendung desselben Rechengebiets aus der Abbildung 6.6, welches das Modellieren einer 2D- und 3D-Wellenleiterstruktur ermöglicht. Die Ergebnisse für die linearen Systeme sind in [PS3] und [PS4] publiziert und werden im Folgenden miteinander verknüpft sowie ausführlicher thematisiert. Durch die Betrachtung der unterschiedlichen Dimensionen soll die Auswertung auf eine allgemeingültige Basis gestellt werden, was theoretisch durch den Skalierungsparameter D abgefangen wurde. Generell ist der Ansatz der lokalen Operatoren nicht auf dieses spezifische Rechengebiet beschränkt, sondern kann auch in jedem beliebigen Rechengebiet genutzt werden. Die verwendeten Parameter sind mit jenen aus dem Abschnitt 6.3.2 abgestimmt.

Bei dem 3D-System wird der Zustandsvektor auf $\vec{\Psi} = [E_x, E_y, E_z, H_x, H_y, H_z]^T$ erweitert und als Startbedingung $E_x = E_y = E_z = H_x = H_y = 0$ sowie $H_z \neq 0$ definiert. Der Hauptfokus liegt auf der Erforschung des *Overheads* und dessen Einfluss auf die Laufzeit. Ähnlich wie zuvor werden die mittels der lokalen Operatoren generierten Laufzeiten jeweils mit einer unteren und oberen Grenze visualisiert. Der Unterschied besteht darin, dass fortan die untere Grenze die Rechenzeit für die reine Expansion und die obere Grenze die Rechenzeit für den gesamten Prozess inklusive der Extraktion und Assemblierung angibt. Zudem bleibt die Darstellung mit relativen Laufzeiten erhalten, die im Rahmen der Normierung auf die höchste vorkommende Laufzeit einer Teilstudie durchgeführt wird.

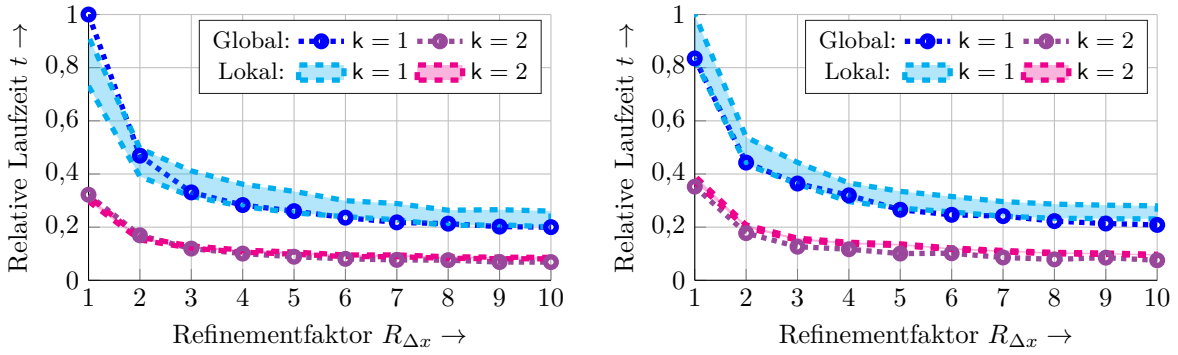
Tabelle 6.3: Hardwarespezifikationen für die linearen Berechnungen auf der CPU.

CPU	Taktrate	Cache	Kerne	RAM
AMD EPYC 7542	2,49 GHz	L3 mit 128 MB	20	1024 GB

Als Programmiersprache wird MATLAB R2024b verwendet, da sich die Faber-Methode bei der Expansion im Wesentlichen auf MVMs stützt. Standardmäßig werden die Berechnungen in MATLAB über Multithreads realisiert, wobei die Basic Linear Algebra Subprograms (BLAS)- und Linear Algebra Package (LAPACK)-Bibliothek ausgenutzt werden [193]. Für die Berechnungen wird auf einen HPC bzw. den Linux Cluster an der TU Dortmund der dritten Generation (LiDO3) [194] mit den Hardwarespezifikationen aus der Tabelle 6.3 zurückgegriffen [195]. Dies ermöglicht insbesondere die Simulation des 3D-Systems mit einer außergewöhnlich hohen Anzahl an Diskretisierungspunkten sowie die Anpassung von q und c_{th}. Auch wird darauf hingewiesen, dass sich das 3D-Systems nur aufgrund der dünnbesetzten Systemmatrix modellieren lässt.

6.4.1 Anpassung der Zeitfaktors

Bei der ersten praktische Laufzeituntersuchung sind der Zeitfaktor mit $q \in \{1, 10\}$ und der Schwellenwert mit $c_{th} = 10^{-6}$ festgelegt. Die Resultate sind in der Abbildung 6.12 dargestellt, wobei die Abbildung 6.12a die Laufzeiten im 2D-System und die Abbildung 6.12b die Laufzeiten im 3D-System zeigt. In beiden Fällen sind fast identische Verläufe zu sehen, jedoch lassen sich bei genauerer Betrachtung marginale Unterschiede erkennen. Eine Abweichung ist beim globalen Operator für $R_{\Delta x} \leq 2$ zu beobachten. Im 2D-System entspricht die Laufzeit für die Evaluation mittels des globalen Operators erst ab $R_{\Delta x} = 4$ in etwa der Expansionszeit der lokalen Operatoren.



(a) Laufzeiten mit $q \in \{1, 10\} \wedge c_{th} = 10^{-6}$ bei 2D.

(b) Laufzeiten mit $q \in \{1, 10\} \wedge c_{th} = 10^{-6}$ bei 3D.

Abbildung 6.12: Vergleich der relativen Laufzeiten der Operatoren für $\mathsf{k} = 1$ und $\mathsf{k} = 2$ in einem linearen 2D- und 3D-System. Die durchgezogene Linie stellt Ersteres dar, die gestrichelten Letzteres. Die untere Grenze gibt die reine Expansionszeit an, die obere inklusive des *Overheads*.

Im Gegensatz dazu gilt dies beim 3D-System für die gesamte Simulation. Zudem ist der durch den *Overhead* bedingte *Offset* zwischen den Grenzen bei den lokalen Operatoren für $\mathsf{k} = 2$ in dem 3D-System geringfügig höher. Beide Beobachtungen können auf die zeitliche Schwankung der Rechenressourcen zurückgeführt werden, weshalb sie nicht übermäßig betont werden. Positiv hervorzuheben sind dagegen die folgenden zwei Erkenntnisse. Auf der einen Seite ist die mittlere Rechenzeit zur Auswertung eines Zeitschritts bereits für $q = 10$ deutlich gesunken im Vergleich zum Referenzfall mit $q = 1 \rightarrow \Delta t = \Delta t_{CFL}$. Auf der anderen Seite ist der Einfluss des *Overheads* kaum wahrzunehmen. Begründet wird dies mit der Tatsache, dass die Extraktion

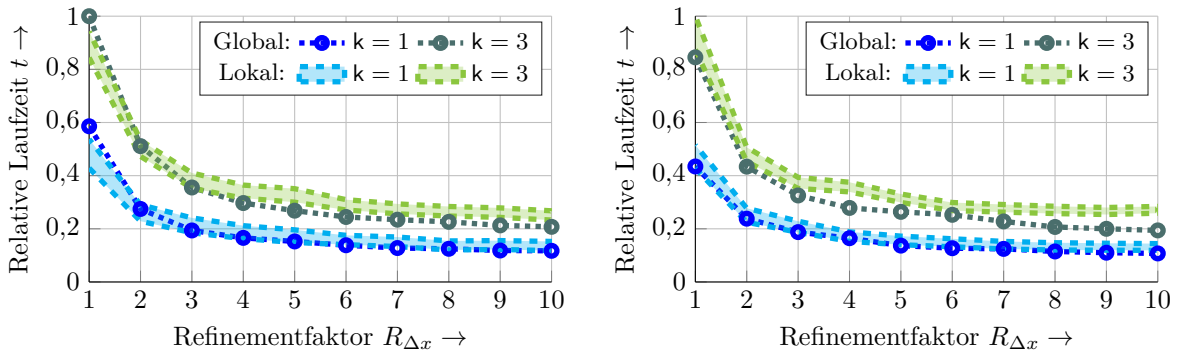
und Assemblierung jeden q-ten Zeitschritt ausgeführt werden muss. Somit muss bei $q = 1$ im Rahmen der Approximation zu jedem Zeitschritt extrahiert und assembliert werden. Wird wie in diesem Fall auf $q = 10$ erhöht, trägt der *Overhead* der Gleichung (6.11) nur jeden zehnten entwickelten Zeitschritt zur Rechenzeit bei. Die Quintessenz ist, dass der Einfluss des *Overheads* mit zunehmendem Wert von q abnimmt. Die Konvertierung in C hat keine nennenswerten Veränderungen ergeben, wie die Tabelle 6.4 beweist. Die relative Laufzeitabweichung zwischen MATLAB und C bleibt stets unter 1 %, wobei MATLAB geringfügig schneller ist.

Tabelle 6.4: Relative Laufzeitabweichungen zwischen MATLAB und C bei $\mathsf{k} = 1$ und $\mathsf{k} = 2$.

	Global 2D		Lokal 2D		Global 3D		Lokal 3D	
$R_{\Delta x}$ \ k	1	2	1	2	1	2	1	2
1	0,0022	0,0021	0,0018	0,0018	0,0019	0,0011	0,0014	0,0010
10	0,0019	0,0011	0,0023	0,0022	0,0008	0,0010	0,0010	0,0012

6.4.2 Anpassung des Schwellenwerts

In der zweiten praktischen Laufzeituntersuchung werden $q = 1$ und $c_{th} \in \{10^{-6}, 10^{-15}\}$ angenommen. Der Effekt des angepassten Schwellenwerts kann in der Abbildung 6.13 eingesehen werden. In der Abbildung 6.13a werden die Ergebnisse im 2D-System und in der Abbildung 6.13b jene im 3D-System demonstriert. Wie in der vorherigen Teilstudie bereits festgestellt, weichen auch hier die Laufzeiten zwischen dem 2D- und 3D-System kaum voneinander ab. Diese Tatsache belegt, dass die Performance der Operatoren nicht von der Dimension des Rechengebiets abhängig ist. Ferner sind die offensichtlichsten Unterschiede der Kurven das Resultat der zeitabhängigen Rechenleistung. Dennoch ist hervorzuheben, dass die Offsets zwischen den Laufzeitgrenzen bei den lokalen Operatoren ungefähr gleich groß bei $\mathsf{k} = 1$ und $\mathsf{k} = 3$ sind.



(a) Laufzeiten mit $q = 1 \wedge c_{th} \in \{10^{-6}, 10^{-15}\}$ bei 2D. **(b)** Laufzeiten mit $q = 1 \wedge c_{th} \in \{10^{-6}, 10^{-15}\}$ bei 3D.

Abbildung 6.13: Vergleich der relativen Laufzeiten der Operatoren für $\mathsf{k} = 1$ und $\mathsf{k} = 3$ in einem linearen 2D- und 3D-System.

Dies bestätigt die bereits getroffene Annahme, dass sich der *Overhead* mit höherem q verringert. Der Hauptunterschied zwischen den Teilstudien ist die höhere Laufzeit unter Verwendung der Konstellation $\mathsf{k} = 3$ im Vergleich zu $\mathsf{k} = 1$. Da beide Simulationen abgesehen vom Schwellenwert

c_{th} mit exakt denselben Parametern durchgeführt wurden, war dies aufgrund der höheren Entwicklungsordnung vorhersehbar. Die mit C generierten Ergebnisse, die mit denen von MATLAB relativiert und in der Tabelle 6.5 zusammengefasst wurden, haben auch in dieser Teilstudie zu keiner evidenten Laufzeitanpassung beigetragen. Weiterhin ist MATLAB minimal schneller in der Berechnung als C.

Tabelle 6.5: Relative Laufzeitabweichungen zwischen MATLAB und C bei $\mathsf{k} = 1$ und $\mathsf{k} = 3$.

	Global 2D		**Lokal 2D**		**Global 3D**		**Lokal 3D**	
k / $R_{\Delta x}$	**1**	**3**	**1**	**3**	**1**	**3**	**1**	**3**
1	0,0018	0,0019	0,0018	0,0017	0,0016	0,0016	0,0014	0,0013
10	0,0019	0,0017	0,0021	0,0012	0,0014	0,0013	0,0012	0,0014

6.4.3 Einsatz von Multiprocessing

Eine andere Möglichkeit, den *Overhead* fernab der Implementierung zu reduzieren, besteht im Einsatz von PU zur Parallelisierung des Prozessablaufs. Bisher wurde im Kontext der lokalen Operatoren von einer seriellen Verarbeitung der Daten ausgegangen. Dies bedeutet, dass bevor ein lokaler Operator für die Evaluation verwendet werden kann, dieser mit Ausnahme des ersten auf die Beendigung der vorherigen Entwicklung warten muss. Die Abbildung 6.14 verdeutlicht die Situation. Im klassischen Sinne besteht zwar keine Abhängigkeit zwischen den separaten lokalen Operatoren im Laufe der Expansion, dennoch ergibt sich aufgrund der seriellen maschinellen Verarbeitung der Daten eine zeitliche Abhängigkeit. Dies widerspricht dem Konzept die lokalen Operatoren unabhängig voneinander zu entwickeln.

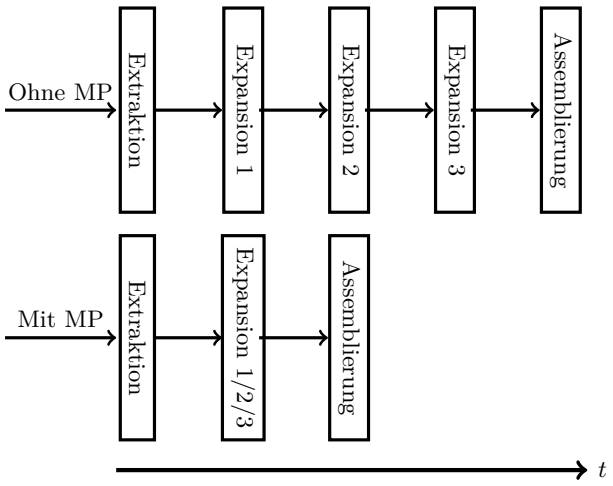
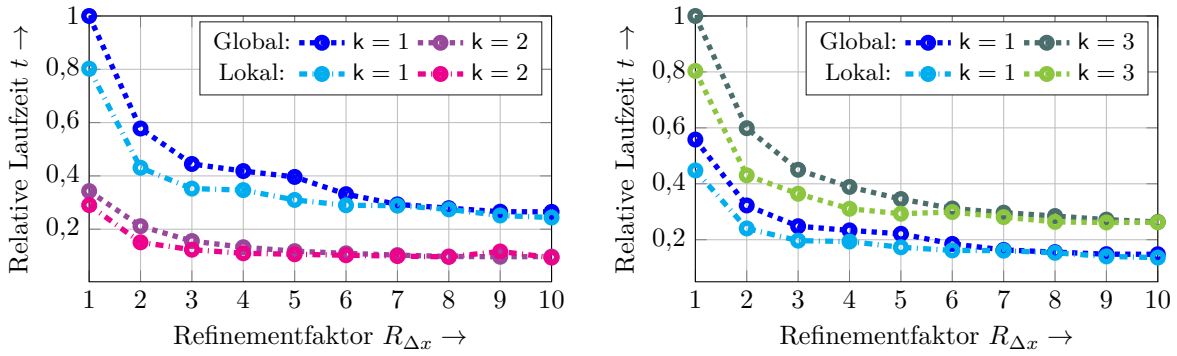


Abbildung 6.14: Schaubild zur Parallelisierung der lokalen Operatoren durch MP.

Zu diesem Zweck soll die Berechnung mit den lokalen Operatoren nicht nur in der Theorie, sondern auch in der Praxis unter dem Aspekt des MPs auf der CPU separat erfolgen. Die vorliegende Hardware erlaubt es während der Berechnungen bis zu 20 Kerne parallel zu beanspruchen. In MATLAB wird dies über die lizenzpflichtige *Parallel Computing Toolbox* [196] bewerkstelligt,

indem die Rechenlast auf sogenannte *Worker* [197] aufgeteilt wird. Dahingegen bietet die Programmiersprache C keine native Unterstützung für MP, weshalb externe Bibliotheken wie Portable Operating System Interface (POSIX) [198] eingebunden werden müssen [199]. Unter Berücksichtigung der bislang erzielten Resultate mit C wird vereinbart, die Laufzeiten ausschließlich mit MATLAB zu analysieren. Dasselbe gilt für die Dimension des Systems, sodass nur noch das 2D-System berücksichtigt wird. Es ist wichtig zu wissen, dass im Standardfall die MVMs bei voll- und dünnbesetzten Matrizen in MATLAB durch die BLAS-Bibliothek parallelisiert werden [200]. Aufgrund dessen wird beim globalen Operator nicht zwischen seriell und parallel generierten Ergebnissen differenziert. Des Weiteren wird nachfolgend für einen besseren Überblick nicht mehr zwischen der reinen Expansion- und der gesamten Rechenzeit separiert.



(a) Laufzeiten mit $q \in \{1, 10\} \wedge c_{th} = 10^{-6}$ bei MP. **(b)** Laufzeiten mit $q = 1 \wedge c_{th} \in \{10^{-6}, 10^{-15}\}$ bei MP.

Abbildung 6.15: Vergleich der relativen Laufzeiten der Operatoren unter Einsatz von MP auf der CPU in einem linearen 2D-System.

Die Laufzeitresultate unter Einsatz von MP auf der CPU können in der Abbildung 6.15 betrachtet werden. Der positive Effekt des MPs ist eindeutig erkennbar, sowohl in der Abbildung 6.15a als auch in der Abbildung 6.15b. Im Falle einer homogenen Diskretisierung mit $R_{\Delta x} = 1$ kann durch den parallelisierten Ablauf bei $\mathsf{k} = 1$ und $\mathsf{k} = 3$ eine Beschleunigung von bis 20 % erzielt werden. Der Effekt schwächt allerdings mit steigendem $R_{\Delta x}$ ab, da die Anzahl der Diskretisierungspunkte in den groben Subgebieten deutlich verringert wird. Bezüglich $\mathsf{k} = 2$ fällt der Rechenzeitgewinn geringer aus. Die Ursache dafür ist der bereits reduzierte *Overhead* aufgrund des höher definierten Zeitfaktors $q = 10$. Dennoch liegt die Zeiteinsparung im einstelligen Prozentbereich. Insgesamt kann für lineare Systeme festgehalten werden, dass die Auswertung mit den parallelisierten lokalen Operatoren auf der CPU immer schneller bzw. im schlimmsten Fall gleich schnell wie die Auswertung mit dem globalen Operator ist.

6.4.4 Einsatz von Multithreading

Neben MP besteht auch die Möglichkeit die Approximation über MT zu parallelisieren. Im Gegensatz zum MP liegt das Hauptaugenmerk dabei die Kerne der GPU, anstatt der CPU zu beanspruchen. Verfügt das System über eine kompatible Nvidia-GPU mit Compute Unified Device Architecture (CUDA), ermöglicht MATLAB durch die *Parallel Computing Toolbox* eine Beschleunigung, ohne dass eine explizite Programmierung in CUDA erforderlich ist [201].

Waren es zuvor die Extraktion und Assemblierung, so ist es infolge von MT insbesondere die Expansion, die parallelisiert wird. Da die Expansion ausschließlich auf MVMs beruht, kann die Anzahl der gleichzeitig durchgeführten Berechnungen um ein Vielfaches gesteigert werden. Dadurch kann nicht nur die Auswertung mit den untersuchten lokalen Operatoren signifikant beschleunigt werden, sondern auch die mit dem konventionellen globalen Operator. Im Falle der lokalen Operatoren ist ebenfalls eine Verknüpfung beider PU sinnvoll, indem die Rechenzeit des *Overheads* durch die CPU und die der Expansion über die GPU reduziert wird. Die Abbildung 6.16 illustriert das Konzept der Parallelisierung der MVMs durch MT.

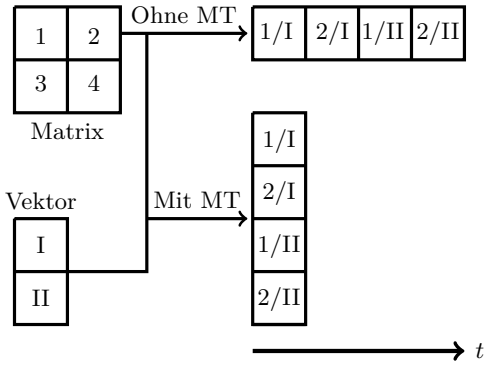


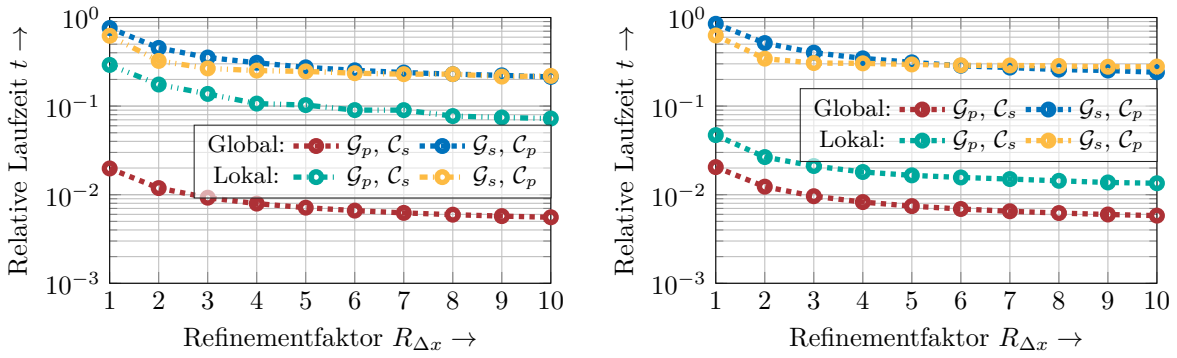
Abbildung 6.16: Schaubild zur Parallelisierung der lokalen Operatoren durch MT.

Basierend auf den zuvor behandelten Ergebnissen aus [PS4], wurde in der Publikation [PS5] der Zeitfaktor q deutlich erhöht. Einerseits wurde der Einfluss des *Overheads* auf ein Minimum begrenzt. Andererseits wird der Hauptvorteil der Faber-Methode, wesentlich höhere Zeitschrittweiten als Δt_{CFL} zu ermöglichen, zur Geltung gebracht. Als Rechengebiet Ω kommt der Filmwellenleiter mit drei *Layern* aus der Abbildung 6.6 zum Einsatz. Damit die angewandten Entwicklungsordnungen m_{f} und m_{g} beim hohem Zeitfaktor $q = 100$ mit dem Fehlerkriterium $c_{th} = 10^{-6}$ und dem maximalen Refinementfaktor $R_{\Delta x} = 10$ konform sind, muss die Anzahl der Diskretisierungspunkte erhöht werden. Daher werden $N_x = N_y = 6000$ Abtastpunkte zur Diskretisierung von Ω verwendet. Die Hardwarespezifikationen sind in der Tabelle 6.6 zusammengefasst.

Tabelle 6.6: Hardwarespezifikationen für die linearen Berechnungen auf der CPU und GPU.

CPU	Taktrate	Cache	Kerne	GPU	VRAM	Kerne	RAM
AMD EPYC 7252	3,1 GHz	L3 mit 64 MB	8	Nvidia RTX A6000	48 GB GDDR6	10752	256 GB

In der Abbildung 6.17 werden die ermittelten Laufzeiten der Operatoren veranschaulicht. Der Prozessablauf kann zwischen seriell `s` und parallelisiert `p` gewechselt werden. Findet primär die Parallelisierung durch MT auf der CPU statt, wird dies mit $\mathcal{C}$ gekennzeichnet. Wird dagegen MT auf der GPU betrieben, so wird dies mit $\mathcal{G}$ bezeichnet. In der Abbildung 6.17a werden die Ergebnisse für $q = 1$ gezeigt. Der Einfluss der Parallelisierung des globalen Operators auf der GPU zeigt sich deutlich. Bei $R_{\Delta x} = 1$ lässt sich die Laufzeit um nahezu zwei Größenordnungen



(a) Laufzeiten mit $q = 1$ bei MP und MT.

(b) Laufzeiten mit $q = 100$ bei MP und MT.

Abbildung 6.17: Vergleich der relativen Laufzeiten der Operatoren unter Einsatz von MP auf der CPU und MT auf der GPU in einem linearen 2D-System.

reduzieren. Währenddessen ist der Rechenzeitgewinn bei der Auswertung mit den lokalen Operatoren relativ gering und steigt bis $R_{\Delta x} = 10$ nur um etwa eine halbe Größenordnung. Die Extraktion und Assemblierung bei der Approximation mit den lokalen Operatoren führen somit zu einer beachtlichen Verzögerung. Durch diesen Umstand profitiert besonders die Evaluation mit dem globalen Operator, da im gesamten Ablauf lediglich MVMs vonnöten sind. Eine Erhöhung des Zeitfaktors auf $q = 100$ kann dem bedeutend entgegenwirken, wie in der Abbildung 6.17b illustriert. Der *Overhead* wird erheblich reduziert, wodurch eine Beschleunigung von ungefähr anderthalb Größenordnungen realisiert werden kann. Dennoch verzögert der *Overhead* auch in diesem Fall die Entwicklung, sodass ein *Offset* von einer halben Größenordnung zwischen den Operatoren bestehen bleibt, wenn diese mit MT auf der GPU parallelisiert werden. Die Ergänzung von MT mit MP führt zu keiner signifikanten Laufzeitveränderung, wie in Tabelle 6.7 ersichtlich. Mit Hilfe von MP liegt der Rechenzeitgewinn weit unter 1 %.

Tabelle 6.7: Relative Laufzeitabweichungen unter Einsatz von MT ohne und mit MP.

	Global		**Lokal**	
$R_{\Delta x}$ \ q	**1**	**100**	**1**	**100**
1	0,0013	0,0011	0,0011	0,0009
10	0,0010	0,0011	0,0014	0,0015

6.5 Numerische Evaluierung in einem nichtlinearen System

Basierend auf der numerischen Evaluierung der lokalen Operatoren in einem linearen System aus dem Abschnitt 6.4 wird nachfolgend der Propagator in einem nichtlinearen System evaluiert. Zu diesem Zweck wird für die Anwendung der Faber-Methode in einem nichtlinearen System die Mastergleichung aus der Gleichung (4.1) in eine semilineare Gleichung überführt [124]:

$$\frac{\partial \vec{\Psi}(t)}{\partial t} = f(\Delta t \mathcal{H}) \cdot \vec{\Psi}(t) \rightarrow \frac{\partial \vec{\Psi}(t)}{\partial t} = \underbrace{f(\Delta t \mathcal{H}) \cdot \vec{\Psi}(t)}_{linear} + \underbrace{\mathcal{N}(\vec{\Psi}(t))}_{nichtlinear}. \tag{6.12}$$

Da im Folgenden die Modellierung eines Lasersystems angestrebt wird, wird dazu das Zwei-Niveau-Modell aus dem Abschnitt 2.2.2 verwendet. Aus diesem Grund werden neben den Feldkomponenten $\vec{E}(t)$ und $\vec{H}(t)$ zusätzlich die Besetzungsdichten $\vec{N}_1$ im *Niveau 1* sowie $\vec{N}_2$ im *Niveau 2* benötigt. Weil diese von der Polarisation $\vec{P}$ und dem Polarisationsstrom $\vec{J}_D$ abhängen, führt dies zur erweiterten Systemmatrix

$$\mathcal{H} = \begin{bmatrix} 0 & \frac{1}{\epsilon}\vec{\nabla}\times & 0 & 0 & 0 & -\frac{1}{\epsilon} \\ -\frac{1}{\mu} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -\gamma_{12} & \gamma_{21} & 0 & 0 \\ 0 & 0 & \gamma_{12} & -\gamma_{21} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & -(\frac{4{\omega_m}^2+{\Gamma_m}^2}{4}) & -\Gamma_m \end{bmatrix} \tag{6.13}$$

eines 1D-Lasersystems. Für den korrespondierenden Zustandsvektor ergibt sich

$$\vec{\Psi} = [\vec{E}, \vec{H}, \vec{N_1}, \vec{N_2}, \vec{P}, \vec{J_D}]^T. \tag{6.14}$$

Der nichtlineare Operator $\mathcal{N}$ in der Gleichung (6.12) wird wie folgt angegeben:

$$\mathcal{N}(\vec{\Psi}) = \begin{bmatrix} 0 \\ 0 \\ \frac{1}{\hbar\omega_m}\vec{E}(\vec{J_D} + \frac{\Gamma_m}{2}\vec{P}) \\ -\frac{1}{\hbar\omega_m}\vec{E}(\vec{J_D} + \frac{\Gamma_m}{2}\vec{P}) \\ 0 \\ \sigma_{\mathrm{m}}\Delta\vec{N}(t)\vec{E}(t) \end{bmatrix} \tag{6.15}$$

In [15] und [16] wurden unterschiedliche Ansätze mit der Faber-Methode kombiniert und hinsichtlich ihrer Effizienz bei der Approximation der Gleichung (6.12) untersucht. Zur besseren Verständlichkeit wird die Betrachtung auf die Exponential-Euler-Methode [143] beschränkt:

$$\frac{\partial\vec{\Psi}(t)}{\partial t} = \underbrace{f(\Delta t\mathcal{H}) \cdot \vec{\Psi}(t)}_{linear} + \underbrace{\Delta t \cdot \varphi(\Delta t\mathcal{H}) \cdot \mathcal{N}(\vec{\Psi}(t))}_{nichtlinear}. \tag{6.16}$$

Die φ-Funktion in der Gleichung (6.15) stellt eine Hilfsfunktion dar und ist definiert als

$$\varphi(\Delta t\mathcal{H}) = \frac{f(\Delta t\mathcal{H}) - \mathbb{1}}{\Delta t\mathcal{H}}. \tag{6.17}$$

Augenscheinlich wird der lineare Operator $f(\Delta t\mathcal{H})$ nicht nur im linearen Teil der Gleichung (6.16) genutzt, sondern ebenso in der φ-Funktion im nichtlinearen Teil. Diese Erkenntnis ist insofern bedeutsam, als das Ergebnis der Approximation von $f(\Delta t\mathcal{H})$ aus der linearen Berechnung unmittelbar im nichtlinearen Teil berücksichtigt werden kann. Diese Tatsache ermöglicht eine Effizienzsteigerung sowohl bei der Berechnung des nichtlinearen Teils als auch der gesamten Exponential-Euler-Methode.

Dennoch ist aufgrund der Matrixordnung von $\mathcal{H}$ bereits bei einem 1D-Lasersystem von einer hohen Komplexität auszugehen. Weil im Anschluss sogar ein 2D-Lasersystem mit einer PML als Grenzschicht betrachtet wird, steigt die Komplexität weiter an. Infolgedessen werden zwei Methoden zur Reduktion der Komplexität integriert. Analog zum vorherigen Abschnitt ist die

erste Methode eine nichtuniforme Gitterstruktur einzusetzen, sodass weniger Elemente in $\mathcal{H}$ abgespeichert werden müssen. Im Hinblick auf das Lasermodell liegt es nahe, das Lasermedium mit einer feinen Diskretisierung abzutasten und das verbleibende Gebiet mit einer groben. Diesbezüglich müssen der lineare und nichtlineare Teil der Gleichung (6.16) nicht mehr mit der vollständigen Systemmatrix $\mathcal{H}$ berechnet werden, stattdessen können die relevanten Submatrizen gemäß der Subgebiete von $\mathcal{H}$ extrahiert und separat in die Teilberechnungen eingebunden werden. In Anlehnung an das *Refinement* in longitudinaler Richtung durch $R_{\Delta x} = \frac{\Delta x_\mathsf{g}}{\Delta x_\mathsf{f}}$ wird der Refinementfaktor $R_{\Delta y} = \frac{\Delta y_\mathsf{g}}{\Delta y_\mathsf{f}}$ für die Anpassung in transversaler Richtung eingeführt. Die Parameter Δy_g und Δy_f stehen jeweils für die Diskretisierungsweite im groben und feinen Subgebiet in y-Richtung. Bei der zweiten Methode wird der Operator gesplittet, sodass dieser separat ausgewertet werden kann. Wird die Zusammensetzung von $\mathcal{H}$ genauer betrachtet, fällt auf, dass bei der Auswertung des linearen Teils die Ratengleichungen vom übrigen System entkoppelt sind. Anstatt die gesamte Systemmatrix in dem Matrixexponential zu evaluieren, kann $\mathcal{H}$ in zwei kommutierende Matrizen der Form $\mathcal{H} = \mathcal{H}_1 + \mathcal{H}_2$ unterteilt werden. Demzufolge werden weder das Lie-Trotter- noch das Strang-Verfahren aus dem Abschnitt 6.1.3 benötigt. Die Submatrizen

$$\mathcal{H}_1 = \begin{bmatrix} 0 & \frac{1}{\epsilon}\vec{\nabla}\times & 0 & -\frac{1}{\epsilon} \\ -\frac{1}{\mu}\vec{\nabla}\times & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -(\frac{4{\omega_m}^2+{\Gamma_m}^2}{4}) & -\Gamma_m \end{bmatrix}, \quad (6.18) \qquad \mathcal{H}_2 = \begin{bmatrix} -\gamma_{12} & \gamma_{21} \\ \gamma_{12} & -\gamma_{21} \end{bmatrix} \quad (6.19)$$

werden von $\mathcal{H}$ aus der Gleichung (6.13) extrahiert. Für das Matrixexponential folgt demnach $\exp(\Delta t\mathcal{H}) = \exp(\Delta t\mathcal{H}_1) \cdot \exp(\Delta t\mathcal{H}_2)$, wodurch zwei Approximationen mit geringerer Komplexität durchgeführt werden können. Ein Fehler aufgrund des *Splittings* ist wegen der Kommutativität der beteiligten Matrizen nicht zu erwarten. Zudem ist es möglich den zeitlichen Verlauf der Ratengleichungen aus der Gleichung (6.19) mit Hilfe einer PDGL zu bestimmen:

$$\frac{\partial}{\partial t}\begin{bmatrix}\vec{N_1} \\ \vec{N_2}\end{bmatrix} = \begin{bmatrix} -\gamma_{12} & \gamma_{21} \\ \gamma_{12} & -\gamma_{21} \end{bmatrix}\begin{bmatrix}\vec{N_1} \\ \vec{N_2}\end{bmatrix}. \quad (6.20)$$

Der Vorteil durch die Überführung in die PDGL besteht darin, dass die Gleichung (6.20) analytisch gelöst werden kann. Hierbei ergeben sich die analytischen Lösungen

$$\vec{N_1}(t) = \mathsf{x}_1\frac{\gamma_{12}\exp(-t(\gamma_{12}+\gamma_{21}))+\gamma_{21}}{\gamma_{12}+\gamma_{21}} - \mathsf{x}_2\frac{\gamma_{21}\exp(-t(\gamma_{12}+\gamma_{21}))-1}{\gamma_{12}+\gamma_{21}}, \quad (6.21)$$

$$\vec{N_2}(t) = \mathsf{x}_2\frac{\gamma_{21}\exp(-t(\gamma_{12}+\gamma_{12}))+\gamma_{12}}{\gamma_{12}+\gamma_{21}} - \mathsf{x}_1\frac{\gamma_{12}\exp(-t(\gamma_{12}+\gamma_{21}))-1}{\gamma_{12}+\gamma_{21}} \quad (6.22)$$

mit den Konstanten x_1 und x_2. Durch das *Operatorsplitting* kann der Propagator mit der extrahieren Submatrix $\mathcal{H}_1$ approximiert werden, wodurch sich ein Gewinn an Rechenzeit erzielen lässt.

Im Rahmen der Evaluation in einem nichtlinearem System wird der Propagator in dem 2D-System aus der Abbildung 6.18 entwickelt. Das Rechengebiet Ω repräsentiert einen Distributed Feedback (DFB)-Laser, der in einem Filmwellenleiter mit drei *Layern* integriert ist. Als Wellenlänge wird $\lambda = 1{,}55\,\mu\text{m}$ angenommen, was charakteristisch für Anwendungen in der Optik

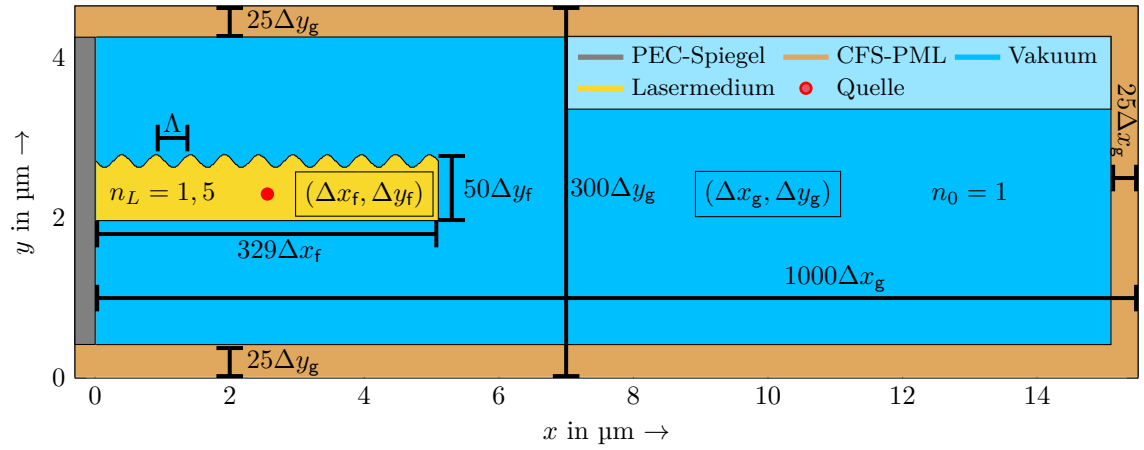


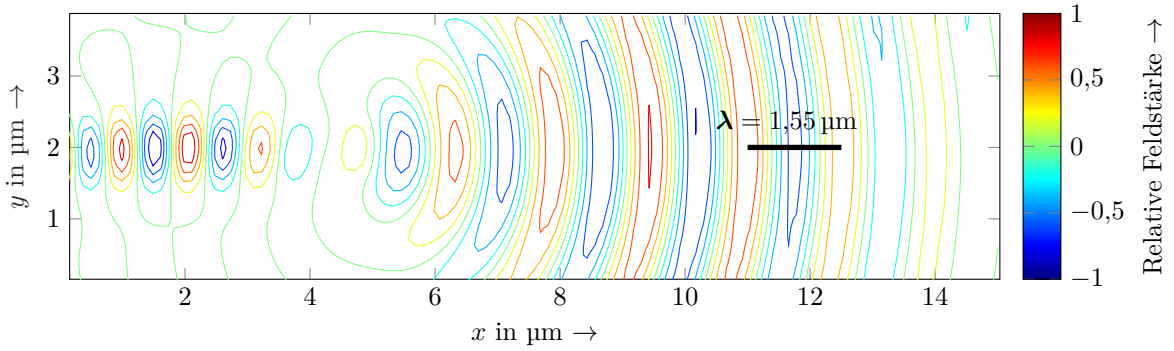
Abbildung 6.18: 2D-DFB-Lasermodell in einem Filmwellenleiter mit drei *Layern*.

ist [202]. Die Diskretisierungsweiten sind auf $\Delta x_f = \Delta x_g = \Delta y_f = \Delta y_g = 15{,}5\,\text{nm}$ festgelegt, sodass die numerische Dispersion mit $\frac{\lambda}{\Delta} = 100$ vernachlässigt werden kann. Die TE$_z$-Mode mit $\vec{\Psi} = [E_z, H_x, H_y]^T$ wird im Zentrum des Lasermediums mit dem Brechungsindex $n_L = 1{,}5$ angeregt und von Vakuum mit dem Brechungsindex $n_0 = 1$ eingegrenzt. Die Ausbreitung des Gaußschen Pulses findet in der x-y-Ebene statt. Des Weiteren wird an der oberen Grenzfläche des Lasermediums zum Vakuum ein Bragg-Gitter [203] mit einer Gitterperiode von $\Lambda = 10$ eingebunden, um die gewünschte Wellenlänge der elektromagnetischen Welle selektiv zu reflektieren. Die linke Grenzfläche von Ω stellt einen PEC-Spiegel dar und ist die einzige, die reflektierend wirkt, während an den verbleibenden Grenzflächen die CFS-PML mit $\alpha_x = 0$, $\kappa_x = 1$ und $\sigma_x = 85626$ zur Absorption der Wellen implementiert sind. Das System ist so konzipiert, dass 1000 Abtastpunkte in x-Richtung und 300 in y-Richtung verwendet werden. Die Breite des Lasermediums beträgt $329\Delta x_f$ und die Höhe $50\Delta y_f$. Für die CFS-PML beträgt die Breite $25\Delta x_g$ und die Höhe $25\Delta y_g$. Hinsichtlich der Ratengleichungen werden als Startbedingungen die Übergangsraten $\gamma_{12} = 1{,}4989 \cdot 10^{-12}\,\text{s}^{-1}$ und $\gamma_{21} = 1{,}5289 \cdot 10^{-12}\,\text{s}^{-1}$ vorgegeben [15], wobei Erstere die Pumpstrahlung modelliert. Ferner gilt $\varkappa_1 = \varkappa_2 = 0{,}5 \cdot 10^{29}\,\text{m}^{-3}$ für die Konstanten aus der Gleichung (6.21) und Gleichung (6.22) sowie für die Kopplung zwischen der effektiven Ladungsträgerdichte und dem externen elektrischen Feld $\sigma_m = 5{,}716 \cdot 10^{-7}\,\text{C}^2/\text{kg}$. Das Lasergebiet wird mit der Besetzungsdichten $\vec{N}_1 = \vec{N}_2 = 0{,}5 \cdot 10^{29}\,\text{m}^{-3}$ initialisiert. Für die zugrunde liegende Wellenlänge λ ergeben sich für die Übergangsfrequenz $\omega_m = 2\pi c/\lambda \approx 1{,}257 \cdot 10^{15}\,\text{Hz}$ und für die FWHM $\Gamma_m = \omega_m/5 \approx 0{,}251 \cdot 10^{15}\,\text{Hz}$. Aus Gründen der Stabilität wird die Schwelle für das Fehlerkriterium der Faber-Methode auf $c_{th} = 10^{-15}$ abgesenkt und der Zeitfaktor auf $q = 5$ gesetzt.

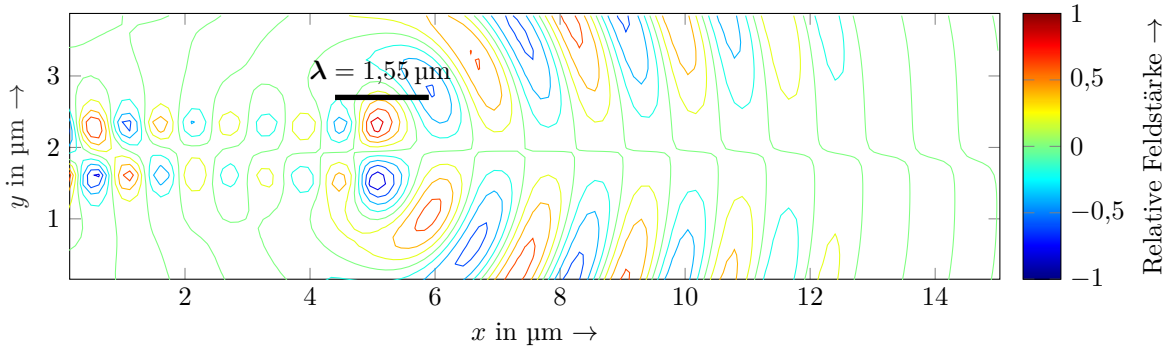
Zur Validierung des Lasermodells wurden die Feldkomponenten bei $t = 6000\Delta t_{CFL}$ aufgezeichnet und entsprechend ihrer maximalen Feldstärke relativiert. Die Feldverteilung kann der Abbildung 6.19 entnommen werden. Dabei illustriert die Abbildung 6.19a die Verteilung der E_z-Komponente, die Abbildung 6.19b die Verteilung der H_x-Komponente und die Abbildung 6.19c die Verteilung der H_y-Komponente. Bei allen Feldkomponenten ist ersichtlich, dass der Laser mit der geforderten Wellenlänge $\lambda = 1{,}55\,\text{µm}$ angeregt wird.

Neben den Möglichkeiten, die Komplexität und Laufzeit durch die Implementierung zu reduzieren, wird wie in dem Abschnitt 6.4 der Einsatz von MP auf der CPU und von MT auf der GPU

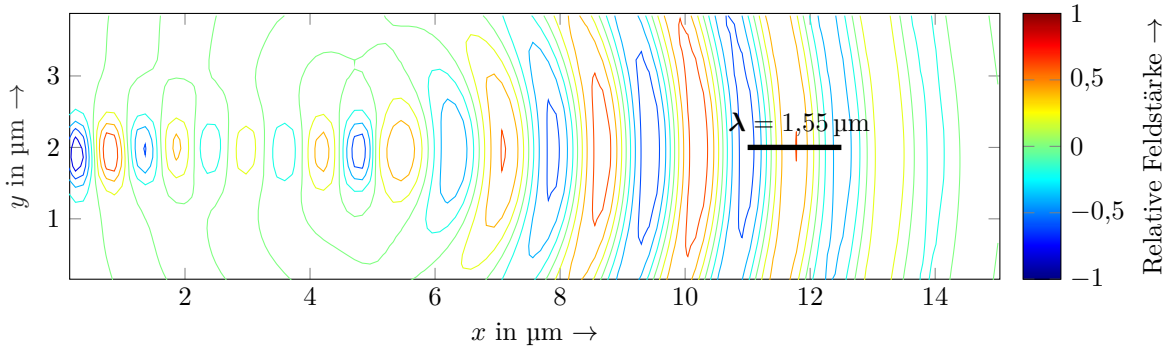
analysiert. Die Hardwarespezifikationen sind der Tabelle 6.8 aufgelistet. Die Berechnungen erfolgen analog zum Abschnitt 6.4 in MATLAB R2024b. Zum einen ist die Parallelisierung mit der *Parallel Computing Toolbox* intuitiv. Zum anderen ist die Approximation mit der niederen Programmiersprache C im Kontext der Faberpolynome nahezu genauso schnell, wie zuvor bewiesen.



(a) Feldverteilung der E_z-Komponente.



(b) Feldverteilung der H_x-Komponente.



(c) Feldverteilung der H_y-Komponente.

Abbildung 6.19: Feldverteilung in dem 2D-DFB-Lasermodell bei $t = 6000\Delta t_{CFL}$ mit $\lambda = 1{,}55\,\mu\text{m}$.

Dennoch ist zu beachten, dass im Vergleich zur Evaluation im linearen System die Systemmatrix $\mathcal{H}$, außer den Feldkomponenten der elektromagnetischen Welle, auch die Besetzungsdichten, die Polarisation, den Polarisationsstrom und die PML enthält. Dies ist insbesondere für die Auswertung auf der GPU relevant, da mit der vorhandenen 16 GB VRAM-Kapazität maximal

$2 \cdot 10^9$ Elemente verarbeitet werden können. Dies gilt unter der Prämisse, dass ein Element mit einer Wertigkeit ungleich Null 8 Byte allokiert, $\mathcal{H}$ als dünnbesetzte Matrix gespeichert ist und andere Hintergrundprozesse nicht berücksichtigt werden.

Nachfolgend werden die Laufzeitresultate in Form von Balkendiagrammen anstelle von Kurvendiagrammen dargestellt, um im Detail den Anteil der linearen Auswertung, der nichtlinearen Auswertung und des *Overheads* nachvollziehen zu können. Dabei bezieht sich der *Overhead* nun nicht mehr ausschließlich auf die Extraktion und Assemblierung der lokalen Operatoren, sondern auch auf Hintergrundprozesse, wie das Kopieren und Verschieben von Daten sowie das Allokieren von Speicherplatz. Somit ist sogar bei der Entwicklung mittels des globalen Operators ein *Overhead* gegeben.

Tabelle 6.8: Hardwarespezifikationen für die nichtlinearen Berechnungen auf der CPU und GPU.

CPU	Taktrate	Cache	Kerne	GPU	VRAM	Kerne	RAM
Intel i7-13700K	3,4 GHz	L3 mit 30 MB	16	Nvidia GeForce RTX 4060 Ti	16 GB GDDR6	4352	32 GB

Um die Ergebnisse vom System unabhängig zu machen und ihre Allgemeingültigkeit zu erhöhen, wird eine ergänzende Studie durchgeführt, in der die Abhängigkeit der Rechenzeit von der Matrixordnung untersucht wird. Dafür wird der Skalierungsfaktor $S \in \mathbb{N}$ eingeführt, sodass das System aus der Abbildung 6.18 mit mehr Abtastpunkten diskretisiert wird und die Matrixordnung entsprechend linear zu S zunimmt. Alle generierten Ergebnisse einer Teilstudie werden in Bezug auf ihr Maximum relativiert. Die im Folgenden analysierten Ergebnisse sind teilweise in [PS6] und [PS7] publiziert.

6.5.1 Unterteilung in transversaler Richtung

Wenn das Rechengebiet aus der Abbildung 6.18 in transversaler Richtung in grobe und feine Subgebiete unterteilt wird, entsteht die in der Abbildung 6.20 illustrierte Struktur. Es wird darauf hingewiesen, dass sich das feine Subgebiet nicht ausschließlich auf das Lasermedium beschränkt, sondern sich entsprechend der gemeinsamen Höhe des Laser- und Zwischengebiets über das Rechengebiet in x-Richtung bis zum Rand erstreckt. Zwar wäre die Implementierung umsetzbar, bei der nur das Lasermedium als feines Subgebiet behandelt wird, jedoch wird zunächst konzeptionell untersucht in welcher Richtung eine Unterteilung den höheren Ertrag bringt. Im Zuge der Unterteilung werden aus dem globalen Operator drei lokale Operatoren extrahiert. Dabei ergeben sich die Subzustandsvektoren $\vec{\Psi}_{\mathsf{g}}^{N_{\mathsf{g}1}}$ und $\vec{\Psi}_{\mathsf{g}}^{N_{\mathsf{g}2}}$ für die groben Subgebiete und $\vec{\Psi}_{\mathsf{f}}^{N_{\mathsf{f}}}$ für das feine Subgebiet. Zur Vereinfachung der Berechnung wurde das Lasermedium mittig in y-Richtung platziert, sodass $N_{\mathsf{g}1} = N_{\mathsf{g}2} = N_{\mathsf{g}}$ gilt. Damit Entwicklungsfehler durch die Expansion proaktiv gemildert werden, soll nur $\vec{\Psi}_{\mathsf{f}}^{N_{\mathsf{f}}}$ in die groben Subgebiete expandieren dürfen. Dafür wird der Ansatz mit dem Zwischenbereich für die Expansion aus der Abbildung 6.7 mit der Bedingung $m_{\mathsf{f}} \geq m_{\mathsf{g}}$ gewählt. Ferner bleibt das feine Subgebiet unabhängig von $R_{\Delta y}$, um die nichtlineare Berechnung nicht durch numerische Dispersion zu beeinflussen.

Die Laufzeitresultate der Auswertungen mit dem globalen Operator G und den lokalen Operatoren L sind in dieser Teilstudie in der Abbildung 6.21 dargestellt, wobei der Index s die Anwendung

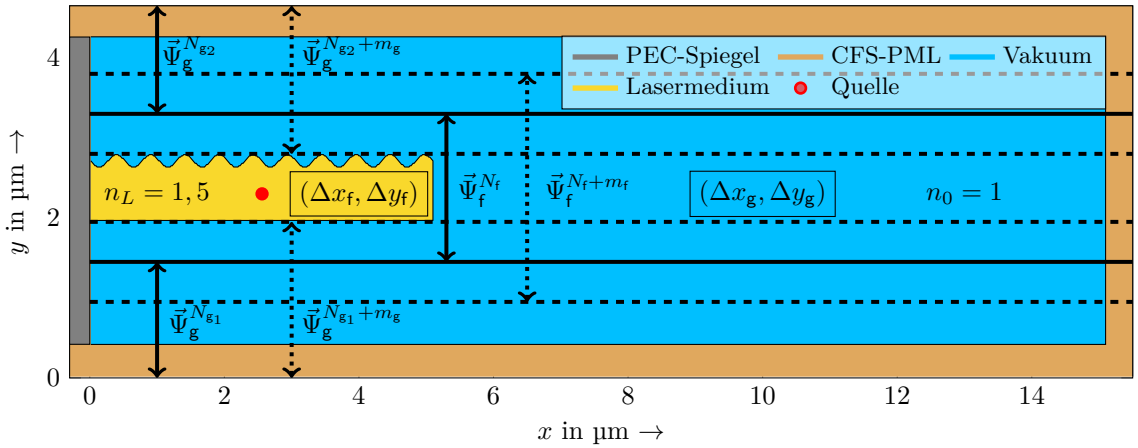


Abbildung 6.20: Lokale Operatoren mit Zwischenbereich für die Expansion zur Vermeidung von numerischer Dispersion in dem 2D-DFB-Lasermodell in transversaler Richtung.

des *Operatorsplittings* kennzeichnet. Für den Refinementfaktor wurde das Intervall $R_{\Delta y} \in [1, 3]$ festgelegt, da bei höherem $R_{\Delta y}$ numerische Dispersion zu beobachten ist. Beim homogen diskretisierten Rechengebiet mit $R_{\Delta y} = 1$ ist zunächst festzuhalten, dass der Wechsel vom globalen Ansatz zum lokalen einen Rechenzeitgewinn von etwa 10 % erzielt. Insbesondere konnte die Rechenzeit für die nichtlineare Berechnung um mehr als ein Drittel verringert werden. Die Begründung dafür liegt in der Verwendung der extrahierten Submatrix, die an die φ-Funktion des nichtlinearen Teils der Exponential-Euler-Approximation aus der Gleichung (6.16) übergeben wird. Im Gegenzug muss eine Laufzeiterhöhung für den linearen Teil toleriert werden, da besonders die Zwischenbereiche für die Expansion der lokalen Operatoren öfters in die Auswertung mit einfließen. Zusätzlich muss die Erhöhung des *Overheads* für die Anwendung der lokalen Operatoren toleriert werden, da die Extraktion und Assemblierung grundlegende Prozessschritte der Methode darstellen.

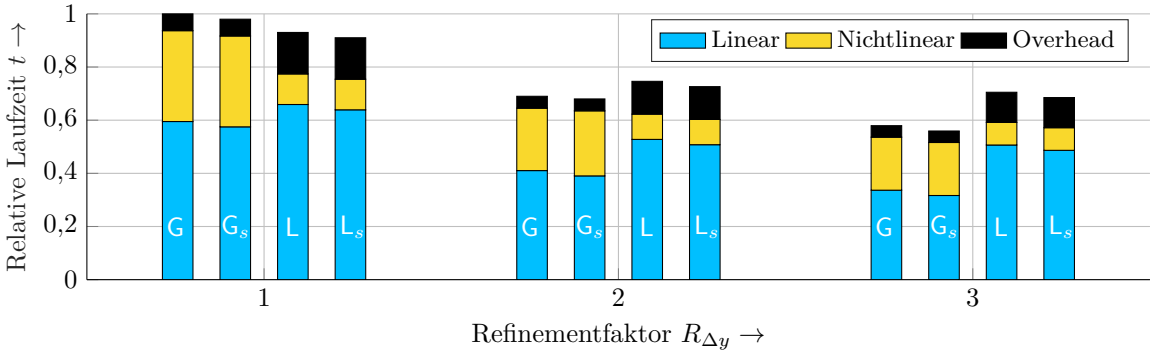


Abbildung 6.21: Laufzeitvergleich bei transversaler Unterteilung in dem 2D-DFB-Lasermodell.

Wird das *Refinement* auf $R_{\Delta y} > 1$ erhöht, ist eine Verschlechterung der Performance durch die lokalen Operatoren zu erkennen. Nach wie vor ist der Anteil der Nichtlinearität geringer, allerdings bleiben der Anteil der Linearität und der des *Overheads* signifikant höher. Folglich wird der Effekt, die Komplexität für die Auswertung der Nichtlinearität zu reduzieren, durch die

wiederholte Entwicklung in den Zwischenbereichen mittels der lokalen Operatoren zunichtegemacht. Der *Splittingansatz* ermöglicht eine Beschleunigung der Berechnung um 2 %, unabhängig vom Refinementfaktor und vom angewandten Operator. Speziell profitiert die lineare Auswertung von der Extraktion der Ratengleichungen. Da das Lasermedium in dem betrachteten System nur einen geringen Anteil des gesamten Rechengebiets ausmacht, müsste entweder das Gebiet des Lasermediums vergrößert oder das restliche Rechengebiet reduziert werden, damit das *Splitting* zu einem größeren Rechenzeitgewinn führt.

6.5.2 Unterteilung in longitudinaler Richtung

Wird das Rechengebiet in longitudinaler Richtung unterteilt, resultieren die Gegebenheiten aus der Abbildung 6.22. Wie zuvor erfolgt die Expansion der lokalen Operatoren mit Hilfe eines vordefinierten Zwischengebiets. Aufgrund der Unterteilung des Systems ist nur noch ein Zwischengebiet anstelle von zweien notwendig. Außerdem sind hier zwei Subzustandsvektoren für die Approximation ausreichend. Der Subzustandsvektor für das feine Subgebiet ist $\vec{\Psi}_{\mathrm{f}}^{N_{\mathrm{f}}}$ und für das grobe $\vec{\Psi}_{\mathrm{g}}^{N_{\mathrm{g}}}$. Ebenso soll das feine Subgebiet unabhängig von der Anpassung des Refinementfaktors $R_{\Delta x}$ sein. Als feines Subgebiet wird das Gebiet definiert, das der gemeinsamen Breite des Lasermediums und des Zwischenbereichs entspricht sowie sich im Rechengebiet entlang der y-Achse bis zum Rand erstreckt.

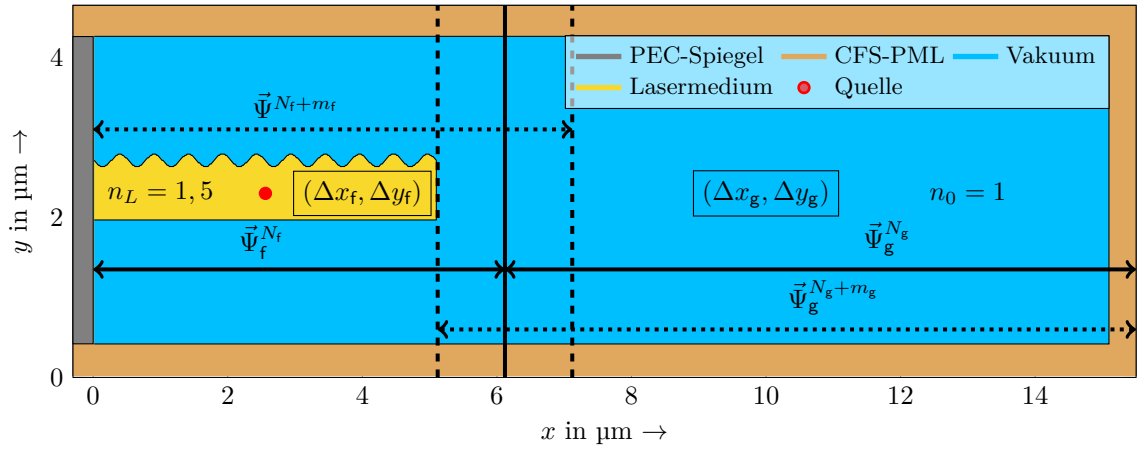


Abbildung 6.22: Lokale Operatoren mit Zwischenbereich für die Expansion zur Vermeidung von numerischer Dispersion in dem 2D-DFB-Lasermodell in longitudinaler Richtung.

Die Resultate für die Auswertung werden in der Abbildung 6.23 gezeigt. Verglichen mit der transversalen Unterteilung kann bei der longitudinalen Unterteilung allein durch die Auswertung mit den lokalen Operatoren der Rechenzeitgewinn fast verdoppelt werden, was zu einer Beschleunigung von nahezu 20 % führt. Der wesentliche Unterschied besteht darin, dass neben der nichtlinearen Evaluation auch die Rechenzeit für den linearen Teil durch den Einsatz der lokalen Operatoren reduziert werden konnte. Dies wird damit begründet, dass auf der einen Seite nur zwei lokale Operatoren verwendet werden und auf der anderen Seite die Fläche des Zwischengebiets kleiner als bei der transversalen Unterteilung ist. Trotzdem wirkt der *Overhead* dem leicht entgegen. In diesem Zusammenhang hat die Studie im Abschnitt 6.4 bewiesen, dass

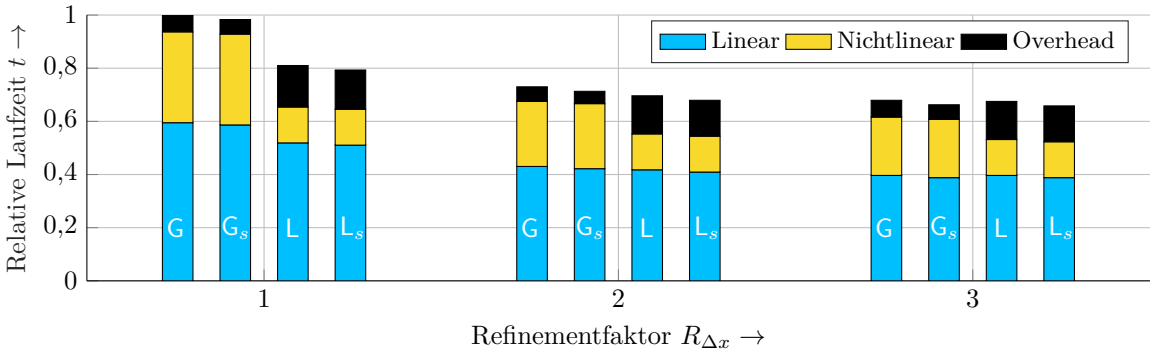


Abbildung 6.23: Laufzeitvergleich bei longitudinaler Unterteilung in dem 2D-DFB-Lasermodell.

bei der Implementierung der lokalen Operatoren lediglich die Erhöhung des Zeitfaktors q den Einfluss des *Overheads* minimieren kann. In Anbetracht der Stabilität ist es für das untersuchte System nicht ratsam den Zeitfaktor auf $q > 5$ zu setzen. Mit steigendem $R_{\Delta x}$ sinkt der Vorteil durch die lokalen Operatoren stetig. Schon bei $R_{\Delta x} = 2$ nimmt dieser rapide ab, sodass die Laufzeitreduktion nur noch im einstelligen Prozentbereich liegt. Bei $R_{\Delta x} = 3$ ist der Laufzeitunterschied nicht mehr nennenswert. Hinsichtlich des *Splittings* des Operators liegen keine neuen Erkenntnisse vor. Es wird weiterhin dieselbe Submatrix zur Beschreibung der Ratengleichungen extrahiert, wodurch der relative Laufzeitgewinn bei 2 % verbleibt.

6.5.3 Einsatz von Multiprocessing und Multithreading

Da die Unterteilung in longitudinaler Richtung im Vergleich zur Unterteilung in transversaler Richtung in dem gegebenen System zu einer stärkeren Verkürzung der Laufzeit führt, wird im Anschluss die longitudinale Unterteilung angenommen. Im Abschnitt 6.4.3 wurde verifiziert, dass der *Overhead* bezüglich der Extraktion und Assemblierung in einem linearen System verringert werden kann. Dagegen belegt die Teilstudie aus dem Abschnitt 6.5.3 die enorme Beschleunigung der Rechenzeit für die Expansion. Diese Erkenntnisse werden für nichtlineare Systeme validiert. Dabei wird das *Operatorsplitting* vernachlässigt, um den Effekt der Parallelisierung deutlicher hervorzuheben.

Die Abbildung 6.24 demonstriert die Ergebnisse, wenn MP auf der CPU ausgenutzt wird. Im Falle des globalen Operators bleibt die Laufzeit unverändert aufgrund der standardisierten Parallelisierung in MATLAB über die BLAS-Bibliothek. Selbst der *Overhead* bleibt unbeeinträchtigt, da lediglich systembedingte Hintergrundprozesse darunter fallen. Deshalb wird der Fokus auf die lokalen Operatoren gerichtet. Hier lässt sich deutlich nachvollziehen, inwiefern das MT Einfluss auf die Approximation nimmt. Die Laufzeiten der mit der CPU parallelisierten lokalen Operatoren entsprechen nahezu denen der seriellen Ausführung ,abzüglich des *Overheads*, und sind sogar minimal geringer. Diese Tatsache lässt sich für jeden beliebigen Wert von $R_{\Delta x}$ feststellen. Es lässt sich nicht eindeutig sagen, ob dies auf die zeitliche Schwankung der Rechenleistung zurückzuführen ist oder darauf, dass mehr *Worker* im Einsatz sind und die Hintergrundprozesse effizienter gehandhabt werden.

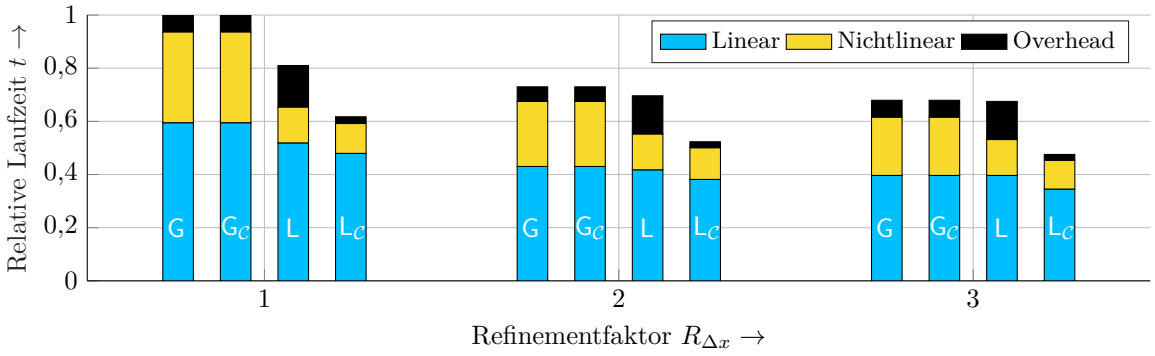
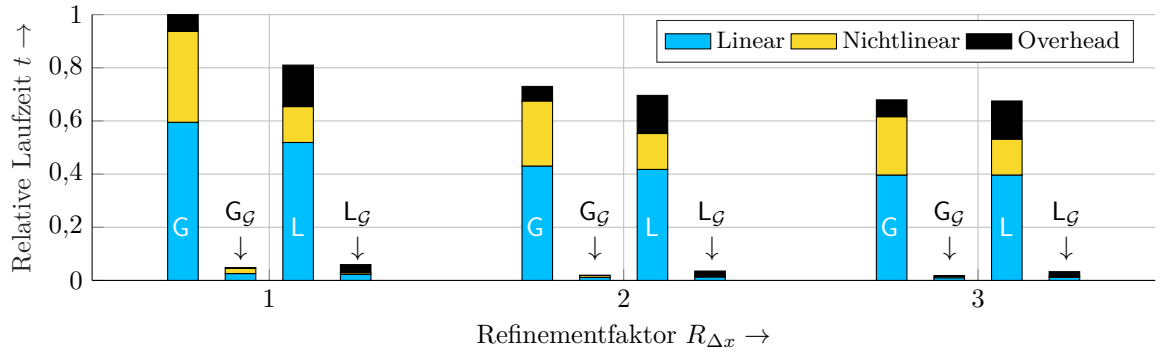
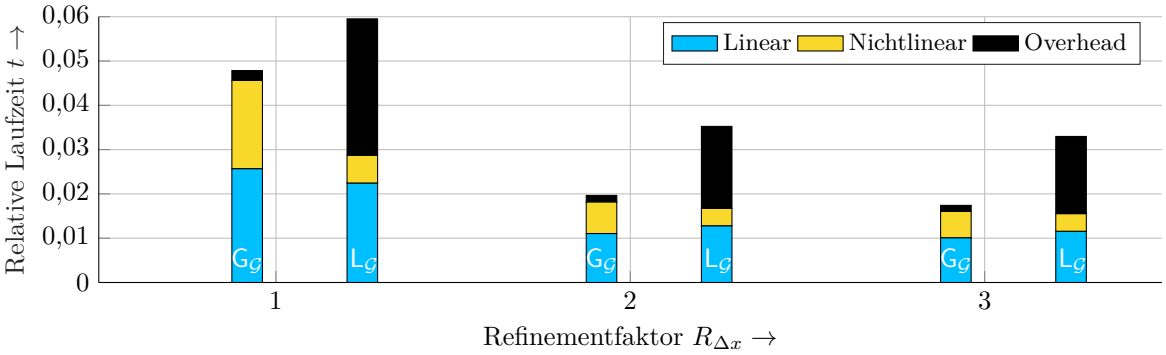


Abbildung 6.24: Laufzeitvergleich beim Einsatz von MP in dem 2D-DFB-Lasermodell.

In der Abbildung 6.25 sind die relativen Laufzeiten ersichtlich, wenn MT auf der GPU verfügbar ist. Die Abbildung 6.25a zeigt die Resultate in vollem Umfang hinsichtlich der Laufzeit. Da die GPU erwartungsgemäß durch die deutlich höhere Anzahl an Rechenkernen die Rechenzeit wesentlich stärker reduzieren kann als die CPU, wird in der Abbildung 6.25b ein verfeinerter Ausschnitt der Ergebnisse dargestellt.



(a) Relativen Laufzeiten ohne und mit MT.



(b) Zoom der relativen Laufzeiten zur besseren Übersicht der Ergebnisse mit MT.

Abbildung 6.25: Laufzeitvergleich beim Einsatz von MT in dem 2D-DFB-Lasermodell.

Ungeachtet des Refinementfaktors bleibt die Evaluierung mittels lokaler Operatoren derjenigen mit globalem Operator konsequent unterlegen. Der Effekt nimmt mit steigendem $R_{\Delta x}$ leicht zu.

Hauptverantwortlich dafür ist der *Overhead*, der in einem ungünstigeren Kompromiss mit der Nichtlinearität steht. Zur Verringerung des *Overheads* im Ablauf der lokalen Operatoren könnte theoretisch eine Kombination aus MP und MT genutzt werden. Dennoch wird darauf verzichtet, da die Expansion den Hauptteil der Evaluation ausmacht und in dem Abschnitt 6.5.3 bereits gezeigt wurde, dass das MT in diesem Kontext dominant ist.

6.5.4 Skalierung auf komplexere Systeme

Um die Übertragbarkeit der Ergebnisse des parallelisierten Ablaufs auf komplexere Systeme zu ermöglichen, wird die Anzahl der Abtastpunkte linear mit dem Skalierungsfaktor S erhöht. Folglich steigen sowohl die Komplexität als auch die Rechenzeit der Approximation. Insbesondere erreicht die Komplexität aufgrund der Skalierung mit S eine Größenordnung, die von der GPU aus der Tabelle 6.8 nicht mehr verarbeitet werden kann. Daher wird auf die leistungsfähigere GPU aus der Tabelle 6.9 zurückgegriffen, die von LiDO3 zur Verfügung gestellt wird. Die Extrapolation der Laufzeit ist in diesem Zusammenhang grundsätzlich möglich, wird jedoch durch schwer vorhersehbare Faktoren, wie Leerlaufzeiten der GPU bei Speicherüberschreitung [204], die Synchronisation der Threads innerhalb eines Thread-Blocks [205] sowie die thermische Drosselung infolge von Überhitzung [206], erheblich beeinträchtigt. Deshalb wurde im Abschnitt 6.3.1 der Ansatz verfolgt, eine hardwareunabhängige Komplexität herzuleiten, die sich ausschließlich auf die Anzahl der Multiplikationen bezieht.

Tabelle 6.9: Hardwarespezifikationen der GPU für die Skalierung auf komplexere Systeme.

GPU	VRAM	Kerne
Nvidia GeForce RTX 4080 SUPER	16 GB GDDR6	10240

Die Referenz der Komplexität in Abhängigkeit von S ist in der Abbildung 6.26 für das bisher untersuchte System bei $S = 1$ dargestellt. Die Normierung erfolgt anhand der Rechenzeit, die der globale Operator bei der Parallelisierung mit der CPU für $S = 1$ benötigt. Die Auswirkungen der Erhöhung von S fallen bei den PU unterschiedlich aus. Wird mit der CPU parallelisiert, erhöht sich die Laufzeit proportional zu S. Der Offset zwischen den Operatoren beträgt etwa 20 % mit Ausnahme von $S = 1$, was auf zeitliche Schwankungen der Rechenleistung zurückzuführen ist. Die Parallelisierung mittels der GPU offenbart, dass bis $S = 4$ der globale Operator eine geringfügig schnellere Entwicklung als mit den lokalen Operatoren ermöglicht. Für $S > 4$ liegen die Kurven übereinander. Diese Ergebnisse belegen, dass selbst beim Einsatz von MT der globale Operator nicht in allen Fällen überlegen ist. Dementsprechend können die lokalen Operatoren in Kombination mit PU auch in Systemen mit außergewöhnlich hoher Komplexität eingesetzt werden, ohne Einbußen bei der Laufzeit befürchten zu müssen.

Trotz der dargelegten Zusammenhänge sollte nicht unerwähnt bleiben, dass sich die Anzahl der Floating Point Operations per Second (FLOPS) in CUDA allein durch den Wechsel von der doppelten in die einfache Genauigkeit steigern lässt [205]. Gemäß *Mathworks* ist es in MATLAB R2024b nicht möglich, eine dünnbesetzte Matrix in den Datentyp *single* zu konvertieren, jedoch ist diese Funktionalität für MATLAB R2025a vorgesehen [207]. Angesichts der Speicheranforderungen bleibt somit nur die Verwendung des Datentyps *double* für dünnbesetzte Matrizen.

Dennoch könnte auf eine andere Programmiersprache wie Python ausgewichen werden, welche die Berechnung in CUDA mit dünnbesetzten Matrizen im Datentyp *single* erlaubt. Diesbezüglich zeigen empirische Studien, dass Python für die Berechnung von MVMs in diesem Kontext deutlich mehr Rechenzeit benötigt als MATLAB. Demnach ist der Vorteil, die einfache Genauigkeit anwenden zu können, gegen die nativ längere Rechenzeit bei der Auswertung von MVMs abzuwägen.

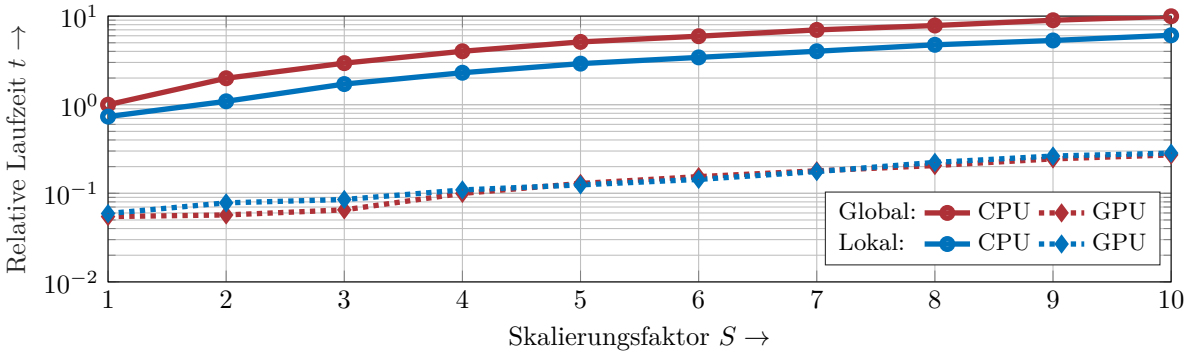


Abbildung 6.26: Übersicht der relativen Laufzeiten bei Parallelisierung mittels CPU und GPU für komplexere Systeme.

6.6 Zusammenfassung und Zwischenfazit

Das Konzept der lokalen Operatoren wurde in diesem Kapitel eingeführt und ausführlich analysiert. Das Ziel war, ein neues Verfahren zu entwickeln, das auf der Faber-Methode beruht und sowohl die Komplexität als auch die Laufzeit der Approximation reduziert. Konventionell wurde der zeitabhängige Propagator über einen globalen Operator approximiert. Dieses Vorgehen berücksichtigt jedoch nicht die unterschiedlichen Diskretisierungsweiten, wie sie bei nichtuniformen Gitterstrukturen zur Reduktion der lokalen Auflösung typisch sind. Entsprechend dem nichtuniformen Gitter lässt sich das Rechengebiet in mehrere Subgebiete unterteilen, wobei für jedes Subgebiet ein lokaler Operator aus dem globalen Operator extrahiert wird. Die separaten Entwicklungen finden weiterhin mit der Faber-Methode statt, allerdings mit den extrahierten Submatrizen und Subzustandsvektoren. Nachdem die Entwicklungsordnung gemäß dem Fehlerkriterium erreicht wurde, werden die Subzustandsvektoren durch eine elementweise Addition zum Zustandsvektor assembliert. Abgesehen von den Submatrizen ist besonders die reduzierte Entwicklungsordnung eines lokalen Operators aufgrund des nichtuniformen Gitters ausschlaggebend für den potenziellen Rechenzeitgewinn. Je höher die Diskretisierungsweite in einem Subgebiet gewählt wird, desto geringer fällt die erforderliche Entwicklungsordnung aus, um das vordefinierte globale Fehlerkriterium zu erreichen.

Zur Untersuchung der Performance wurde die Komplexität für einen Testfall theoretisch hergeleitet und sowohl in einem linearen als auch in einem nichtlinearen System numerisch validiert. Alle Prozesse, die nicht mit der Expansion in Zusammenhang standen, wurden als *Overhead* zusammengefasst. In der Theorie betraf dies die Extraktion und Assemblierung, während in der Praxis auch systembedingte Hintergrundprozesse hinzukamen. In Bezug auf die Implementierung wurden zwei Ansätze verfolgt. Der erste Ansatz sah eine dynamische Expansion vor, bei der die

Matrixordnung der Submatrizen mit jeder Entwicklungsordnung zunahm. Der zweite Ansatz sollte hingegen die Dynamik vollständig eliminieren, sodass eine statische Entwicklung mit der vollständig expandierten Submatrix des ersten Ansatzes realisiert wurde. Theoretisch weist der dynamische Ansatz die geringere Komplexität auf. Entgegen der Erwartung erwies sich jedoch die Implementierung des statischen Ansatzes in der Praxis als effizienter aufgrund der besseren Handhabung des unberechenbaren *Overheads*.

Mit Hilfe einer Parameterstudie wurde anhand der Approximation des Propagators in einem linearen System gezeigt, dass nur die Erhöhung der Zeitschrittweite effektiv zur Minderung des *Overheads* beiträgt. Zu diesem Zweck wurden der Einfluss von PU untersucht. Zunächst war die Intention den *Overhead* durch MT auf der CPU zu verringern. Danach sollte die Rechenzeit für die Expansion durch MP auf der GPU reduziert werden. Beide Maßnahmen konnten den Anforderungen gerecht werden, wenn auch das MP eine deutlich höhere Laufzeitreduktion aufgrund der hohen Anzahl an Rechenkernen ermöglichte. Auch im nichtlinearen Rechengebiet bestätigten sich die zuvor erzielten Resultate. Ferner wurde nun ein *Splitting* des Operators implementiert. Bedingt durch das Verhältnis zwischen den Größen des Lasermediums und des Rechengebiets ergab sich ein konstanter Rechenvorteil von 2 %. Zusätzlich hat sich aufgrund der Systemdimensionierung die Unterteilung in longitudinaler Richtung als vorteilhafter für die lokalen Operatoren im Vergleich zur transversalen Unterteilung erwiesen. Durch eine weitere Teilstudie sollte validiert werden, ob der globale Operator stets zu einer schnelleren Evaluierung führt, sobald mittels einer GPU parallelisiert wird. Dafür wurde die Anzahl der Diskretisierungspunkte im Rechengebiet proportional erhöht. Tatsächlich überlagerten sich die Laufzeiten der Operatoren ab einer gewissen Komplexität.

Die Tabelle 6.10 fasst die Ergebnisse aller Studien in diesem Kapitel zusammen, wobei jeweils die schnellere Methode aufgelistet ist. Insgesamt ist festzuhalten, dass die Evaluation der Faber-Methode mittels lokaler Operatoren derjenigen mit dem globalen Operator in nahezu allen Belangen überlegen ist, wenn ein nichtlineares System vorliegt.

Tabelle 6.10: Übersicht des zu präferierenden Operators gemäß des Systems in Abhängigkeit vom Modell, Ablauf, $R_{\Delta x}$, q und S.

Modell	**Ablauf**	$\boldsymbol{R_{\Delta x} = 1}$ $\boldsymbol{q = 1}$	$\boldsymbol{R_{\Delta x} = 1}$ $\boldsymbol{q > 1}$	$\boldsymbol{R_{\Delta x} > 1}$ $\boldsymbol{q = 1}$	$\boldsymbol{R_{\Delta x} > 1}$ $\boldsymbol{q > 1}$	$\boldsymbol{S > 1}$ $\boldsymbol{q > 1}$
Linear	Sequentiell	G	=	G	=	-
	MP	L	L	=	=	-
	MT	G	G	G	G	-
Nichtlinear	Sequentiell	-	L	-	=	-
	MP	-	L	-	L	-
	MT	-	G	-	G	=

Kapitel 7

Model Order Reduction

Die zuvor untersuchten Ansätze zur Verringerung der Komplexität haben sich im Wesentlichen auf die Faber-Methode beschränkt, sei es durch die Anpassung der Kontur um das skalierte Eigenwertspektrum oder durch die Anwendung der lokalen Operatoren. Eine Methode zur Reduktion der Matrixordnung der Systemmatrix ohne Einbußen an Genauigkeit wurde bislang nur in rudimentärer Form durch die Anwendung des Refinementfaktors zur Realisierung eines nichtuniformen Gitters untersucht. In Anbetracht dessen bietet das Konzept der Model Order Reduction (MOR) das Potenzial, die Evaluation deutlich zu beschleunigen. Nach aktuellem Stand wurden bislang keine Untersuchungen durchgeführt, in denen die Approximation mittels der Faber-Methode durch die MOR optimiert wurde. Dementsprechend wurde die Kompatibilität beider Verfahren erstmals in der zur Veröffentlichung eingereichten Publikation [PS8] analysiert. Im Folgenden werden die Erkenntnisse aus [PS8] ausführlich behandelt und erweitert. Als Einstieg in die Thematik wird die MOR eingeführt. Im Anschluss wird die Proper Orthogonal Decomposition (POD) erläutert, die ein spezifisches Verfahren der MOR und das Hauptverfahren der Untersuchung darstellt. Zudem wird die Erweiterung der POD zur Hybrid Proper Orthogonal Decomposition (HPOD) detailliert betrachtet.

7.1 Einführung in die MOR

Obwohl sich die FD, FEM und FVM zur Diskretisierung des Ortes in der numerischen Mathematik etabliert haben, teilen sie sich den Schwachpunkt sehr speicherintensiv zu sein. Bei all diesen Methoden steigt die Anzahl der Freiheitsgrade exponentiell mit der örtlichen Auflösung, was zu einem erheblichen Anstieg des Speicherbedarfs führt. Aus diesem Grund ist es erstrebenswert die Komplexität nicht nur nach der Diskretisierung, sondern bereits währenddessen oder im Idealfall sogar davor zu reduzieren [208]. In diesem Zusammenhang stellt die MOR einen vielversprechenden Ansatz zur effizienten Reduktion komplexer Systeme dar. Die Grundidee der MOR besteht darin, das Verhalten eines gegebenen Systems mit einem komprimierten Parametersatz so zu beschreiben, dass redundante Information vernachlässigt und eine effiziente Speicherverwaltung sowie eine Reduktion der Laufzeit ermöglicht werden. Obwohl das Konzept auf den ersten Blick überzeugend erscheint, zeigen sich bei der Umsetzung erhebliche Herausforderungen. Beispielsweise müssen in der Regelungstechnik Steuerungen in Echtzeit erfolgen, weshalb die Hardwareanforderungen so gering wie möglich gehalten werden sollen [209]. Weiterhin muss zwischen der Komplexität und der Genauigkeit der Approximation abgewogen werden, außerdem darf die Stabilität des Systems nicht beeinträchtigt werden. Gemäß [209] gibt es keine universelle MOR, die für jede Applikation umgesetzt werden kann. Diesbezüglich kann die MOR wie in der Tabelle 7.1 klassifiziert werden.

Tabelle 7.1: Klassifikationen der MOR.

Frequenzbereich	Zeitbereich	Zustandsraumdarstellung
Polynomapproximation	Parameterapproximation	Zustandsreduktion

Prinzipiell wird die MOR in zwei Phasen unterteilt. Die erste Phase ist die Offline-Phase und dient der Akquisition von Daten. In dieser Phase werden diskrete Momentaufnahmen bzw. Snapshots der unveränderten Datensätze im Full Order Model (FOM) gesammelt und in der charakteristischen Snapshotmatrix gespeichert, sodass währenddessen das vereinfachte Modell trainiert wird. Nach Abschluss dieser Phase erfolgt der Übergang in die Online-Phase, in der anstelle des FOMs das Reduced Order Model (ROM) mit einem komprimierten Datensatz für die Evaluation verwendet wird. Damit das ROM aus dem FOM abgeleitet werden kann, ist eine Projektion in den Unterraum erforderlich. Dahingehend ist die POD als Vertreter der Low-Rank Approximation (LRA) ein weit verbreitetes Verfahren, das eine solche Projektion durch die Anwendung der Singular Value Decompostion (SVD) ermöglicht [210].

7.2 Einführung in die POD

Um das Konzept der POD zu demonstrieren, werden zunächst die Methoden der SVD, der LRA und der Projektion in den Unterraum rudimentär erläutert. Dadurch werden wichtige mathematische Zusammenhänge nachvollziehbar und insbesondere die Reduktion der relevanten Größen hervorgehoben.

7.2.1 Singulärwertzerlegung

Die SVD [211] ist ein Verfahren aus der linearen Algebra, welches die Zerlegung der Matrix $\mathcal{H} \in \mathbb{R}^{\mathsf{m}\times\mathsf{n}}$ mit $\text{rang}(\mathcal{H}) = r \leq \min(\mathsf{m}, \mathsf{n})$ in drei spezifische Untermatrizen ermöglicht, wodurch bedeutende Eigenschaften von $\mathcal{H}$ erkennbar werden [212]. Bei der Anwendung der POD gilt standardmäßig $\mathsf{m} \ll \mathsf{n}$, weshalb der Fall einer quadratischen Matrix nicht behandelt wird. Die allgemeine Definition der SVD lautet

$$\mathcal{H} = U\Sigma V^{\dagger}. \tag{7.1}$$

Die Matrizen $U = [u_1, u_2, \ldots, u_r] \in \mathbb{R}^{\mathsf{m}\times\mathsf{m}}$ und $V^{\dagger} \in \mathbb{R}^{\mathsf{n}\times\mathsf{n}}$ in der Gleichung (7.1) sind orthogonale Matrizen, wobei $V^{\dagger}$ die adjungierte Matrix zu $V = [v_1, v_2, \ldots, v_r]$ darstellt. Die Elemente u_i sind die linken Singulärvektoren und geben die Moden des Systems wieder. Die Elemente v_i sind die rechten Singulärvektoren von $\mathcal{H}$. In der Matrix $\Sigma \in \mathbb{R}^{\mathsf{m}\times\mathsf{n}}$ sind die Singulärwerte σ_i von $\mathcal{H}$ enthalten. Konventionell werden die Singulärwerte in absteigender Reihenfolge $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_n > 0$ angegeben, die zugehörigen Energien mit σ_i^2 [213]. Gängige Anwendungsgebiete, in denen die SVD zum Einsatz kommt, sind neben der numerischen Mathematik [214], die Geophysik [215] die Bildverarbeitung [216] und in der jüngeren Vergangenheit das maschinelle Lernen [217]. Falls $r < \min(\mathsf{m}, \mathsf{n})$ gilt, kann die reduzierte SVD [218] angewandt werden. Dies erlaubt eine effizientere Berechnung, da $U_r \in \mathbb{R}^{\mathsf{m}\times r}$ und $V_r \in \mathbb{R}^{r\times\mathsf{n}}$ mit geringerer Matrixordnung resultieren. Ferner ergibt sich $\Sigma_r = \text{diag}(\sigma_1, \sigma_2, \ldots \sigma_r) \in \mathbb{R}^{r\times r}$, wodurch unnötige Nullmatrizen ohne Informationsverlust vernachlässigt werden können und Σ zu einer Diagonalmatrix wird.

7.2.2 Low-Rank Approximation

Die LRA [219] ist ein grundlegender Bestandteil der POD. Mit ihr kann die Matrix $\mathcal{H}$ durch die Matrix $\mathcal{H}_{\tilde{r}}$ mit dem reduzierten Rang $\tilde{r} < r$ approximiert werden. Gemäß dem Satz von Eckart-Young [220] kann die ideale Low-Rank Approximation bezüglich der Frobenius- und Spektralnorm durch die Anwendung der SVD konstruiert werden, allerdings ist die Verteilung der Daten in der Matrix entscheidend [221]. In diesem Sinne werden zunächst die ersten $\tilde{r}$ Singulärwerte von $\mathcal{H}$ entnommen und $\mathcal{H}_{\tilde{r}}$ anschließend über die SVD zerlegt:

$$\mathcal{H} \approx \mathcal{H}_{\tilde{r}} = U_{\tilde{r}} \Sigma_{\tilde{r}} V_{\tilde{r}}^{\dagger}. \tag{7.2}$$

Dadurch werden sowohl die Frobeniusnorm [222]

$$||\mathcal{H} - \mathcal{H}_{\tilde{r}}||_F = \sqrt{\sigma_{\tilde{r}+1}^2 + \sigma_{\tilde{r}+2}^2 + \ldots + \sigma_{\tilde{r}+r}^2} \tag{7.3}$$

als auch die Spektralnorm [223]

$$||\mathcal{H} - \mathcal{H}_{\tilde{r}}||_2 = \sigma_{\tilde{r}+1} \tag{7.4}$$

minimiert. Somit kann die Komplexität komprimiert und die Rechenzeit beschleunigt werden, während die relevanten Daten erhalten bleiben und eine minimale Fehlerordnung gewährleistet wird.

7.2.3 Projektion in den Unterraum

Bedingt durch die endliche Darstellbarkeit kontinuierlicher Systeme in der numerischen Mathematik ist die Approximation eines unendlichdimensionalen Funktionenraumes durch einen endlichdimensionalen Unterraum unvermeidlich. Hierbei stellt die Galerkin-Projektion ein bewährtes Verfahren dar, welches eine effiziente Approximation der Lösung ermöglicht, indem es das System auf einen geeigneten Unterraum projiziert [224]. Angenommen, das Verhalten eines Systems wird über eine PDGL beschrieben, welches wiederum örtlich diskretisiert wurde, so ergibt sich eine lineare DGL der Form

$$\frac{\partial}{\partial t}\vec{\Psi}(t) = P\vec{\Psi}(t). \tag{7.5}$$

In der Gleichung (7.5) repräsentiert P die orthogonale Projektionsmatrix des m-dimensionalen Funktionenraumes auf den r-dimensionalen Unterraum, die über die Spalten der Matrix U bestimmt werden kann:

$$P = U(U^T U)^{-1} U^T. \tag{7.6}$$

Zur Realisierung einer expliziten Berechnung wird die folgende Annäherung mit den Entwicklungskoeffizienten $\vec{p}$ angewandt:

$$\vec{\Psi}(t) \approx U\vec{p}(t). \tag{7.7}$$

Durch das Einsetzen der Gleichung (7.7) in die Gleichung (7.5) sowie die Multiplikation mit $U^{\dagger}$ von der linken Seite kann das DGL-System für die Entwicklungskoeffizienten $\vec{p}$ abgeleitet werden:

$$\frac{\partial}{\partial t}\vec{p}(t) = U^{\dagger} P U \vec{p}(t) = \tilde{P}\vec{p}(t) \tag{7.8}$$

Um die Lösung aus dem reduzierten r-dimensionalen Unterraum zurück in den ursprünglichen m-dimensionalen Funktionenraum zu projizieren, genügt die Multiplikation der Gleichung (7.8) mit der Matrix U.

7.2.4 Konzept der POD

Die POD, auch als Karhunen-Loeve Zerlegung bekannt, ist eine LRA, die es ermöglicht die Anzahl der Freiheitsgerade von zeitabhängigen PDGLs in numerischen Modellen zu verringern [225]. Das Konzept basiert darauf, einen Datensatz durch die Aufstellung einer Orthogonalbasis so zu charakterisieren, dass der quadratische Fehler zwischen der POD-Basis und dem ursprünglichen Datensatz minimiert wird. Die POD-Basis wird durch die örtliche und zeitliche Erfassung der Datensätze aufgestellt, was als Methode der Snapshots bekannt ist [226]. In diesem Kontext entsprechen die Datensätze den Moden der propagierenden elektromagnetischen Welle. Im Verlauf der Basisbildung werden die unveränderten Daten im FOM zunächst zu jedem Zeitpunkt spaltenweise in der Snapshotmatrix ψ, die synonym zur POD-Basis verwendet wird, gesammelt. Da die POD im diskreten Fall der SVD entspricht [210], wird ψ anschließend über die SVD zerlegt. Dies dient der Bestimmung der Singulärwerte. Anhand jener Singulärwerte kann beurteilt werden, ob eine Projektion in das ROM sinnvoll ist. Die POD findet u.a. Anwendung in der Strömungsmechanik [224], der Bildverarbeitung [227] sowie in der Mustererkennung [228]. Darüber hinaus gestattet die POD die Interpolation von Daten [229] als auch deren Extrapolation [230]. Für nichtlineare Systeme ermöglicht die Discrete Empirical Interpolation Method (DEIM) eine effiziente Berechnung [231].

Zur Anwendung der konventionellen POD müssen Snapshots des zeitabhängigen Zustandsvektors $\vec{\Psi}(t)$ in einem vordefinierten Zeitintervall aufgezeichnet werden, sodass die Snapshotmatrix ψ aufgestellt werden kann. Wenn die Ortsdiskretisierung des Rechengebiets gemäß Yee mit N Abtastpunkten durchgeführt wird und das Zeitintervall in x Zeitschritte unterteilt ist, ergibt sich die Zusammensetzung von ψ wie folgt:

$$\begin{aligned}\psi &= \begin{bmatrix} \vec{\Psi}(t_1), & \vec{\Psi}(t_2), & \dots & \vec{\Psi}(t_{\mathsf{x}}) \end{bmatrix} \\ &= \begin{bmatrix} \Psi(x_1,t_1) & \Psi(x_1,t_2) & \dots & \Psi(x_1,t_{\mathsf{x}}) \\ \Psi(x_2,t_1) & \Psi(x_2,t_2) & \dots & \Psi(x_2,t_{\mathsf{x}}) \\ \vdots & \vdots & \ddots & \vdots \\ \Psi(x_N,t_1) & \Psi(x_N,t_2) & \dots & \Psi(x_N,t_{\mathsf{x}}) \end{bmatrix}. \end{aligned} \tag{7.9}$$

Wie in der Gleichung (7.9) ersichtlich, setzt sich der Zustandsvektor $\vec{\Psi}$ zu einem festen Zeitpunkt t aus den diskretisierten ortsabhängigen Werten der Lösung Ψ einer PDGL zusammen. Diese Lösungen Ψ lassen sich durch die Überlagerung von $\tilde{r}$ Moden und den dazugehörigen Entwicklungskoeffizienten p approximieren, wobei $1 \leq \tilde{r} \leq r$ gilt [232]:

$$\Psi(x,t) \approx \Psi^{\tilde{r}}(x,t) = \sum_{i=1}^{\tilde{r}} p_i(t) u_i(x) \tag{7.10}$$

Aufgrund der erforderlichen Orthogonalität bei der Anwendung der POD lässt sich p durch das kanonische Skalarprodukt $\langle \cdot, \cdot \rangle$ zwischen den Moden u und den Teillösungen Ψ ermitteln:

$$p_i(t) = \langle u_i(x), \Psi(x,t) \rangle \tag{7.11}$$

Weil die Anzahl der berücksichtigten Moden $\tilde{r}$ frei wählbar ist, ist es wünschenswert, dass die betragsmäßigen Mittelwerte der Entwicklungskoeffizienten p_i mit steigendem Index i schnell gegen Null konvergieren. Auf diese Weise wird der Einfluss höherer Moden mit $i > \tilde{r}$ gezielt reduziert.

7.3 Parameterstudie zur POD

Die Kompatibilität der Faber-Methode mit der POD wurde bisher noch nicht erforscht. Zu diesem Zweck wird die Propagation einer elektromagnetischen Welle im homogen diskretisierten Filmwellenleiter aus der Abbildung 6.8 nach der Konstruktion der Snapshotmatrix extrapoliert. An den Rändern des Filmwellenleiters wird eine PEC-Randbedingung eingesetzt. Der Puls wird im Zentrum des Rechengebiets mit 3000 Diskretisierungspunkten in x- und y-Richtung sowie einer Diskretisierungsweite von $\Delta x = \Delta y = 10\,\mathrm{nm}$ initialisiert. Als Verfahren werden die FDTD- und Faber-Methode gegenübergestellt. Zur Klassifizierung der Extrapolation werden im Rahmen einer Parameterstudie der absolute Fehler

$$e_a(t) = |\vec{H}_z(t) - \vec{H}_{z,ref}(t)| \tag{7.12}$$

und der relative Fehler

$$e_r(t) = \frac{||\vec{H}_z(t) - \vec{H}_{z,ref}(t)||_\infty}{||\vec{H}_{z,ref}(t)||_\infty} \tag{7.13}$$

analysiert. Die verwendete $||\cdot||_\infty$-Norm ist dabei über $||\vec{H}_z(t)||_\infty = \max_i |\vec{H}_{z,i}(t)|$ definiert. Die Untersuchung erfolgt der Übersicht halber ausschließlich für die $\vec{H}_z$-Komponente. Die Referenzgröße $\vec{H}_z(t)$ wird mit der Faber-Methode bestimmt, die im Vergleich zum FDTD-Verfahren eine deutlich höhere zeitliche Auflösung ermöglicht [19]. Wie im Abschnitt 5.1.2 beschrieben, wird dazu der Entwicklungskoeffizient auf $|c_m| < 2{,}22507 \cdot 10^{-308}$ festgelegt. Dies entspricht dem kleinsten darstellbaren Wert in doppelter Genauigkeit in MATLAB. Für die Auswertung wird der Zeitfaktor $q = \Delta t / \Delta t_{CFL}$ herangezogen.

Intuitiv würden die Snapshots des Zustandsvektors $\vec{\Psi}$ unabhängig von den einzelnen Feldkomponenten als gemeinsamer Vektor zur Generierung der POD-Basis gesammelt werden. Aufgrund der stark unterschiedlichen Größenordnungen der Feldkomponenten wird stattdessen der Ansatz verfolgt, für jede Komponente eine eigene Snapshotmatrix zu erzeugen [233]:

$$\psi_{E_x} = [\vec{E}_x(t_1),\, \vec{E}_x(t_2),\, \ldots]^T = U_{E_x} \Sigma_{E_x} V_{E_x}^\dagger, \tag{7.14}$$

$$\psi_{H_y} = [\vec{H}_y(t_1),\, \vec{H}_y(t_2),\, \ldots]^T = U_{H_y} \Sigma_{H_y} V_{H_y}^\dagger, \tag{7.15}$$

$$\psi_{H_z} = [\vec{H}_z(t_1),\, \vec{H}_z(t_2),\, \ldots]^T = U_{H_z} \Sigma_{H_z} V_{H_z}^\dagger. \tag{7.16}$$

Würde dieser Ansatz nicht gewählt werden, wäre die Feldkomponente mit der geringsten Größenordnung bedingt durch den geringen Energiebeitrag in den POD-Moden unterrepräsentiert. Wegen der Kopplung des elektrischen Feldes mit dem magnetischen Feld ist eine steigende Fehlerrate zu erwarten.

Durch die Anwendung der Galerkin-Projektion aus dem Abschnitt 7.2.3 und der Entwicklung nach den POD-Moden aus der Gleichung (7.10) können der Zustandsvektor $\vec{\Psi}$ und die Systemmatrix $\mathcal{H}$ in einen reduzierten Unterraum projiziert werden. Dabei resultieren der reduzierte Zustandsvektor

$$\vec{\Psi}_r(t) = [\vec{p}_{E_x}(t), \vec{p}_{E_y}(t), \vec{p}_{H_z}(t)]^T. \tag{7.17}$$

sowie die reduzierte Zustandsmatrix

$$\mathcal{H}_r = \begin{bmatrix} 0 & 0 & \frac{U_{E_x}^\dagger D_y^H U_{H_z}}{\epsilon_0 \Delta y} \\ 0 & 0 & \frac{U_{E_y}^\dagger D_x^H U_{H_z}}{\epsilon_0 \Delta x} \\ \frac{U_{H_z}^\dagger D_y^E U_{E_x}}{\mu_0 \Delta y} & -\frac{U_{H_z}^\dagger D_x^E}{\mu_0 \Delta x} & 0 \end{bmatrix}. \tag{7.18}$$

Zur Einschätzung des Projektionsausmaßes in den reduzierten Unterraum werden pro Feldkomponente $M = 100$ POD-Moden verwendet. Während die Matrixordnung von $\mathcal{H}$ vor der Projektion 26801×26801 betrug, wurde sie nach der Reduktion auf 300×300 verringert. Dies entspricht einem Kompressionsfaktor von über 80000. Die Hardwarespezifikationen sind in der Tabelle 7.2 zusammengefasst.

Tabelle 7.2: Hardwarespezifikationen für die Berechnung der POD.

CPU	Taktrate	Cache	Kerne	GPU	VRAM	Kerne	RAM
Intel i9-14900K	3,2 GHz	L3 mit 36 MB	16	Nvidia GeForce RTX 4080 SUPER	16 GB GDDR6	10240	32 GB

7.3.1 Extrapolation der Faber- und FDTD-Methode

Für die Analyse der POD-Methode wird die gesamte Simulationszeit in eine Offline-Phase $t_{off} \in [0,\ 5000\Delta t_{CFL}]$ und eine anschließende Online-Phase $t_{on} \in [5000\Delta t_{CFL},\ 25000\Delta t_{CFL}]$ unterteilt. Erstere wird mit dem FOM und Letztere mit dem ROM evaluiert. Die Berechnung einer Phase ist nicht zwingend an eine bestimmte Methode gebunden, wodurch der Vergleich zwischen der FDTD- und der Faber-Methode objektiv gehalten wird. Dementsprechend resultieren vier zu betrachtende Fälle. Da der Approximationsfehler der Faber-Methode von der Zeitschrittweite abhängt, wird der Zeitfaktor zunächst auf $q = 10$ festgelegt und im weiteren Verlauf variiert. Im Gegensatz dazu wird die FDTD-Methode mit $q = 1$ evaluiert. Um die zeitliche Synchronisation mit der Faber-Methode zu wahren, wird nur jeder zehnte Zeitschritt als Snapshot gespeichert. In Anbetracht der Dauer der Offline-Phase ergeben sich $n_S = 500$ Snapshots.

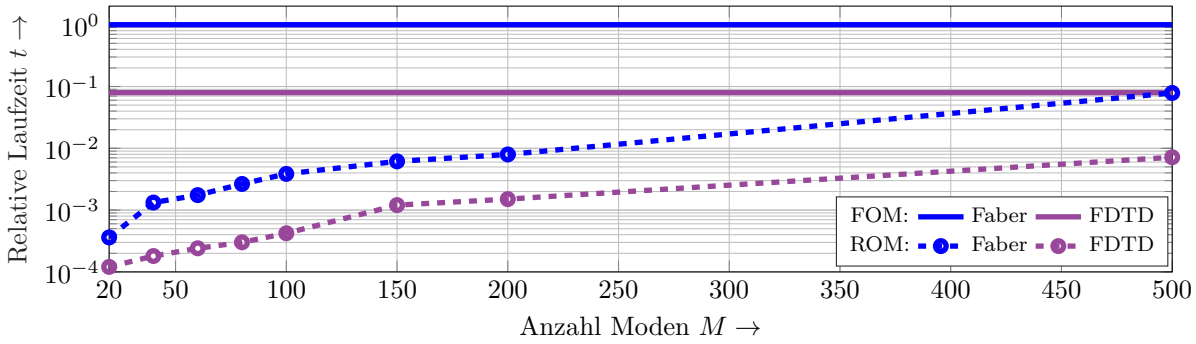
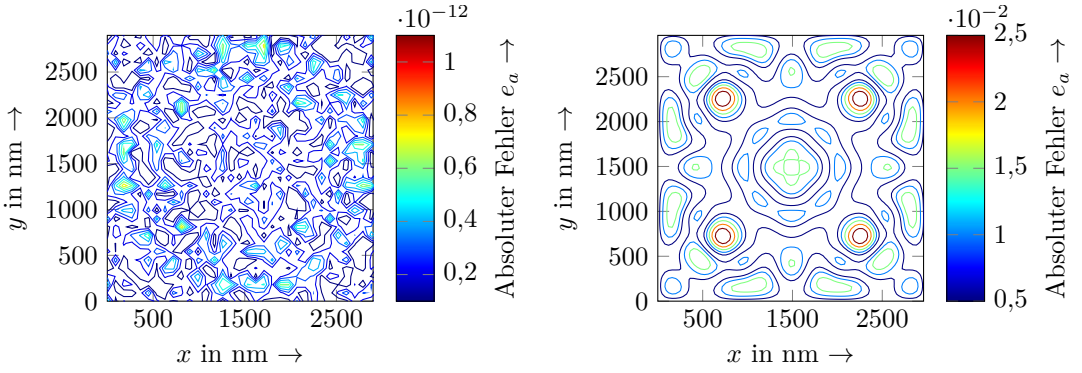
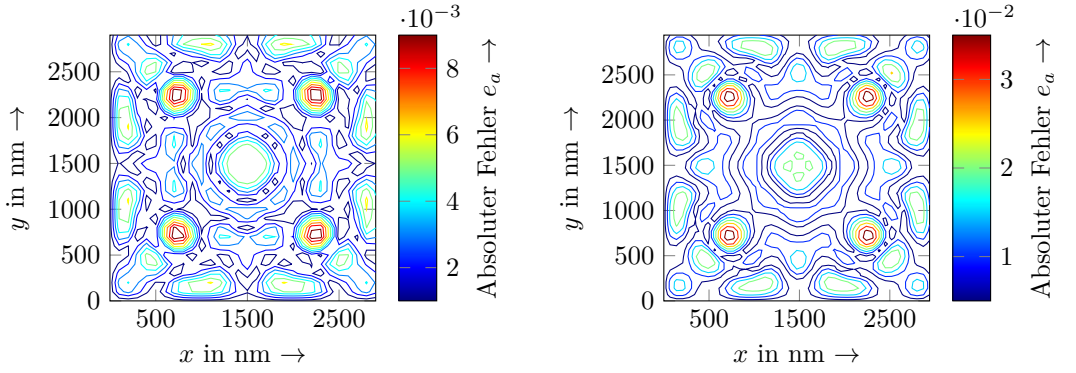


Abbildung 7.1: Laufzeitvergleich zwischen der Faber- und FDTD-Methode bei Anpassung der Moden M unter Einsatz von FOM und ROM.

In der Abbildung 7.1 ist die relative Laufzeit in Abhängigkeit der POD-Moden M zu sehen. Es fällt auf, dass die Laufzeit der Methoden während der Offline-Phase konstant bleibt, trotz des Anstiegs von M. Weil in dieser Phase das FOM zum Trainieren des Systems genutzt wird und keine Projektion in den Unterraum stattfindet, war dieses Verhalten absehbar. Sobald die Aufzeichnung der Snapshots allerdings abgeschlossen ist und in die Online-Phase übergegangen wird, lässt sich der unmittelbare Einfluss der POD beobachten. Wenn $M = 20$ Moden für die Reduktion im Unterraum verwendet werden, kann bei der Faber-Methode eine Beschleunigung der Rechenzeit um mehr als drei Größenordnungen erreicht werden. Dahingegen beträgt die Beschleunigung der Rechenzeit bei der FDTD-Methode knapp weniger als drei Größenordnungen. Für $M = 500$ liegt der Rechenzeitgewinn bei ungefähr einer Größenordnung. Die Zunahme der Rechenzeit beim ROM ist auf die steigende Ordnung der Snapshotmatrix zurückzuführen, wodurch mehr MVMs erforderlich werden. Dass die Faber-Methode sowohl beim FOM als auch beim ROM langsamer als die FDTD-Methode ist, bedingt durch den verhältnismäßig geringen Zeitfaktor q, ist in diesem Kontext unproblematisch. Der Fokus der Analyse liegt vorerst auf der Kompatibilität zwischen der Faber-Methode und der POD.



(a) Training mit Faber, Extrapolation mit Faber. **(b)** Training mit Faber, Extrapolation mit FDTD.



(c) Training mit FDTD, Extrapolation mit Faber. **(d)** Training mit FDTD, Extrapolation mit FDTD.

Abbildung 7.2: Visualisierung des absoluten Fehlers e_a der H_z-Komponente für $M = 500$ und $t = 25000\Delta t_{CFL}$, wenn die Faber- und FDTD-Methode abwechselnd für das Training und die Extrapolation bei der POD-Methode genutzt werden.

Inwiefern die Extrapolation mittels der POD die Genauigkeit beeinflusst, wird in der Abbildung 7.2 sichtbar. Im Sinne der höchsten Genauigkeit wurden $M = 500$ Moden festgehalten

und der absolute Fehler e_a am Ende der Simulation bei $t = 25000\Delta t_{CFL}$ untersucht. Die Abbildung 7.2a und Abbildung 7.2b zeigen die Ergebnisse, wenn die Faber-Methode für das Training genutzt wurde. Die Abbildung 7.2c und Abbildung 7.2d stellen die Ergebnisse dar, wenn mit der FDTD-Methode trainiert wurde. Die bei weitem höchste Genauigkeit aller Konstellationen wurde durch die ausschließliche Berechnung mit der Faber-Methode in der Abbildung 7.2a erzielt. Hier beträgt die absolute Abweichung zur Referenzlösung $e_a \approx 10^{-12}$ bei $t = 25000\Delta t_{CFL}$. Sobald eine der Phasen mit der FDTD-Methode berechnet wird, steigt e_a erheblich an. Findet die Extrapolation mit dem FDTD-Verfahren statt, folgt $e_a \approx 10^{-2}$, wie die Abbildung 7.2b und Abbildung 7.2d belegen.

Wird stattdessen nur mit der FDTD-Methode trainiert, kann der absolute Fehler um eine Größenordnung auf $e_a \approx 10^{-3}$ verringert werden. Dies spiegelt sich in der Abbildung 7.2d wider. Da die Extrapolation lediglich durch die Anwendung des FDTD-Verfahrens an Genauigkeit verliert, liegt der Schluss nahe, dass nicht die POD-Methode selbst, sondern das FDTD-Verfahren für die sinkende Genauigkeit verantwortlich ist. Im Folgenden wird entsprechend der gewonnenen Erkenntnisse auf die Auswertung mit der FDTD-Methode verzichtet und stattdessen ausschließlich die Faber-Methode verwendet.

7.3.2 Anpassung der Anzahl an Moden

Trivialerweise nimmt die Laufzeit mit der Anzahl der gespeicherten Moden M zu, wie bereits in der Abbildung 7.1 demonstriert wurde. Dennoch ließ sich daraus kein direkter Zusammenhang mit der Genauigkeit ableiten, weshalb M schrittweise für gezielt ausgewählte Werte erhöht wird. In der Abbildung 7.3 ist der relative Fehler e_r als Funktion von M im Verlauf der Extrapolation dargestellt. Für $M = 20$ ergibt sich $e_r \approx 10^{-2}$. Die sukzessive Erhöhung von M um 20 führt im Mittel zu einer Reduktion von e_r um etwa 1,5 Größenordnungen, was sich bis $M = 150$ beobachten lässt. Da sich bei $M = 150$ eine Sättigung hinsichtlich e_r einstellt, führt jede weitere Erhöhung lediglich zu einer verlängerten Rechenzeit ohne tatsächlichen Mehrwert. Allgemein verläuft e_r im Zeitverlauf schwankend, liegt am Ende der Online-Phase bei $t_{on} = 25000\Delta t_{CFL}$ jedoch stets etwa eine Größenordnung über dem Anfangswert bei $t_{off} = 5000\Delta t_{CFL}$. Die gelegentliche Reduktion von e_r lässt sich dadurch erklären, dass das Rechengebiet zunächst mit einer ideal reflektierenden Grenzschicht initialisiert wurde, was infolge von Reflexionen zu unvorhersehbaren Überlagerungen der Moden führen kann.

Nach wie vor ist ungeklärt, warum bei $M = 150$ eine Sättigung von e_r auftritt und ob sich die maximal sinnvolle Anzahl an POD-Moden nicht im Voraus bestimmen lässt. Zur Klärung dieser Fragestellung werden im Folgenden die Singulärwerte am Ende der Offline-Phase analysiert, wobei weiterhin von $n_S = 500$ Snapshots ausgegangen wird. Damit die reale Energie pro Snapshot ersichtlich wird, werden die Singulärwerte mit $\sigma_i/\sqrt{n_S}$ normiert. Die normierten Singulärwerte der H_z-Snapshotmatrix sind in der Abbildung 7.4 zu sehen. Der abfallende Verlauf ergibt sich aus der SVD, bei der die Singulärwerte entsprechend ihres Beitrags zur Gesamtdatenenergie in absteigender Reihenfolge sortiert sind. Wird standardmäßig mit doppelter Genauigkeit in MATLAB gerechnet, so verläuft der Abfall der normierten Singulärwerte bis einschließlich $M = 150$ annähernd linear. Dies deckt sich mit der Erkenntnis aus der Abbildung 7.1. Zwar nimmt die durch die Singulärwerte beschriebene Energie mit steigendem Index i weiterhin ab, jedoch ist der Abfall für $i > 150$ deutlich flacher.

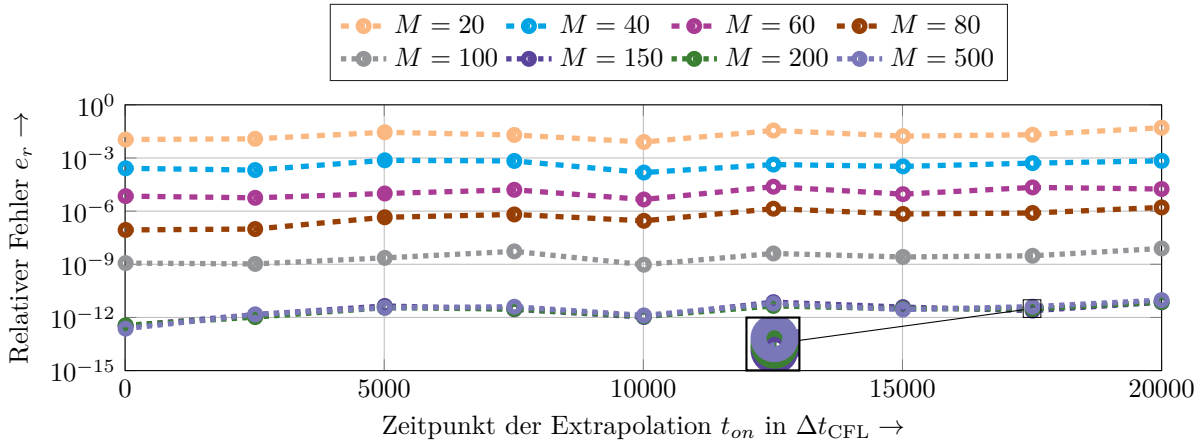


Abbildung 7.3: Relativer Fehler e_r des extrapolierten zeitabhängigen Propagators bei Anpassung der Anzahl an Moden M.

Um auszuschließen, dass die beobachtete Sättigung durch die POD- oder Faber-Methode verursacht wird, wird zusätzlich eine Auswertung mit der einfachen Genauigkeit durchgeführt. Bis einschließlich $i = 50$ stimmen die Singulärwerte in beiden Formaten weitgehend überein, ehe sie zunehmend voneinander abweichen. Bei $i = 100$ zeigt sich die erste Auffälligkeit im Verhalten der Auswertung mit einfacher Genauigkeit, da hier bereits die Sättigung des Singulärwertspektrums einsetzt. Dies deuten darauf hin, dass die beobachtete Sättigung auf numerische Rundungsfehler zurückzuführen ist. Diese Rundungsfehler können als Zufallsvariablen modelliert werden, auch wenn einzelne Rundungsoperationen deterministisch erfolgen. Unter der Prämisse, dass alle Rundungsfehler unabhängig sind, identisch verteilt auftreten und keine einzelne Rundung das Gesamtergebnis wesentlich dominiert, folgt aus dem zentralen Grenzwertsatz, dass die Gesamtabweichung von dem exakten Ergebnis näherungsweise normalverteilt ist. Diese Annahme erlaubt eine statistische Einschätzung der numerischen Genauigkeit und liefert eine plausible Erklärung für die zufällig anmutenden Abweichungen bei begrenzter numerischer Präzision. Sind diese Voraussetzungen erfüllt, lässt sich die erforderliche Anzahl an Moden gemäß [234] abschätzen.

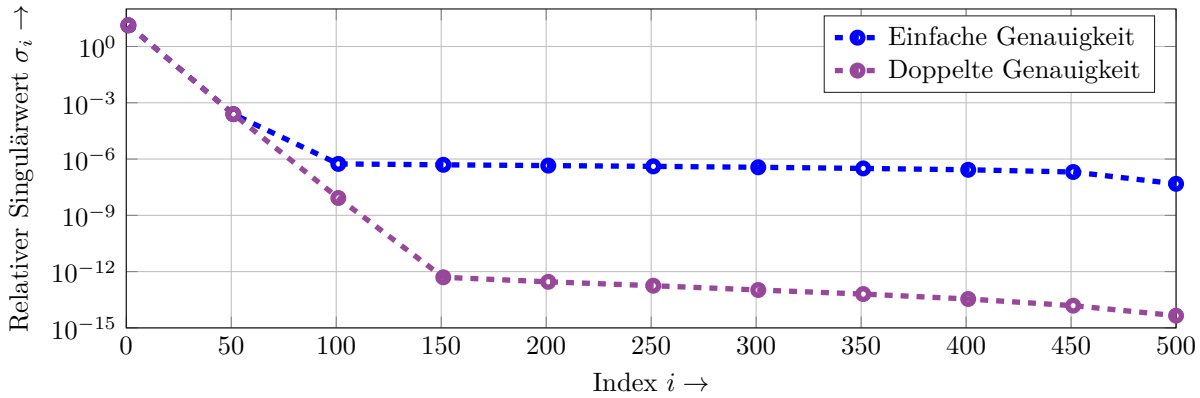


Abbildung 7.4: Relative Singulärwerte σ_i der H_z-Snapshotmatrix nach dem Trainingsintervall in Abhängigkeit der Genauigkeit.

7.3.3 Anpassung der Zeitschrittweite

Neben der Anzahl der POD-Moden M können auch die Zeitschrittweite Δt respektive der Zeitfaktor q variabel gewählt werden. Ein wesentliches Qualitätsmerkmal der Faber-Methode ist ihre garantierte Stabilität der Approximation, selbst bei außergewöhnlich hohen Zeitfaktoren, sofern die inhärenten Eigenwerte der Systemmatrix adäquat abgeschätzt wurden. Auf den ersten Blick scheint dies eine hervorragende Eigenschaft zu sein. Im Kontext der POD-Methode offenbart sich jedoch ein Widerspruch. Angenommen, die Dauer der Offline-Phase sei fest definiert, dann würde die Anzahl der maximal möglichen Snapshots um den Faktor q abnehmen. Des Weiteren können nur maximal so viele POD-Moden berücksichtigt werden wie Snapshots aufgezeichnet wurden.

Aus dem vorausgegangenen Abschnitt 7.3.2 geht hervor, dass M maßgeblich die Genauigkeit der Extrapolation bestimmt. Somit muss ein Gleichgewicht zwischen q und M gefunden werden. Aus diesem Grund werden in diesem Abschnitt drei Konstellationen und eine weitere im nächsten Abschnitt untersucht. Die erste Konstellation repräsentiert den Standardfall mit $\mathsf{k} = 1 \rightarrow q = 10 \wedge t_{off} \in [0, 5000\Delta t_{CFL}]$. Bei der zweiten Konstellation wird die Zeitschrittweite erhöht, sodass $\mathsf{k} = 2 \rightarrow q = 50 \wedge t_{off} \in [0, 5000\Delta t_{CFL}]$ gilt. Die dritte Konstellation mit $\mathsf{k} = 3 \rightarrow q = 50 \wedge t_{off} \in [0, 25000\Delta t_{CFL}]$ wird eingeführt, um bei erhöhtem q das gleiche Verhältnis zwischen q und der Dauer der Offline-Phase, wie im Referenzfall, herzustellen. Dementsprechend wird die Dauer der Offline-Phase um das Fünffache erhöht. Mit der verbleibenden Konstellation $\mathsf{k} = 4 \rightarrow q = 1 \wedge t_{off} \in [0, 500\Delta t_{CFL}]$ wird der umgekehrte Fall zu $\mathsf{k} = 3$ analysiert, indem sowohl q als auch t_{off} auf ein Zehntel ihres ursprünglichen Werts reduziert werden. Die Zusammenfassung der Konstellationen kann der Tabelle 7.3 entnommen werden.

Tabelle 7.3: Konstellationen für die POD-Parameterstudie.

k	q	$t_{off} \in$
1	10	$[0, 5000\Delta t_{CFL}]$
2	50	$[0, 5000\Delta t_{CFL}]$
3	50	$[0, 25000\Delta t_{CFL}]$
4	1	$[0, 500\Delta t_{CFL}]$

Die Resultate für die jeweiligen Konstellationen sind in der Abbildung 7.5 dargestellt. Trotz der Festlegung auf $M = 100$ zur Sicherstellung der Vergleichbarkeit variiert der relative Fehler e_r der Extrapolationen bei der Gegenüberstellung der ersten beiden Konstellationen um bis zu sechs Größenordnungen. Unter diesen Bedingungen ist die Anzahl an Snapshots bei $\mathsf{k} = 2$ zu gering, um bei $q = 50$ eine ausreichende Genauigkeit zu erzielen. Der Ansatz der Konstellation $\mathsf{k} = 3$, die Fehlerrate durch eine Verlängerung der Offline-Phase mit $q = 50$ zu reduzieren, zeigt eine deutliche Wirkung. Zu Beginn der Extrapolation befindet sich die Fehlerrate der dritten Konstellation auf dem Niveau der ersten, nimmt im weiteren Verlauf jedoch mit schwankender Tendenz zu. Verglichen mit $\mathsf{k} = 3$ ist e_r bei $\mathsf{k} = 2$ im Durchschnitt um vier Größenordnungen geringer.

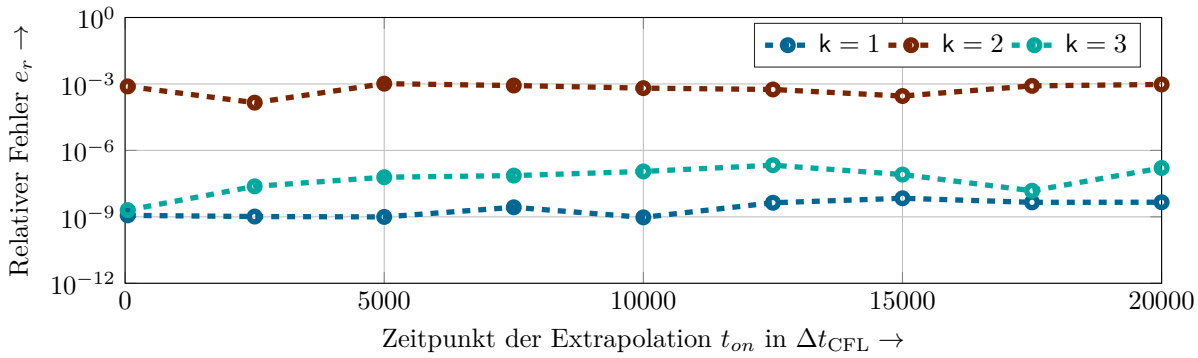


Abbildung 7.5: Relativer Fehler e_r des extrapolierten zeitabhängigen Propagators für die Konstellation $\mathsf{k} \in \{1, 2, 3\}$.

7.3.4 Anpassung der Intervalllänge

Wie im Abschnitt 7.3.3 gezeigt, beeinflussen sowohl die Dauer der Offline-Phase als auch der Zeitfaktor q den relativen Fehler e_r. Im Folgenden liegt das Hauptaugenmerk jedoch auf der Dauer der Offline-Phase. Im Vordergrund steht die Frage, ob sich die Offline-Phase proportional mit q verkürzen lässt, sofern keine Propagation mit großer Zeitschrittweite notwendig ist. In Abgrenzung zur Abbildung 7.5 werden die Konstellationen $\mathsf{k} = 1$ und $\mathsf{k} = 4$ aus der Tabelle 7.3 miteinander verglichen. In dieser Untersuchung wird eine feste Anzahl an $n_S = 500$ Snapshots und $M = 150$ Moden gewählt.

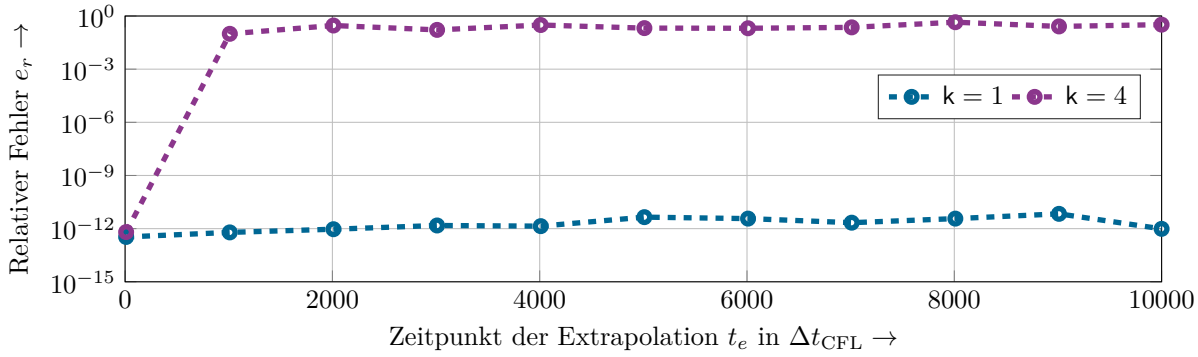


Abbildung 7.6: Relativer Fehler e_r des extrapolierten zeitabhängigen Propagators für die Konstellation $\mathsf{k} \in \{1, 4\}$.

Die Abbildung 7.6 zeigt die Ergebnisse für beide Konstellationen. Während bei der Referenzkonstellation $\mathsf{k} = 1$ ein relativer Fehler von $e_r \approx 10^{-12}$ erreicht wird, steigt die Fehlerrate bei $\mathsf{k} = 4$ rapide auf $e_r \approx 10^{-1}$ an. Die Ursache dafür ist, dass die ermittelten Moden aufgrund der kurzen Offline-Phase die tatsächlichen Lösungen in der Online-Phase nur unzureichend abbilden. Demnach kann die Offline-Phase trotz eines geringen Zeitfaktors nicht willkürlich verkürzt werden.

7.4 Erweiterung der POD zur HPOD

Die HPOD stellt eine optimierte Variante der POD dar. Sie wurde erstmals in [235] untersucht und in [236] weiterentwickelt. Die Neuheit der HPOD besteht darin, dass während der Simulation nicht nur ein einmaliger Wechsel vom FOM zum ROM erfolgt, sondern ein kontinuierlicher Übergang zwischen den beiden Modellen möglich ist. Besonders bei der Extrapolation des zeitabhängigen Propagators in komplexen Systemen kann die Dynamik zu einer unzureichenden Approximation mittels der POD-Moden führen. Vor diesem Hintergrund sieht das Konzept der HPOD einen Kompromiss zwischen der Genauigkeit und der Rechenzeit vor.

Dabei sollte nicht außer Acht gelassen werden, dass sich die erläuterte SVD aus dem Abschnitt 7.2.1 ausschließlich für die konventionelle POD eignet, jedoch nicht für die HPOD. Der Grund hierfür ist, dass bei der HPOD kontinuierlich neue Daten gesammelt und in der Snapshotmatrix ψ aufgezeichnet werden. Dies führt zu zwei wesentlichen Einschränkungen. Zum einen werden die Speicheranforderungen durch den kontinuierlichen Updateprozess von ψ größer. Zum anderen wird keine Rücksicht auf bislang bekannte Datensätze genommen, weshalb selbst bei der Erweiterung um neue Datensätze die SVD auf das komplette ψ angewandt wird. Deshalb wird bei HPOD die Incremental Singular Value Decompostion (ISVD) aus [237] verwendet, wo die bisherige Snapshotmatrix stärker gewichtet wird als die neu gesammelten Snapshots. Bei der ISVD wird die Komplexität mit $\mathcal{O}(r^2(\mathsf{m}+\mathsf{n}))$ abgeschätzt [237], während die Laufzeitschranke bei konventionellen Methoden mit $\mathcal{O}(\mathsf{m}^2\mathsf{n}+\mathsf{m}\mathsf{n}^2+\mathsf{n}^3)$ angegeben wird. Dies kann zwar durch Lanczos-Methoden auf $\mathcal{O}(\mathsf{m}\mathsf{n}r^2)$ reduziert werden [238], allerdings muss r dafür im Voraus bekannt sein [239].

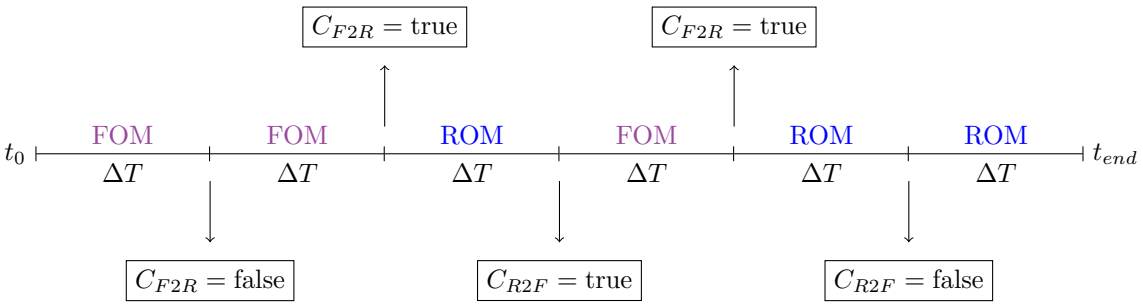


Abbildung 7.7: Illustration des Wechselkriteriums zwischen FOM und ROM bei der HPOD.

Zur Verdeutlichung der HPOD wird das Verfahren anhand eines Beispiels erläutert und in der Abbildung 7.7 visualisiert. Angenommen, die gesamte Simulationszeit im Intervall $[t_0, t_{end}]$ wird in sechs Teilintervalle der Länge ΔT unterteilt. Als Startbedingung wird festgelegt, dass das erste Teilintervall wie üblich mit dem FOM evaluiert werden muss, um einen unkomprimierten Datensatz zur Konstruktion der Snapshotmatrix bereitzustellen. Für die Entscheidung, ob das zweite Teilintervall weiterhin mit dem FOM oder stattdessen mit dem ROM berechnet werden soll, müssen Kriterien für den Wechsel zwischen den Modellen am Ende eines Teilintervalls definiert werden. Diesbezüglich wurden in [236] zwei Wechselkriterien formuliert. Das erste Kriterium C_{F2R} beschreibt den Wechsel vom FOM zum FOM und beruht auf der Verteilung der Singulärwerte σ_i:

$$C_{F2R,1} = \frac{\sum_{i=\mathsf{x}_1+1}^{r} {\sigma_i}^2}{\sum_{i=1}^{r} {\sigma_i}^2} \leq \varepsilon_1, \qquad (7.19) \qquad C_{F2R,2} = \frac{\sum_{i=\mathsf{x}_1+\mathsf{x}_2+1}^{r} {\sigma_i}^2}{\sum_{i=1}^{r} {\sigma_i}^2} \leq \varepsilon_2. \qquad (7.20)$$

Die Singulärwerte dienen dabei als Maß für den jeweiligen Einfluss der POD-Moden auf die Gesamtvarianz der zugrunde liegenden Daten der Snapshots. Offensichtlich ist C_{F2R} in zwei Bedingungen unterteilt, weshalb die beiden Schwellenwerte ε_1 und ε_2 mit $\varepsilon_1 > \varepsilon_2$ unterschritten werden müssen. Zwar wird mit $C_{F2R,1}$ die kumulierte Energie der POD-Moden im Teilintervall $[\mathsf{x}_1{+}1,\, r]$ relativ zur Gesamtenergie aller POD-Moden bestimmt, doch erlaubt dieser Wert allein keine Aussage über die Änderungsrate der Singulärwerte in Abhängigkeit von der Modenzahl. Um dennoch Einblicke in die Abfallrate der Energie zu gewinnen, müssen dafür die erweiterten Moden in dem Teilintervall $[\mathsf{x}_1{+}\mathsf{x}_2{+}1,\, r]$ in $C_{F2R,2}$ berücksichtigt werden [240]. Zudem werden $\mathsf{x}_2 > \mathsf{x}_1$ und $\varepsilon_1 > \varepsilon_2$ vorausgesetzt. Die relative Energieanteil der i-ten POD-Mode kann wie folgt bestimmt werden [241]:

$$\mathcal{E}_i = \frac{{\sigma_i}^2}{\sum_{m=1}^{r} {\sigma_m}^2} \qquad (7.21)$$

Da die Auswertung der SVD am Ende des ersten Teilintervalls ergibt, dass beide Bedingungen von C_{F2R} nicht erfüllt sind, verbleibt das zweite Teilintervall im FOM. Das System ist nicht ausreichend trainiert worden für eine Extrapolation, weshalb weitere Snapshots aufgezeichnet werden. Ferner wird am Ende des zweiten Teilintervalls anstelle der SVD die recheneffizientere ISVD verwendet. Weil nun C_{F2R} erfüllt ist, wird das dritte Teilintervall mit dem ROM approximiert. Hierbei wird auf das Sammeln von Snapshots verzichtet, weshalb keine POD-Moden ermittelt werden. Damit beurteilt werden kann, ob ein Wechsel vom ROM zurück zum FOM sinnvoll ist, wird nachfolgend das zweite Wechselkriterium C_{R2F} eingeführt [236]:

$$C_{R2F} = \frac{\sum_{m=\mathsf{x}_1+1}^{r} p_i^2}{\sum_{m=1}^{r} p_i^2} > \varepsilon. \qquad (7.22)$$

Im Gegensatz zu den Bedingungen aus der Gleichung (7.19) und Gleichung (7.20) wird die Gleichung (7.22) über die Entwicklungskoeffizienten p_i statt der Singulärwerte σ_i bewertet. Überschreiten die Entwicklungskoeffizienten der erweiterten x_2 Moden die Schwelle ε, nimmt die Dominanz der führenden Moden ab, zugunsten der höherwertigen Moden. In diesem Fall kann das ROM die wesentlichen Systemdynamiken nur unzureichend erfassen und es sollte zurück zum FOM gewechselt werden [240], wie am Ende des dritten Teilintervalls demonstriert. Die konventionelle POD würde den Wechsel nicht vollziehen, sodass eine fehlerhafte Extrapolation resultiert. Demnach muss die POD-Basis modifiziert werden. Die Evaluation der ISVD am Ende des vierten Teilintervalls beweist, dass C_{F2R} erneut erfüllt ist, nachdem eine ausreichende Anzahl an Snapshots erfasst wurde. Die Extrapolation der letzten beiden Teilintervall erfolgt auf Grundlage der POD-Basis vom Ende des vierten Intervalls, weil C_{R2F} nicht erfüllt ist.

7.4.1 Untersuchung der bekannten Wechselkriterien

Auf Grundlage der Parameterstudie im Abschnitt 7.3 wird nachfolgend die HPOD analysiert. Anstatt einmalig von der Offline-Phase in die Online-Phase überzugehen, soll die Transition nun

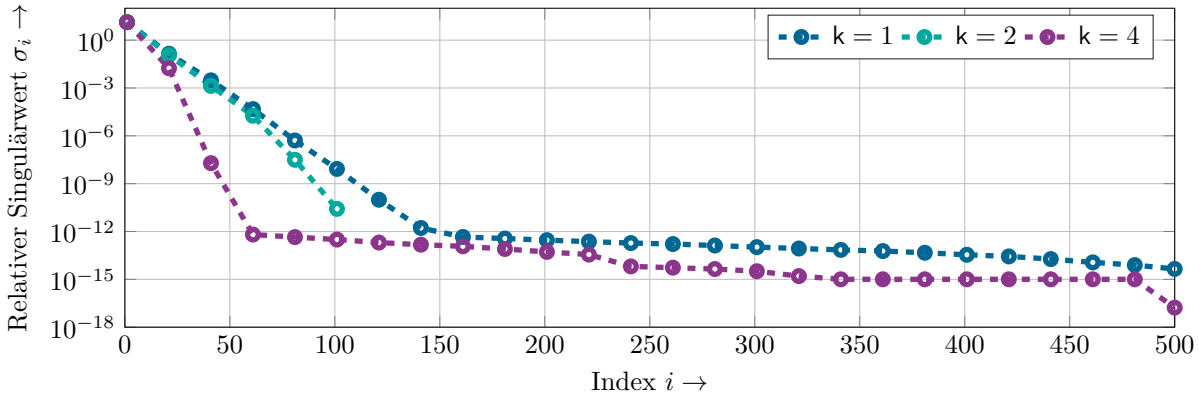


Abbildung 7.8: Singulärwerte σ_i der H_z-Snapshotmatrix bei ausreichend und unzureichend langem Training.

entsprechend der Abbildung 7.7 situativ zwischen dem FOM und dem ROM stattfinden. Im vorherigen Abschnitt 7.3 wurde ein statischer Ansatz gewählt, bei dem die Dauer der Offline-Phase im Voraus definiert wurde. Dies wäre auch für die HPOD realisierbar, indem zu Beginn der Approximation die Dauer der jeweiligen Offline- und Online-Phasen festgelegt wird. Der entscheidende Punkt hierbei ist, dass ein Fehler aufgrund eines unzureichend langen Trainings in der Offline-Phase zu einer Fehlerfortpflanzung beim Wechsel in die Online-Phase führt. Bedingt durch die Projektion in den Unterraum besteht keine Möglichkeit den Fehler zu korrigieren, sobald dieser einmal vorkommt.

Angesichts dieser Tatsachen wird ein dynamischer Ansatz verfolgt, bei dem die eine Unterbindung der Fehlerfortpflanzung möglich ist. Dazu bedarf es Kriterien, welche den Wechsel zwischen den Phasen vorgeben. In diesem Zusammenhang werden C_{F2R} und C_{R2F} aus dem Abschnitt 7.4 validiert. Ob die beiden Kriterien für die HPOD taugen, wird anhand des zuvor untersuchten Systems aus dem Abschnitt 7.3 validiert. Diesbezüglich wird die Referenzkonstellation mit $\mathsf{k} = 1$ mit den Konstellationen $\mathsf{k} = 2$ und $\mathsf{k} = 4$ gegenübergestellt, die bewiesenermaßen durch die zu kurze Offline-Phase oder durch die zu geringe Anzahl an Snapshots zu höheren Fehlerraten führen. Die Erwartung an die Wechselkriterien C_{F2R} und C_{R2F} ist, je stärker der Abfall der Singulärwerte ist, desto eher treten sie ein. Die Abbildung 7.8 verdeutlicht, dass es sich um eine fehlerhafte Annahme handelt.

Obwohl σ_i in allen drei Fällen rapide abfällt, tritt die Sättigung bei den Vergleichskonstellationen bei einem deutlich geringeren Index i ein als bei der Referenz. Somit reicht der alleinige Blick auf die Singulärwerte gemäß den vorgegebenen Kriterien nicht aus, und es müssen alternative Kriterien definiert werden.

7.4.2 Bestimmung eines neuen Wechselkriteriums

Anstelle von zwei Kriterien ist das Ziel, die HPOD künftig über ein einziges Wechselkriterium zu steuern. Die Grundidee besteht darin, mit Ausnahme des ersten Intervalls, jedes weitere Intervall direkt mit dem ROM zu berechnen. Dies ist durch die deutlichen Laufzeitunterschiede zwischen dem FOM und dem ROM motiviert, wie bereits in der Abbildung 7.1 gezeigt wurde. Zudem ist die Projektion in den Unterraum mit weniger Rechenzeit verbunden als die Propagation innerhalb

eines Intervalls. Sollte allerdings die Berechnung eines Intervalls mit dem ROM zu einem Anstieg des Fehlers führen, muss das aktuelle Intervall erneut mit dem FOM evaluiert werden. Das Konzept basiert auf einem ständigen Kompromiss zwischen der Laufzeit und Genauigkeit. Zur Verwirklichung dessen bedarf es eines neuen Kriteriums. Ein möglicher Ansatz wäre die stärkere Berücksichtigung der Entwicklungskoeffizienten p_i, ähnlich wie in [242] erforscht. Dazu wird der relative Fehler der i-ten POD-Mode des reduzierten Zustandsvektors herangezogen:

$$\Delta\mathcal{E}_i = \frac{\left|\frac{1}{n_S}(\sigma_{i\,E_x}^2 + \sigma_{i\,E_y}^2 + \sigma_{i\,H_z}^2) - (\overline{p}_{i,E_x}^2 + \overline{p}_{i,E_y}^2 + \overline{p}_{i,H_z}^2)\right|}{\left|\frac{1}{n_S}(\sigma_{i\,E_x}^2 + \sigma_{i\,E_y}^2 + \sigma_{i\,H_z}^2)\right|}. \tag{7.23}$$

Mit der Gleichung (7.23) werden die erwartete und tatsächliche Gesamtenergie in ein Verhältnis gesetzt. Die erwartete Energie wird durch die Eigenwertsumme der elektrischen und magnetischen Feldkomponenten bestimmt. Im Gegensatz dazu wird die tatsächliche Energie durch die Quadratsumme der Zeitmittelwerte der Entwicklungskoeffizienten p_i über ein Intervall berechnet. Mittels eines Schwellenwertes in der Form von

$$\max_i \Delta\mathcal{E}_i \leq \varepsilon_h \tag{7.24}$$

wird überprüft, ob eine Nachberechnung durch das FOM notwendig ist. Dies wäre der Fall, wenn eine zu starke Abweichung zwischen der erwarteten Energie und der tatsächlichen Energie festgestellt wird. Prinzipiell gilt zu beachten, dass die Entwicklungskoeffizienten sich ausschließlich auf ein einziges Intervall beschränken, wohingegen die Eigenwerte alle vorherigen Intervalle umfassen. Die Gesamtenergien nähern sich somit lediglich der Summe der Eigenwerte an, ohne jedoch zu konvergieren.

7.4.3 Implementierung

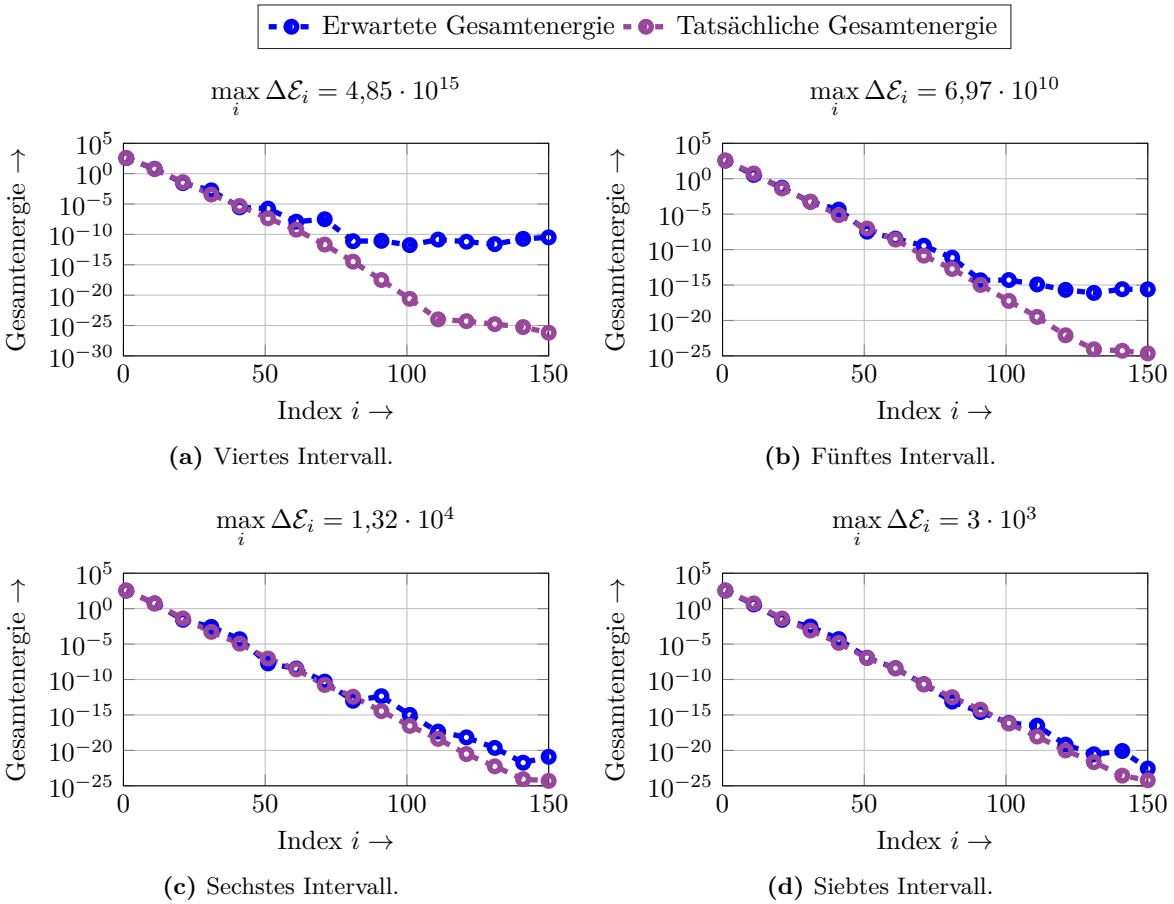
In der folgenden Untersuchung wird die Eignung des neu definierten Wechselkriteriums aus dem Abschnitt 7.4.2 für die HPOD untersucht. Zur Sicherstellung eines objektiven Vergleichs mit der Untersuchung aus dem Abschnitt 7.3 werden dasselbe Rechengebiet und dieselben Parameter übernommen. Der einzige Unterschied zur vorherigen Analyse liegt in der Aufteilung zwischen der Offline- und Online-Phase. Anstatt die Phasen statisch vorzugeben, können sie sich im Rahmen der Simulation dynamisch abwechseln. Für die Dauer der Simulation werden insgesamt 50 Intervalle evaluiert. Bezogen auf die gesamte Simulationsdauer von $t = 25000\Delta t_{CFL}$ ergibt sich für jedes Intervall eine Länge von $\tilde{t} = 500\Delta t_{CFL}$. Die Faber-Methode wird mit $q = 10$ implementiert. Obwohl bereits gezeigt wurde, dass die FDTD-Methode eine signifikant höhere Fehlerrate aufweist als die Faber-Methode, wird sie zur besseren Einordnung erneut berücksichtigt. Wie zuvor wird mit $q = 1$ propagiert und aus Gründen der Synchronität nur jeder zehnte Zeitschritt als Snapshot festgehalten. Da als Kompromiss zwischen der Rechenzeit und der Genauigkeit $M = 150$ POD-Moden verwendet werden, müssen mindestens die ersten drei Intervalle mit dem FOM ausgewertet werden.

Basierend auf der mit steigendem Index i abnehmenden Energie der Singulärwerte σ_i wird angenommen, dass $\Delta\mathcal{E}_i$ mit steigendem i zunimmt. Demnach muss der Schwellenwert ε_h aus der Gleichung (7.24) bedacht gewählt werden. Ein zu hoher Wert für ε_h führt zu einer vermeintlich fehlerhaften Berechnung mit dem ROM, wie sie mit dem FOM nicht auftreten würde. Umgekehrt kann eine zu niedrige Schwelle im schlimmsten Fall darin ausarten, dass sie nie unterschritten

wird, wodurch jedes Intervall nach der initialen Berechnung mit dem FOM stets erneut mit dem FOM evaluiert werden müsste. Im Rahmen von [PS8] hat sich auf Basis empirischer Studien ein Schwellenwert von $\varepsilon_h \leq 10^4$ als geeignet erwiesen.

7.4.4 Ergebnisse

In der Abbildung 7.9 werden die erwartete und die tatsächliche Gesamtenergie des Systems gegenübergestellt. Weil für die ersten drei Intervalle die Berechnung mit dem FOM vorgegeben ist, beginnt die Betrachtung ab dem vierten Intervall, wie in der Abbildung 7.9a zu sehen ist. Die Energien des fünften Intervalls sind in der Abbildung 7.9b dargestellt, die des sechsten Intervalls in der Abbildung 7.9c und die des siebten Intervalls in der Abbildung 7.9d. Offensichtlich gilt $\Delta\mathcal{E}_i \gg \varepsilon_h$ in dem vierten und fünften Intervall. Es ist bemerkenswert, dass $\Delta\mathcal{E}_i$ nach einem Intervall bereits um fünf Größenordnungen abnimmt. Diese Abnahme ist im sechsten Intervall besonders ausgeprägt, wo nur noch $\Delta\mathcal{E}_i > \varepsilon_h$ gilt und eine Reduktion um sechs Größenordnungen zu beobachten ist. Ab dem siebten Intervall kann erstmals $\Delta\mathcal{E}_i \leq \varepsilon_h$ festgestellt werden. Demnach ist das Kriterium erfüllt, und es wird keine erneute Berechnung des Intervalls mit dem FOM benötigt.



(a) Viertes Intervall.

(b) Fünftes Intervall.

(c) Sechstes Intervall.

(d) Siebtes Intervall.

Abbildung 7.9: Es sind die erwarteten und tatsächlichen Gesamtenergien des vierten bis siebten Intervalls abgebildet. Im letzten Intervall ist das Wechselkriterium mit $\max_i \mathcal{E}_i < \varepsilon = 10^4$ erfüllt, sodass vom FOM in das ROM gewechselt werden kann.

Ob das Wechselkriterium auch für die folgenden Intervalle valide bleibt, wird mit der Tabelle 7.4 ersichtlich. Hier zeigt insbesondere die Faber-Methode beachtliche Resultate. Zum einen muss nur das zehnte Intervall mit dem FOM wiederholt werden, nachdem ab dem siebten Intervall die Evaluierung mit dem ROM vorerst ausreichend ist. Zum anderen kann durch den Wechsel zwischen den Modellen eine Laufzeitbeschleunigung von bis zu 180 erzielt werden. Im Gegensatz dazu stellt sich bei der FDTD-Methode die Lage jedoch etwas anders dar. Neben den ersten sechs Intervallen ist die Berechnung von drei weiteren Intervallen mit dem FOM notwendig. Zudem liegt die maximal erreichbare Laufzeitreduktion bei 80. Allerdings sollte die durchschnittliche Rechenzeit der FDTD-Methode besonders hervorgehoben werden, da sie stets die der Faber-Methode übertrifft, unabhängig von der Wahl des Modells.

Tabelle 7.4: Überblick der Laufzeiten bei der HPOD unter Einsatz von Faber und FDTD, wenn für die Evaluation der 50 Intervalle das FOM und ROM verwendet werden.

Methode	Modell	Intervall	∅ Zeit in s	Faktor
Faber	FOM	[1-6,10]	4,464	180
	ROM	[7-9,11-50]	0,0248	
FDTD	FOM	[1-6,8,28,49]	0,36	80
	ROM	[7,9-27,29-48,50]	0,0045	

Ergänzend zur Tabelle 7.4 wird der relative Fehler e_r eines Intervalls in Bezug auf die jeweilige relative Laufzeit aufgetragen. Im Falle der Faber-Methode wird deutlich, dass die sieben Berechnungen mit dem FOM über 90 % der gesamten Rechenzeit ausmachen. Dahingegen repräsentieren die verbleibenden 43 Intervalle mit dem ROM nur einen Anteil im einstelligen Prozentbereich. Darüber hinaus führt die erste eigenständige Berechnung eines Intervalls mit dem ROM zu einer Fehlerzunahme, sodass $e_r \approx 10^{-15}$ auf $e_r \approx 10^{-11}$ ansteigt.

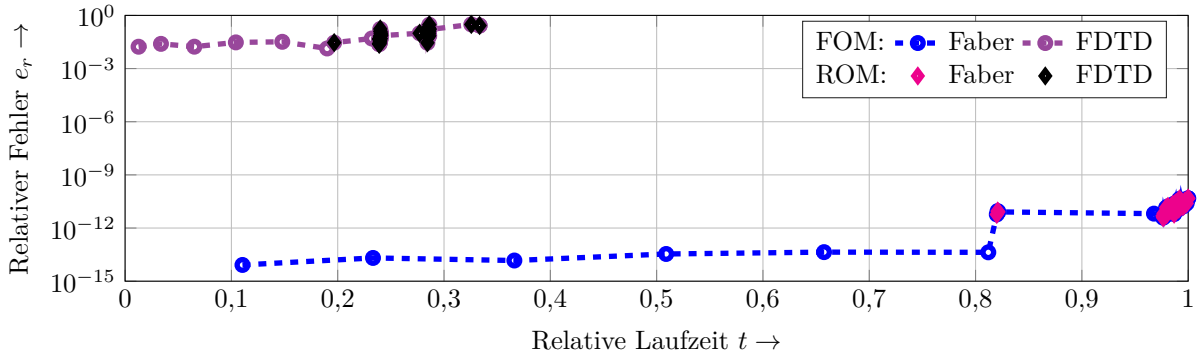


Abbildung 7.10: Gegenüberstellung des relativen Fehlers e_r mit der relativen Laufzeit unter Einsatz der Faber- und FDTD-Methode bei der HPOD. In der Tabelle 7.4 sind die angewandten Modelle zur Evaluation eines Intervalls aufgelistet.

Angesichts der im weiteren Verlauf reduzierten Laufzeit und der nur minimal steigenden Fehlerfortpflanzung stellt dies einen vertretbaren Kompromiss dar. Obwohl das FDTD-Verfahren mit

der HPOD-Methode rund 65 % schneller ist, bleibt es deutlich ungenauer. Bereits die Auswertung des ersten Intervalls mit dem FOM führt zu einem Fehler von $e_r \approx 10^{-2}$. Auf Basis dessen kann das weitere Sammeln von POD-Moden höchstens verhindern, dass e_r weiter ansteigt. Wird jedoch über mehrere Intervalle hinweg auf das FOM verzichtet, ist die Zunahme von e_r unvermeidlich.

7.5 Numerische Evaluierung der HPOD

Zur Untersuchung der Effizienz der HPOD in einem realistischen System wird im Folgenden ein 2×2 Multimode Interference (MMI)-Koppler [243] implementiert, der u.a. in optischen Schaltern und Multiplexern zum Einsatz kommt [244]. Eine schematische Darstellung des Systems ist in der Abbildung 7.11 zu sehen. Im Rahmen der Studie ist der Brechungsindex außerhalb eines Wellenleiters mit $n_1 = 1{,}44$ und innerhalb eines Wellenleiters mit $n_2 = 1{,}45$ festgelegt. Für die Breite des Rechengebiets wird $\mathsf{w} = 5000\Delta x$ gewählt. Die Teillängen der jeweiligen Subgebiete betragen $\mathsf{l}_1 = 50\Delta y$, $\mathsf{l}_2 = 40\Delta y$ und $\mathsf{l}_3 = 20\Delta y$. Ferner wird das System mit einer Wellenlänge von $\lambda = 1{,}55\,\mu\text{m}$ angeregt, wobei die Anregung nahezu aller ausbreitungsfähigen Grundmoden im oberen Wellenleiter in der Nähe der PEC-Grenzschicht erfolgt. Präziser werden 300 ausbreitungsfähige Grundmoden angeregt, worauf im weiteren Verlauf noch näher eingegangen wird. Zur Vermeidung numerischer Dispersion werden die Diskretisierungsweiten so definiert, dass $\Delta x = \Delta y = 100\,\text{nm} \leq \lambda/10$ gilt. Generell weisen die Grenzschichten in Länge und Breite jeweils $\mathsf{w}_0 = 5\Delta x = \mathsf{l}_0 = 5\Delta y$ auf. Des Weiteren dienen die CFS-PML mit $\alpha_x = 0$, $\kappa_x = 1$ und $\sigma_x = 85626$ zur Unterdrückung der Reflexionen an den Rändern des Rechengebiets. Die verbleibenden Parameter der CFS-PML können dem Abschnitt 3.3.2 entnommen werden. Somit verbleibt ausschließlich der rechte Rand des Systems reflektierend. Aus Gründen der Konsistenz mit den vorherigen Analysen wird eine 2D-Wellenleiterstruktur beibehalten. Die erzielten Ergebnisse lassen sich jedoch auch auf eine 3D-Struktur übertragen, da sich lediglich die Anzahl der Diskretisierungspunkte respektive der Yee-Zellen erhöht. Damit die elektromagnetische Welle im zugrunde liegenden System propagieren kann, muss der Zustandsvektor angepasst werden [191]. Dementsprechend wird fortan $\vec{\Psi} = [H_x, E_y, H_z]^T$ verwendet. Die Initialisierung der Grundmode der E_y- und H_z-Komponente in dem oberen Wellenleiter wird wie folgt vorgegeben:

$$E_y = \begin{cases} \cos\left(\frac{\mathsf{u}_1}{\mathsf{l}_1}\right) \cdot \exp\left(\frac{\mathsf{u}_2}{\mathsf{l}_1} \cdot \left(y + \frac{\mathsf{l}_1}{2}\right)\right) & \text{, überhalb des Wellenleiters} \\ \cos\left(\frac{\mathsf{u}_1}{\mathsf{l}_2} \cdot y\right) & \text{, innerhalb des Wellenleiters} \\ \cos\left(\frac{\mathsf{u}_1}{\mathsf{l}}\right) \cdot \exp\left(\frac{\mathsf{u}_2}{\mathsf{l}} \cdot \left(-y + \frac{\mathsf{l}}{2}\right)\right) & \text{, unterhalb des Wellenleiters,} \end{cases} \tag{7.25}$$

$$H_z = \begin{cases} \frac{j \cdot \mathsf{u}_2}{\mathsf{l}_1 \cdot k_0} \cdot \sqrt{\frac{\epsilon_0}{\mu_0}} \cdot \cos\left(\frac{\mathsf{u}_1}{\mathsf{l}_1}\right) \cdot \exp\left(\frac{\mathsf{u}_2}{\mathsf{l}_1} \cdot \left(y + \frac{\mathsf{l}_1}{2}\right)\right) & \text{, überhalb des Wellenleiters} \\ -\frac{j \cdot \mathsf{u}_1}{\mathsf{l}_2 \cdot k_0} \cdot \sqrt{\frac{\epsilon_0}{\mu_0}} \cdot \sin\left(\frac{\mathsf{u}_1}{\mathsf{l}_2} \cdot y\right) & \text{, innerhalb des Wellenleiters} \\ \frac{j \cdot \mathsf{u}_2}{k_0 \cdot \mathsf{l}} \cdot \sqrt{\frac{\epsilon_0}{\mu_0}} \cdot \cos\left(\frac{\mathsf{u}_1}{\mathsf{l}}\right) \cdot \exp\left(\frac{\mathsf{u}_2}{\mathsf{l}} \cdot \left(-y + \frac{\mathsf{l}}{2}\right)\right) & \text{, unterhalb des Wellenleiters.} \end{cases} \tag{7.26}$$

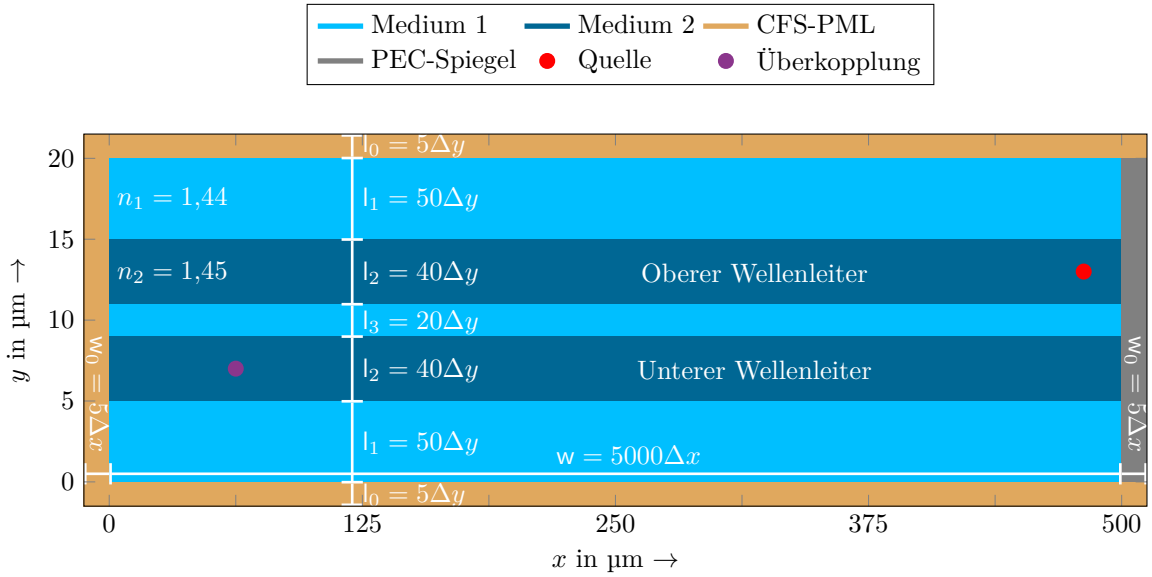


Abbildung 7.11: Praktisches Rechengebiet zur Modellierung der Modenkopplung zwischen zwei symmetrischen Wellenleitern.

Die Länge des gesamten Rechengebiets l in der Gleichung (7.25) und Gleichung (7.26) ergibt sich als Summe $\mathsf{l} = \mathsf{l}_1 + \mathsf{l}_2 + \mathsf{l}_3$ der Längen der einzelnen Subgebieten. Weiterhin lassen sich die Hilfsvariablen über

$$\mathsf{u}_1 = \mathsf{l}_1 \cdot \sqrt{{n_2}^2 \cdot {k_0}^2 - \beta^2}, \tag{7.27}$$

$$\mathsf{u}_2 = \mathsf{l}_1 \cdot \sqrt{\beta^2 - {n_1}^2 \cdot {k_0}^2} \tag{7.28}$$

bestimmen, wobei die Phasenkonstante durch

$$\beta = k_0 \cdot n_{eff} = \frac{2\pi}{\lambda} \cdot n_{eff} \tag{7.29}$$

als Funktion der Kreiswellenzahl k_0 definiert ist. Der effektive Brechungsindex $n_{eff} = 1{,}4459$ wurde für das System durch eine Eigenmodenanalyse basierend auf der Helmholtz-Gleichung bestimmt [245]. Sofern die E_y-Komponente aus der Gleichung (7.25) bekannt ist, kann die H_x-Komponente über den folgenden Zusammenhang berechnet werden [191]:

$$H_x = -\frac{\beta \cdot \epsilon_0 \cdot \omega}{{k_0}^2} \cdot E_y = n_{eff} \cdot \sqrt{\frac{\epsilon_0}{\mu_0}} \cdot E_y. \tag{7.30}$$

Aufgrund der in Abschnitt 7.3 nachgewiesenen Kompatibilität zwischen der Faber-Methode und der HPOD wird der Zeitfaktor auf $q \in \{10, 100\}$ erhöht. Das System ist so konzipiert, dass zehn Intervalle der Länge $t = 800\Delta t_{CFL}$ erforderlich sind, damit eine Kopplung der Moden zwischen den Wellenleitern erfolgt. Eine Übersicht der verwendeten Hardwarespezifikationen ist in Tabelle 7.2 dargestellt.

Obwohl sich die Konstruktion der Snapshotmatrix in den vorherigen Untersuchungen gemäß der Gleichung (7.9) bewährt hat, ist in dieser Analyse eine Modifikation der Snapshotmatrix unumgänglich. Empirisch durchgeführte Studien im Rahmen von [PS8] haben gezeigt, dass eine sehr hohe Anzahl an Snapshots erforderlich ist, um die Berechnung vollständig mit dem ROM

zu ermöglichen. Infolgedessen steigt der Speicherbedarf exponentiell an, wodurch der Vorteil der reduzierten Rechenzeit durch die Extrapolation weitgehend aufgehoben wird. Die Ursache dafür liegt an der Größe des Rechengebiets und der verhältnismäßig langsamen Propagation der elektromagnetischen Welle. Zwar bilden die einzelnen Snapshots das System ab, doch sind die für die Hauptenergie relevanten Einträge aufgrund der vektoriellen Implementierung des Zustandsvektors über den gesamten Vektor verteilt. Dadurch lässt sich die Korrelation zwischen den Snapshots nur erschwert herstellen. Vor diesem Hintergrund werden die vektoriellen Snapshots vor der Aufnahme in die Snapshotmatrix jeweils in eine Matrix überführt, die das Rechengebiet entsprechend der zugrunde liegenden Yee-Zellen abbildet. Durch diese Modifikation kann die Korrelation innerhalb eines Snapshots deutlich verbessert werden.

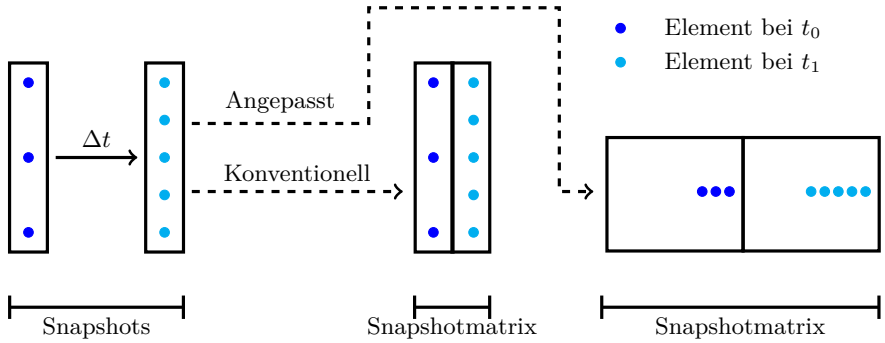


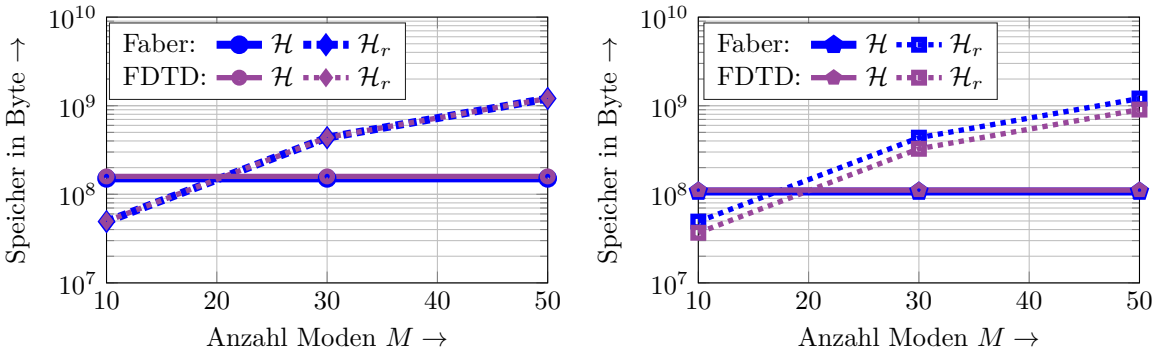
Abbildung 7.12: Erzeugung der Snapshotmatrix mit dem konventionellen und angepassten Ansatz. Die Korrelation zu einem Zeitpunkt wird durch den Abstand zwischen den Elementen illustriert.

Die Abbildung 7.12 veranschaulicht das Konzept der modifizierten Konstruktion der Snapshotmatrix. Der Nachteil der Modifizierung ist eine signifikante Verringerung der Kompressionsrate. Insbesondere ist für die Projektion in den Unterraum mittels der Galerkin-Projektion gemäß dem Abschnitt 7.2.3 die Anzahl der Spalten r der Matrix U von entscheidender Bedeutung, die durch die SVD bestimmt wird. Aufgrund der Modifikation gilt statt $\mathsf{m} \gg r$ lediglich $\mathsf{m} > r$. Außerdem sollte beachtet werden, dass beim Speichern einer dünnbesetzten Matrix in MATLAB die Anzahl der Nicht-Null-Elemente hinsichtlich der Komplexität die Matrixordnung dominiert. Aus diesem Grund kann der Speicherbedarf von $\mathcal{H}_r$ denjenigen von $\mathcal{H}$ überschreiten. Durch die Variation der Anzahl der Moden, die in dieser Untersuchung auf $M \in \{10, 30, 50\}$ beschränkt ist, kann dem entgegengewirkt werden. Damit der Einfluss der Modifikation umfassend beurteilt werden kann, wird neben der Rechenzeit und der Genauigkeit auch der Speicherbedarf untersucht. Darüber hinaus wird analysiert, inwiefern sich der Einsatz von MT auf einer GPU auf die genannten Parameter auswirkt.

7.5.1 Ergebnisse

Zunächst werden die Speicheranforderungen zur Durchführung der Studie untersucht, da grundsätzlich nur eine endliche Menge an Information verarbeitet werden kann. Angesichts der zunehmenden Größe der POD-Basis während der Auswertung mit dem FOM ist eine Reduktion des Speicherbedarfs sowohl der Systemmatrix $\mathcal{H}$ als auch der reduzierten Systemmatrix $\mathcal{H}_r$ unabdingbar.

In der Abbildung 7.13 kann der benötigte Speicher für $\mathcal{H}$ und $\mathcal{H}_r$ bei Anwendung der Faber- bzw. der FDTD-Methode nachvollzogen werden. Da bei der FDTD-Methode keine konventionelle Systemmatrix vorliegt, wird $\mathcal{H}$ der Lesbarkeit halber als Zusammensetzung aller benötigten Submatrizen betrachtet. Bei der Berechnung mit der CPU entsprechen die Verläufe denen in der Abbildung 7.13a. Für die Systemmatrix $\mathcal{H}$ besteht keine Abhängigkeit von der Anzahl der Moden M, dementsprechend bleiben die Speicheranforderungen konstant. Dennoch benötigt die Faber-Methode geringfügig weniger Speicher als die FDTD-Methode, obwohl bei letzterer die Untermatrizen zu $\mathcal{H}$ assembliert werden. Dieser Effekt ist auf die Speicherverwaltung dünnbesetzter Matrizen zurückzuführen, die von MATLAB im Hintergrund uneinsehbar gesteuert wird. Im Falle von $\mathcal{H}_r$ kann festgestellt werden, dass für $M \leq 20$ weniger Speicher erforderlich ist als bei $\mathcal{H}$. Für $M > 20$ ist die Anzahl der Moden mit Bedacht zu wählen. Bereits bei $M = 50$ führt die Reduktion aufgrund der in Abbildung 7.12 dargestellten Modifikation tatsächlich zu einem Anstieg um fast eine Größenordnung. Verglichen mit $\mathcal{H}$ liegen die Kurven bei $\mathcal{H}_r$ bei beiden Methoden gleichauf. Wird die GPU für die Auswertung ausgenutzt, kann sogar zusätzlich Speicher gespart werden, wie die Abbildung 7.13b darlegt. Diese Tatsache fällt besonders bei $\mathcal{H}_r$ auf, wo für das FDTD-Verfahren leicht erkennbar weniger Speicher als für die Faber-Methode nötig ist. Die Erklärung für dieses Verhalten liegt in der Art, wie auf die Einträge einer dünnbesetzten Matrix in MATLAB zugegriffen wird. Während auf der CPU üblicherweise das Compressed Sparse Column (CSC)-Format verwendet wird [207], führt dessen Anwendung auf der GPU aufgrund unregelmäßiger Speicherzugriffsmuster, Lastungleichgewichten und vermindertem Parallelismus zu Effizienzverlusten. Deshalb wurden in der jüngsten Vergangenheit Untersuchungen durchgeführt, um das Compressed Sparse Row (CSR)-Format für Berechnungen auf der GPU zu etablieren [246].



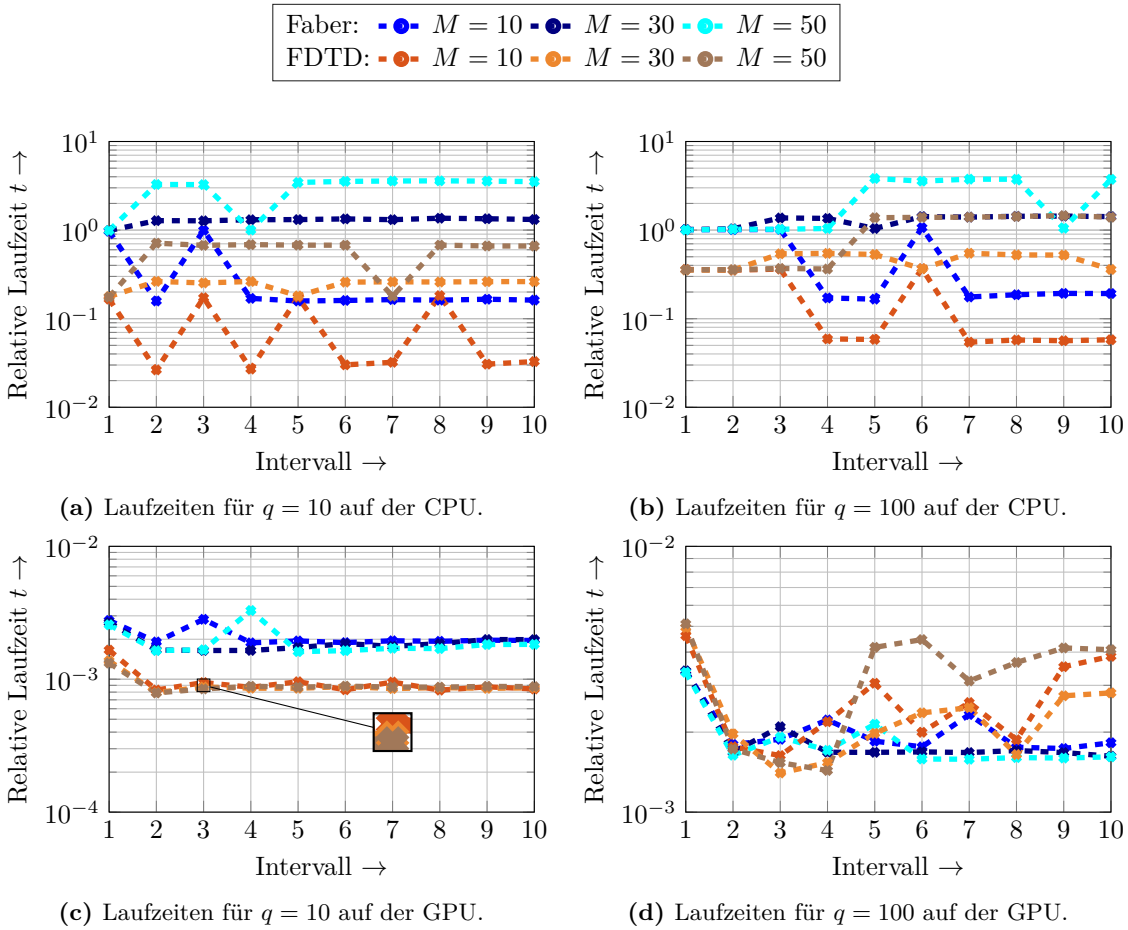
(a) Auswertung auf der CPU.

(b) Auswertung auf der GPU.

Abbildung 7.13: Speicherbedarf von $\mathcal{H}$ und $\mathcal{H}_r$ bei der Faber- und FDTD-Methode, wenn auf der CPU und GPU gerechnet wird.

Die Gegenüberstellung der Laufzeiten ist in der Abbildung 7.14 ersichtlich. Als Referenz wird dabei jeweils die Faber-Methode mit $q = 10$ und $q = 100$ sowie $c_{th} = 10^{-15}$ herangezogen, wobei für die zehn betrachteten Intervalle ausschließlich die Auswertung mittels des FOMs erfolgt. Von einer Referenz mit $q < 10$ ist abzusehen, da der Speicherbedarf durch die größere Anzahl der Snapshots exponentiell ansteigt. Die Resultate für die Auswertung auf der CPU sind in der Abbildung 7.14a und Abbildung 7.14b zu sehen. Der Einfluss der HPOD ist unabhängig von der Methode unmittelbar zu erkennen. Offensichtlich pendeln sich zwei Laufzeitniveaus ein. Naheliegenderweise lässt sich das untere Laufzeitniveau dem ROM und das obere dem

FOM zuordnen. Diese Annahme erweist sich jedoch als Trugschluss. Diesbezüglich gibt die Tabelle 7.5 einen Einblick darüber, ob das ROM zur Auswertung eines Intervalls ausreichte oder eine ergänzende Berechnung mit dem FOM erforderlich war. Zur besseren Übersicht werden das FOM mit F und das ROM mit R bezeichnet. Es zeigt sich, dass für $M = 10$ das untere Laufzeitniveau tatsächlich dem ROM und das obere dem FOM entspricht. Für $M \in \{30, 50\}$ verhält es sich allerdings umgekehrt. Die vorliegenden Resultate lassen sich durch die in der Abbildung 7.12 dargestellte Modifikation bei der Konstruktion der Snapshotmatrix erklären. Auch wenn die Modifikation auf den ersten Blick nachteilig erscheint, sei nochmals betont, dass ohne sie die Durchführung der Studie mit der HPOD nicht möglich wäre. Auch verweist die Abbildung 7.13 mit dem erhöhten Speicherbedarf auf die Vorkommnisse hin. Außerdem ist erwähnenswert, dass es erstmals mit einem verhältnismäßig geringem Zeitfaktor möglich ist mit der Faber-Methode eine schnelle Rechenzeit zu verwirklichen als mit dem FDTD-Verfahren. Dies zeigt sich deutlich, wenn bei der Faber-Methode lediglich $M = 10$ Moden berücksichtigt werden, während beim FDTD-Verfahren $M \in \{30, 50\}$ zur Anwendung kommen.



(a) Laufzeiten für $q = 10$ auf der CPU.

(b) Laufzeiten für $q = 100$ auf der CPU.

(c) Laufzeiten für $q = 10$ auf der GPU.

(d) Laufzeiten für $q = 100$ auf der GPU.

Abbildung 7.14: Relative Laufzeiten für die Auswertung von zehn Intervallen mit der Faber- und FDTD-Methode bei $M \in \{10, 30, 50\}$ und $q \in \{10, 100\}$ auf der CPU und GPU.

Wird die Berechnung auf der GPU ausgeführt, ergeben sich die Laufzeiten aus der Abbildung 7.14c und Abbildung 7.14d. Zunächst fällt auf, dass der Einsatz von MT bei $q = 10$ zu einer Glättung

Tabelle 7.5: Überblick der Intervalle und des eingesetzten Modells beim Anwendungsbeispiel.

			Intervall									
q	M	Methode	1	2	3	4	5	6	7	8	9	10
10	10	Faber	F	R	F	R	R	R	R	R	R	R
		FDTD	F	R	F	R	F	R	R	F	R	R
	30	Faber	F	R	R	R	R	R	R	R	R	R
		FDTD	F	R	R	R	F	R	R	R	R	R
	50	Faber	F	R	R	F	R	R	R	R	R	R
		FDTD	F	R	R	R	R	R	F	R	R	R
100	10	Faber	F	F	F	R	R	F	R	R	R	R
		FDTD	F	F	F	R	R	F	R	R	R	R
	30	Faber	F	F	R	R	F	R	R	R	R	R
		FDTD	F	F	R	R	R	F	R	R	R	F
	50	Faber	F	F	F	F	R	R	R	R	F	R
		FDTD	F	F	F	F	R	R	R	R	R	R

der Laufzeiten führt. Beim FDTD-Verfahren überlagern sich die Ergebnisse für alle Werte von M nahezu perfekt, sodass keine nennenswerten Unterschiede erkennbar sind. Konträr dazu kann bei der Faber-Methode die Auswertung mit dem FOM deutlich wahrgenommen werden. Weiterhin konnte die Laufzeit durch die Anwendung der GPU signifikant reduziert werden, was ebenso den *Offset* zwischen den Methoden betrifft. Die Erhöhung des Zeitfaktors auf $q = 100$ führt zu einigen beachtlichen Ergebnissen. Im Vergleich zu $q = 10$ reagiert nun das FDTD-Verfahren wesentlich stärker auf das Modell als die Faber-Methode. Zusätzlich zeigt die Gegenüberstellung mit den Auswertungen auf der CPU, dass sich keine klaren Laufzeitniveaus abzeichnen. Dies wird mit der zeitlichen Schwankungen der Performance der GPU begründet. Ohne die Tabelle 7.5 könnte keine Aussage über das angewandte Modell in einem Intervall getroffen werden. Darüber hinaus fällt der Rechenzeitgewinn geringer aus, je höher q gewählt wird. In diesem Kontext sollte zudem auf die Berechnung des ersten Intervalls aufmerksam gemacht werden. Diese wird im Hintergrund von MATLAB bzw. CUDA im Laufe der Initialisierung anders gehandhabt als die verbleibenden Intervalle.

Zur Überprüfung der Genauigkeit der HPOD werden zwei Untersuchungen durchgeführt, wobei als Referenz äquivalent zum Abschnitt 7.3 die Faber-Methode mit höchster Maschinengenauigkeit dient. Dafür werden alle zehn Intervalle mit dem FOM evaluiert. Die erste Teiluntersuchung widmet sich dem Vergleich der Koppellängen, die mittels der Faber- und FDTD-Methode ermittelt werden. Im Kontext der MMI-Koppler kann die Überkopplung auch als *Single Image* bezeichnet werden, da sie seine Rekonstruktion des Eingangsprofils der angeregten Grundmoden in einem Wellenleiter darstellt, der ursprünglich nicht angeregt wurde [244, 247]. Wie bereits erwähnt, werden 300 ausbreitungsfähige Grundmoden im oberen Wellenleiter angeregt. Die hohe

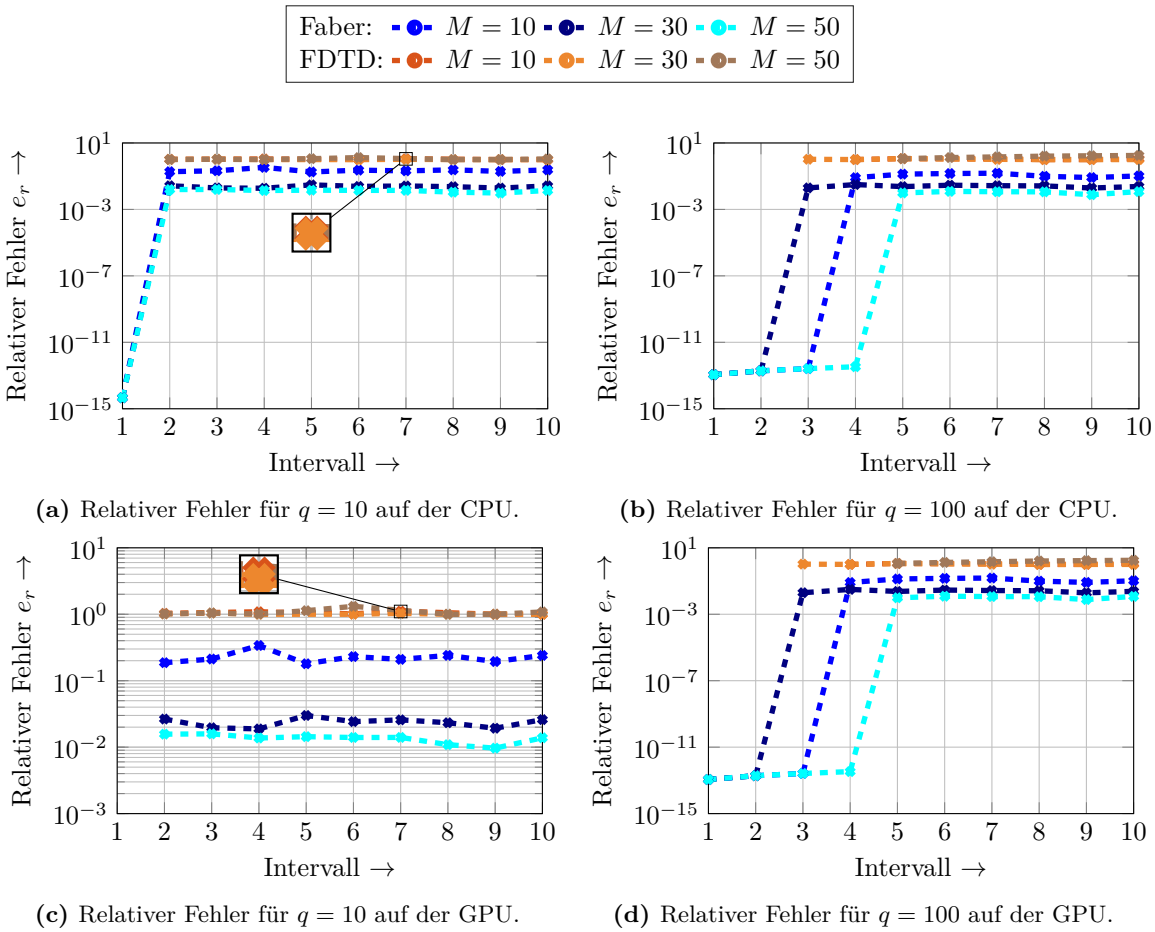
Tabelle 7.6: Berechnete Koppellängen in Abhängigkeit von q und M beim Anwendungsbeispiel.

q	M	Methode	$\mathcal{L}$ in µm	e_r
10	10	**Faber**	323,2	0,012
		FDTD	210,2	0,357
	30	**Faber**	325,6	0,005
		FDTD	210,2	0,357
	50	**Faber**	327,1	0
		FDTD	210,2	0,357
100	10	**Faber**	328,7	0,005
		FDTD	298,2	0,088
	30	**Faber**	325,6	0,005
		FDTD	226,4	0,308
	50	**Faber**	333,6	0,020
		FDTD	332,6	0,017

Anzahl an Moden ist darin begründet, dass während der Studie von [PS8] beobachtet werden konnte, dass bei einer zu geringen Modenzahl ein Verblassen dieser Moden mit fortschreitender Extrapolation im Rahmen des ROMs auftritt. Gleichzeitig erschwert die große Anzahl an Moden die analytische Bestimmung einer exakten Wellenlänge erheblich, sodass die Referenz numerisch bestimmt wird und sich demnach eine Koppellänge von $\mathcal{L}_{ref} = 327{,}1\,\mu\text{m}$ einstellt. Dem stehen die approximierten Koppellängen aus der Berechnung mit der HPOD in der Tabelle 7.6 gegenüber. Für $q = 10$ resultiert beim FDTD-Verfahren konstant dieselbe Koppellänge, trotz der Erhöhung von M. Im Gegensatz dazu kann der relative Fehler e_r bei der Faber-Methode durch die Hinzunahme mehrerer Moden bedeutend gesenkt werden. Bei $M = 50$ beträgt dieser sogar $e_r = 0$. Für den Fall $q = 100$ erreicht keine der definierten Konstellation die Referenzkoppellänge. Dafür kann jedoch die unerwartete Beobachtung gemacht werden, dass mit steigender Modenzahl in einigen Fällen eine Fehlerzunahme auftritt. Als Ursache für dieses Phänomen kann die Verteilung der Moden in dem System angeführt werden. Demnach ist in dieser Studie nicht unbedingt die Quantität der Moden ausschlaggebend, sondern deren Qualität. Es sollte nicht vergessen werden, dass sowohl die hohe Zeitschrittweite mit $q = 100$ als auch die hohe Anzahl der extrapolierten Intervalle mit $\Delta t = 800\Delta t_{CFL}$ eine Herausforderung für die Genauigkeit der Approximation darstellen.

Die zweite Teiluntersuchung im Rahmen der Genauigkeit erfolgt analog zum Abschnitt 7.3. Gemäß der Gleichung (7.13) wird am Ende jedes Intervalls das Verhältnis zwischen der approximierten Lösung und der Referenzlösung für die E_y-Komponente gebildet und als zeitabhängiger relativer Fehler e_r festgehalten. Die Abbildung 7.15 stellt die Ergebnisse dar. Die Abbildung 7.15a und Abbildung 7.15b resultieren aus der Auswertung auf der CPU, während die Abbildung 7.15c und Abbildung 7.15d die Ergebnisse der Berechnung auf der GPU zeigen. Weil das erste Intervall

immer mit dem FOM zum Trainieren des Systems evaluiert wird, ist kein relativer Fehler zu erwarten. Da Rundungsfehler bei einer Berechnung auf einem PC mit endlicher Rechengenauigkeit jedoch nie vollständig ausgeschlossen werden können, kann bereits beim ersten Intervall ein relativer Fehler der Ordnung $e_r \approx 10^{-15}$ auftreten. Dieser Umstand ist für die Teiluntersuchung unerheblich und wird daher vernachlässigt. Dagegen ist die Tatsache positiv hervorzuheben, dass keine starken Schwankungen in Bezug auf e_r zu erkennen sind. Trotz der Berechnung einiger Intervalle entweder ausschließlich mit dem ROM oder mit der Korrektur durch das FOM bleibt die Fehlerordnung in allen Konstellationen konstant. Sollte der Fehler von einem zum anderen Intervall etwas ansteigen, ist die HPOD dennoch in der Lage den Fehler wieder zu minimieren. Dies ist beispielsweise in der Abbildung 7.15c bei der Faber-Methode für $M = 10$ ersichtlich. Es besteht die Möglichkeit, dass die Genauigkeit weiter erhöht werden könnte, wenn die Anzahl der Zeitschritte pro Intervall gesenkt wird. Dennoch ist aufgrund des Speicherbedarfs diese Option mit Vorsicht zu betrachten. Des Weiteren muss betont werden, dass die Fehlerordnung nicht allein durch die HPOD bestimmt wird. Die Anwendung der HPOD dient primär der Extrapolation des zeitabhängigen Propagators mit dem Ziel, die Laufzeit zu reduzieren, während die Fehlerrate idealerweise konstant bleibt.



(a) Relativer Fehler für $q = 10$ auf der CPU.

(b) Relativer Fehler für $q = 100$ auf der CPU.

(c) Relativer Fehler für $q = 10$ auf der GPU.

(d) Relativer Fehler für $q = 100$ auf der GPU.

Abbildung 7.15: Relativer Fehler für die Auswertung von zehn Intervallen mit der Faber- und FDTD-Methode bei $M \in \{10, 30, 50\}$ und $q \in \{10, 100\}$ auf der CPU und GPU.

7.6 Zusammenfassung und Zwischenfazit

Das Hauptaugenmerk dieser Untersuchung galt der Extrapolation des zeitabhängigen Propagators zur Beschleunigung der Rechenzeit. Zu diesem Zweck wurde die POD als Vertreter der MOR-Methoden herangezogen und ihre Kompatibilität mit der Faber-Methode analysiert. Da die Ordnung der Systemmatrix maßgeblich für die Komplexität der Approximation ist, bestand die Intention darin, diese mit Hilfe der POD zu reduzieren. Zur Umsetzung dieses Vorhabens wurde das System zunächst im Rahmen der Offline-Phase mit dem FOM trainiert, indem Snapshots des Zustandsvektors zu festgelegten Zeitpunkten in einer Snapshotmatrix gespeichert wurden. Nach Abschluss dieses Prozesses erfolgte eine SVD der Snapshotmatrix. Die daraus resultierenden Matrizen dienten nachfolgend zur Projektion in einen reduzierten Unterraum. In der anschließenden Online-Phase war es möglich, die Propagation der elektromagnetischen Welle mit dem ROM zu extrapolieren.

Der beschriebene Ablauf entspricht dem der konventionellen POD. Wurde die Transition vom FOM zum ROM bei Erfüllung des vordefinierten Wechselkriteriums einmal vollzogen, war eine fehlerfreie Rückkehr zum FOM nicht mehr möglich. Aus diesem Grund wurde mittels einer Parameterstudie erforscht, unter welchen Bedingungen eine geringe Fehlerrate erzielt werden kann. Dabei zeigte sich, dass sowohl die Anzahl der Moden als auch die verwendete Genauigkeit ab einer bestimmten Schwelle zu einer Sättigung der Fehlerrate führen. Aufgrund der inhärenten Statik der POD wurde in einem nächsten Schritt die HPOD als dynamische Erweiterung untersucht. Die HPOD unterscheidet sich von der POD durch zwei zentrale Merkmale. Auf der einen Seite wird anstelle der SVD die ISVD auf die Snapshotmatrix angewandt, was eine effizientere Berechnung der Singulärwerte erlaubt. Zum anderen ermöglicht die HPOD im laufenden Betrieb einen dynamischen Wechsel zwischen dem FOM und ROM, wodurch ein gezielter Kompromiss zwischen der Genauigkeit und Rechenzeit realisierbar wird. Für den Wechsel zwischen den Modellen mussten neue Kriterien formuliert werden, da die bestehenden aus vorhergehenden Studien nicht geeignet waren. Durch eine geringfügige Modifikation wurde ein neues Kriterium entwickelt, das anschließend validiert wurde. Weiterhin wurde mit dem neuen Kriterium der Ansatz verfolgt, jedes Intervall zunächst mit dem ROM zu berechnen und bei Überschreiten eines festgelegten Fehlers mit dem FOM zu korrigieren. Dies wurde damit gerechtfertigt, dass die kombinierte Rechenzeit für die Projektion und Approximation beim ROM geringer ist als die reine Expansionszeit des FOMs.

Nach der Validierung des neuen Wechselkriteriums im theoretischen Rechengebiet wurde auf einen praktischen 2×2 MMI-Koppler zwischen zwei symmetrischen Wellenleitern umgestellt. Dies erforderte eine Modifikation der Konstruktion der Snapshotmatrix, damit die Auswertung auch mit dem ROM möglich war. Jene Modifikation ging jedoch mit einem erhöhten Speicheraufwand einher, der dennoch akzeptabel war. Zudem wurde der potenzielle Mehrwert durch den Einsatz einer GPU im Hinblick auf den Speicherbedarf und die Rechenzeit betrachtet. Während der Speicherbedarf durch nicht einsehbare Hintergrundprozesse der GPU in MATLAB geringfügig reduziert werden konnte, führte deren Einsatz zu einem Rechenzeitgewinn von mehr als einer Größenordnung. Zur Überprüfung der Genauigkeit der HPOD wurden einerseits die Koppellänge und andererseits der relative Fehler als Indikatoren für die Genauigkeit verwendet. Beide Analysen ergaben, dass trotz der hohen Anzahl extrapolierter Zeitschritte und einer großen Zeitschrittweite die HPOD in Kombination mit der Faber-Methode einen tolerierbaren Fehler

liefert. Alles in allem konnte das Potenzial, die HPOD mit der Faber-Methode zu verknüpfen, hinsichtlich der Laufzeit erfolgreich demonstriert werden.

Kapitel 8

Zusammenfassung

Das Ziel der vorliegenden Arbeit war es, die Approximation der Maxwell-Gleichungen im Zeitbereich für Anwendungen in der THz-Technik und Photonik zu optimieren. Auf Grundlage etablierter numerischer Verfahren wurde die Weiterentwicklung dieser Ansätze im Hinblick auf eine potenzielle Effizienzsteigerung analysiert. Zur Beurteilung der generierten Resultate wurden dabei die Laufzeit, Genauigkeit und Komplexität sowie der Speicherbedarf als zentrale Kriterien herangezogen.

Bevor die praktische Validierung erfolgte, wurde zunächst auf die wesentlichen Grundlagen Bezug genommen. In Kapitel 2 lag der Schwerpunkt auf der klassischen elektromagnetischen Feldtheorie nach Maxwell. Ausgehend davon wurden die im weiteren Verlauf verwendeten Materialmodelle hergeleitet. Aufgrund der realistischen Darstellung der Materialantwort fiel die Wahl im linearen Fall auf das Drude- und Lorentz-Modell. Analog dazu wurde für den nichtlinearen Fall das Zwei- bzw. Mehr-Niveau-Modell gewählt.

Um die Modellierung der Physik für die simulative Untersuchungen zu ermöglichen, widmete sich das Kapitel 3 den numerischen Grundlagen. Wegen ihrer weiten Verbreitung, der simplen Implementierung und der expliziten Berechnung lag der Fokus auf der Yee-Methode. Insbesondere wurde die örtliche Diskretisierung im Rahmen dieser Arbeit ausschließlich mit der Yee-Methode durchgeführt. Im Gegensatz dazu wurde auf die zeitliche Diskretisierung mit jener Methode verzichtet. Diese Entscheidung ergab sich primär daraus, dass die Zeitschrittweite an die CFL-Bedingung gekoppelt ist. Dementsprechend wurden alternative numerische Ansätze mit ihren Stärken und Schwächen vorgestellt. Des Weiteren wurde die Operatornotation eingeführt, welche die relevanten Feld- und Systemgrößen in einer kompakten Schreibweise zusammenfasst. Bezogen auf die nachfolgend eingeführten Randbedingungen ermöglicht die Operatornotation mit der Systemmatrix und dem Zustandsvektor eine vereinfachte Darstellung. Als Vertreter der reflektierenden Randbedingungen wurden die PEC und PMC thematisiert, während bei den dämpfenden Randbedingungen die ABC und PML im Mittelpunkt standen.

Da die Lösung der Maxwell-Gleichungen aufgrund der expliziten Formulierung mit einem Matrixexponential-Ansatz erfolgte, war nach der örtlichen Diskretisierung die Approximation des zeitabhängigen Propagators erforderlich. Vor dem Hintergrund, eine Approximation mit einer hohen Genauigkeit und außergewöhnlich großen Zeitschrittweite zu realisieren, wurde bewusst die Faber-Methode ausgewählt. Demnach wurden die relevanten theoretischen Grundlagen der Faber-Methode in Kapitel 4 behandelt. Damit die Faber-Methode effizient angewandt werden konnte, war eine möglichst exakte Abschätzung des Eigenwertspektrums der zugrunde liegenden Systemmatrix notwendig. Je präziser das Spektrum durch die charakteristische Jordankurve eingegrenzt wird, desto höhere Zeitschrittweiten können verwendet werden. Hierbei bestand die Herausforderung darin, die Eigenwerte ohne Kenntnis über ihre Verteilung zu ermitteln. Dazu wurden im ersten Schritt grundlegende bekannte Eigenschaften aus der Numerik gezielt genutzt.

So treten beispielsweise bei einem System ohne Verstärkung oder Dämpfung ausschließlich rein imaginäre Eigenwerte auf, welche sich analytisch bestimmen ließen. Außerdem treten sie stets in Form von komplex konjugierten Paaren auf. Im gedämpften Fall ließ sich der kleinste Realteil in der linken Halbebene des Eigenwertraums mit Hilfe der k-Raum-Methode aus einer vorausgegangenen Untersuchung abschätzen. Mit diesem Wissen konnte zunächst ein Rechteck anhand der Eckpunkte zur Eingrenzung der Eigenwerte festgelegt werden. Darauf basierend konnte anschließend eine Ellipse als Kontur konstruiert werden, auf die eine analytische Abbildungsvorschrift anwendbar war. Somit konnten die Faberpolynome mit den korrespondierenden Entwicklungskoeffizienten für die Approximation ausgewertet werden. Damit die Faber-Methode numerisch stabil bleibt, müssen einerseits alle Eigenwerte eingeschlossen sein und andererseits die logarithmische Kapazität durch eine Skalierung auf einen Maximalwert von Eins normiert werden. Weiterhin wurde gezeigt, dass das direkte Einbinden des Zustandsvektors elementar für die Approximation ist. Nur durch diese Maßnahme lässt sich der fehlerfreie Einsatz der Faber-Methode bei komplexeren Systemen gewährleisten, andernfalls wird der Speicherbedarf der Systemmatrix zum unüberwindbaren Hindernis.

Bisher wurde lediglich die Umschließung der Eigenwerte durch eine ellipsenförmige Kontur betrachtet. Zwar bieten sich auch andere Konturen an, jedoch ist die dafür benötigte Abbildungsfunktion in den seltensten Fällen bekannt bzw. analytisch zugänglich. Nach aktuellem Kenntnisstand lagen neben der Ellipse nur für den Einheitskreis sowie für quadratische Konturen mit abgerundeten Kanten analytische Abbildungsvorschriften vor. Aus diesem Grund wurde in Kapitel 5 das Ziel verfolgt eine noch engere Kontur als die Ellipse numerisch herzuleiten, um die Laufzeit weiter zu verringern. Für das Vorhaben erwies sich die SC-Transformation als besonders geeignet, da sie eine Abbildung auf Polygone erlaubt. Dennoch mussten zur Anwendung der SC-Transformation sowohl das inhärente Parameterproblem als auch das Crowding-Phänomen berücksichtigt werden, um auf die gewünschte Kontur korrekt abzubilden. Für die numerische Evaluation bot sich das Drude-Modell an, weil es eine simple Modellierung gedämpfter Systeme zulässt. Die analytisch bestimmbaren Konturen wurden mit drei- bis siebeneckigen Polygonen verglichen. Ferner wurde bereits vor der Hauptuntersuchung darauf hingewiesen, die Anzahl der Koeffizienten für die Abbildung im Sinne der Komplexität auf ein Minimum zu reduzieren. Daher kamen Polygone zum Einsatz, welche mit maximal fünf Koeffizienten entwickelt wurden. Bezüglich der Konvergenz der Entwicklungskoeffizienten einer Kontur konnte demonstriert werden, dass Polygone eine geringere Entwicklungsordnung erfordern als die analytisch ermittelten Vertreter, um den vorgegebenen Schwellenwert zu erreichen. Dies hatte jedoch keinen signifikanten Einfluss auf die Laufzeit. Entgegen der Erwartung konnte die schnellste Laufzeit beim Einheitskreis beobachtet werden. Dies führte zur Schlussfolgerung, dass nicht die Form der Kontur für die Rechenzeit ausschlaggebend ist, sondern die Anzahl der zur Entwicklung erforderlichen Koeffizienten. Für die Polygone ergab sich ein unlösbarer Konflikt. Auf der einen Seite kann eine engere Kontur die Konvergenz beschleunigen, was jedoch mit einer steigenden Anzahl an Koeffizienten einhergeht. Auf der anderen Seite ist es prinzipiell nicht möglich für die Entwicklung weniger Koeffizienten als beim Einheitskreis zu verwenden, da sich dieser bereits mit einem einzigen Koeffizienten beschreiben lässt.

In den vorausgegangenen Untersuchungen wurde die Simulation der elektromagnetischen Wellenausbreitung generell in einem homogen diskretisierten Rechengebiet durchgeführt. Auch wenn dies die Implementierung durch den Einsatz eines einzigen globalen Operators zu vereinfachen mag, lässt sich die Komplexität der Approximation über Dekompositions- und *Splitting*-Ansätze

deutlich senken. Infolgedessen wurde in Kapitel 6 der neue Ansatz der lokalen Operatoren erforscht, um eine separate Auswertung des zeitabhängigen Propagators in unterschiedlich diskretisierten Subgebieten zu verwirklichen. Das Konzept sah eine Verteilung der Rechenlast vor, wodurch die Rechenzeit und der Speicherbedarf reduziert werden können. Bei der Rechenzeit wurde die Tatsache ausgenutzt, dass die Genauigkeit der Approximation über die Entwicklungskoeffizienten der Faber-Methode definiert ist. Folglich wurde der lokale Operator in einem grob aufgelösten Subgebiet mit einer niedrigeren Entwicklungsordnung ausgewertet als in einem fein aufgelösten Subgebiet, ohne die Genauigkeit der Approximation zu beeinträchtigen. Der Prozess wurde in die drei Schritte Expansion, Extraktion und Assemblierung unterteilt, wobei die Extraktion und Assemblierung als *Overhead* zusammengefasst wurden. Solange die Verteilung der maximalen Eigenwerte des am feinsten aufgelösten Subgebiets präzise approximiert werden konnten, galt der Ansatz der lokalen Operatoren als stabil. Zudem sind die lokalen Operatoren systemunabhängig und flexibel anwendbar. Zur Bewertung der lokalen Operatoren wurde zunächst die theoretische Komplexität hergeleitet und im Anschluss mit den Ergebnissen praktischer Simulationen in einem linearen und einem nichtlinearen System gegenübergestellt. Im Blickpunkt der Analyse stand besonders die Implementierung. Dabei sollte im Voraus geklärt werden, ob die Expansion der Subzustandsmatrizen dynamisch in Abhängigkeit von der Entwicklungsordnung oder unabhängig davon statisch erfolgen sollte. Während sich der dynamische Ansatz theoretisch durch eine reduzierte Komplexität als effizienter erwies, wurde der statische Ansatz in der praktischen Umsetzung wesentlich durch die Expansion limitiert, was im Durchschnitt zu einer vierfach längeren Laufzeit führte. Programmiertechnisch ließ sich der *Overhead* nur durch die Erhöhung der Zeitschrittweite reduzieren. Um dem verbleibenden *Overhead* entgegenzusteuern, wurde ergänzend auf der Hardwareebene der Einsatz von MP auf der CPU sowie MT auf der GPU evaluiert. Im linearen System wurde der Einfluss der Programmiersprache analysiert, wobei kein nennenswerter Unterschied zwischen MATLAB und C festgestellt wurde. Im nichtlinearen System wurde ein *Operatorsplitting* implementiert, das im vorgegebenen System nur marginal zur Laufzeitreduktion beitrug. Insgesamt konnten die lokalen Operatoren in Relation zum globalen Operator sowohl im nichtlinearen System als auch bei der Parallelisierung mittels CPU eine schnellere Rechenzeit erzielen.

Die hohen Speicheranforderungen der Faber-Methode sind in erster Linie auf die Systemmatrix zurückzuführen, obwohl diese aufgrund der Ortsdiskretisierung gemäß Yee dünnbesetzt ist. Die Einbindung von Materialmodellen und Grenzschichten erschwert diesen Umstand. Demzufolge wurde in Kapitel 7 die Thematik der MOR aufgegriffen. Die Intention war, die Komplexität der Systemmatrix durch die Projektion in einen Unterraum um mehrere Größenordnungen zu senken, ohne dabei die relevanten Informationen der propagierenden elektromagnetischen Welle zu verlieren. Zu diesem Zweck wurde die Vereinbarkeit der POD- mit der Faber-Methode geprüft und durch eine numerische Evaluation belegt. Dennoch erfüllte die Kombination der Methoden die Anforderungen an die Approximation nicht. Zum einen wurde anhand einer Parameterstudie ein optimaler Parametersatz angestrebt, dessen Resultate jedoch keineswegs allgemeingültig sind. Zum anderen ließ sich der Übergang von der Trainings- in die Extrapolationsphase respektive von der Offline- in die Online-Phase nicht ohne eine Fehlerzunahme revidieren. In Anbetracht dessen wurde mit den Erkenntnissen aus der POD- auf die HPOD-Methode umgestellt, die einen dynamischen Wechsel zwischen den Phasen vorsieht. Hierbei rückte speziell das Wechselkriterium in den Vordergrund, da sich jenes des konventionellen POD-Verfahrens in Bezug auf die Genauigkeit als unzureichend erwies. Dementsprechend wurde ein abweichendes Wechselkriterium auf Basis der tatsächlichen und erwarteten Energien im System definiert.

Diesbezüglich wurde anstelle zweier separater Wechselkriterien zwischen den Phasen ein einziges Kriterium bei der HPOD-Methode implementiert. Die Idee bestand darin, jedes Intervall anfänglich mit dem ROM zu berechnen und erst bei Erfüllung des Wechselkriteriums auf das FOM umzusteigen. Diese Vorgehensweise wurde damit gerechtfertigt, dass die Rechenzeit für die Auswertung eines Intervalls mit dem ROM mehrere Größenordnungen unter der für die notwendige SVD und Projektion lag. Im Rahmen einer weiteren numerischen Evaluation konnten überzeugende Resultate erzielt werden, sodass die HPOD-Methode und das Wechselkriterium als erfolgreich angesehen werden können. Zum Abschluss wurden die Laufzeiten der HPOD-Methode auf der CPU und GPU verglichen. Nach wie vor war die Hardware der Programmierung überlegen, jedoch ermöglichte die HPOD-Methode unter bestimmten Bedingungen eine Laufzeitreduzierung um nahezu eine Größenordnung.

8.1 Fazit

Die Anwendung der SC-Transformation zur Anpassung der Kontur bietet im Kontext der Faberpolynome keinen erkennbaren Vorteil. Zwar kann sie die Konvergenz der Entwicklungskoeffizienten beschleunigen, allerdings geschieht dies auf Kosten einer erhöhten Laufzeit pro Rekursionsschritt. Sofern das Eigenwertspektrum in seinen Extrempunkten bekannt ist, kann eine einfache Abbildung auf den Einheitskreis bereits genügen. Auch wenn die Untersuchung auf ein verlustbehaftetes System beschränkt war, kann das Ergebnis jedoch als allgemeingültig für unterschiedliche Systemarten angesehen werden.

Das Fazit zu den lokalen Operatoren fällt grundsätzlich positiv aus, bedarf jedoch einer differenzierten Betrachtung. Bei linearen Systemen wirkt der *Overhead* als limitierender Faktor, der durch die verteilte Rechenlast nicht kompensiert werden kann. Werden die lokalen Operatoren hingegen in einem nichtlinearen System eingesetzt, kann insbesondere die Rechenzeit für die Nichtlinearität im Verhältnis zum *Overhead* stärker reduziert werden. Dieser Effekt verstärkt sich mit zunehmender Komplexität. Während sich der *Overhead* softwareseitig lediglich durch eine Erhöhung der Zeitschrittweite verringern lässt, kann er hardwareseitig zusätzlich durch den Einsatz von MP auf der CPU gesenkt werden. Darüber hinaus ist im Zuge der inhärenten Expansion lokaler Operatoren der Systemdimensionierung besondere Aufmerksamkeit zu widmen, da diese im Vergleich zu globalen Operatoren die Entwicklung stärker einschränken. Die Abhängigkeit besteht dabei nicht nur von den Entwicklungskoeffizienten, sondern auch von der Anzahl der Diskretisierungspunkte, wodurch die maximal zulässige Zeitschrittweite weiter begrenzt wird.

Die POD-Methode ist aufgrund ihrer fehlenden Flexibilität zwischen der Offline- und Online-Phase nur von begrenztem Wert für die Approximation mittels der Faber-Methode. Dahingegen stellt die HPOD-Methode einen vielversprechenden Ansatz dar, der aber eine sorgfältige Abstimmung der Simulationsparameter voraussetzt, um das volle Potenzial zu entfalten. Trotz einer durchgeführten Parameterstudie ließ sich kein universelles Parametersatz ermitteln. Das Konzept, ein Intervall generell mit dem ROM zu berechnen und bei Bedarf mit Hilfe des FOMs nachzubessern, führte zu einer deutlich robusteren Berechnung. Hierbei spielte das neu definierte Wechselkriterium eine zentrale Rolle, da es die Energien im System anders bewertet als bisher. Außerdem war eine alternative Implementierung zur Erzeugung der Snapshot-Matrix erforderlich,

da die POD-Moden mit dem größten Beitrag mit der klassischen Implementierung sonst keine ausreichende Korrelation aufwiesen und somit die Extrapolation signifikant erschwert wurde.

Trotz aller softwareseitigen Optimierungen bleibt die Leistungsfähigkeit der Hardware der ausschlaggebende Faktor. Da die Faber-Methode hauptsächlich auf MVMs basiert, kann die Rechenzeit allein durch das Verschieben der Rechenlast auf die GPU um mehrere Größenordnungen reduziert werden. Selbst unter Einsatz der HPOD-Methode, die in dieser Arbeit den größten Rechenzeitgewinn ermöglichte, blieb die Diskrepanz zu den mit MT erzielten Laufzeiten erheblich. Angesichts der stetig wachsenden Anzahl an CUDA-Kernen in den GPUs von Nvidia ist künftig mit einer noch größeren Abweichung zu rechnen. Abschließend lässt sich festhalten, dass die Parallelisierung in MATLAB mittlerweile auch ohne tiefgehende Vorkenntnisse effizient umgesetzt werden kann unter der Annahme, dass die entsprechenden *Toolboxen* und Lizenzen zur Verfügung stehen.

8.2 Ausblick

Ungeachtet der vermeintlich unschlagbaren Effizienz der GPU können in der Zukunft weitere ergänzende Untersuchungen getätigt werden. Im Rahmen dieser Arbeit wurden die lokalen Operatoren ausschließlich in Rechengebieten eingesetzt, welche nach Yee örtlich diskretisiert wurden. Beispielsweise könnte bei der FEM und FVM das Aufteilen der Rechenlast der inhärenten hohen Komplexität dieser Verfahren entgegenwirken. Des Weiteren zeichnete sich schon beim Zwei-Niveau-Modell ein offensichtlicher Rechenzeitgewinn ab. Durch die Überführung in das Mehr-Niveau-Modell respektive durch das Einfügen von mehr Nichtlinearität wird die Approximation voraussichtlich schneller erfolgen. Die HPOD-Methode zeigte in Ansätzen ihren Mehrwert in Kombination mit der Faber-Methode. Trotzdem gilt es zu klären, ob ein ideales Parametersatz existiert oder wie sich ein solches mit geringem Aufwand situationsabhängig bestimmen lässt. Außerdem kann das neue Wechselkriterium durch weitere Anwendungen validiert und im besten Fall eine einheitliche Implementierung ableitet werden. Darüber hinaus könnte die Kombination der lokalen Operatoren mit der HPOD erforscht werden.

Unabhängig von den numerischen Methoden soll MATLAB R2025a die Berechnung von dünnbesetzten Matrizen mit dem Datentyp *single* gestatten. Dadurch lässt sich der Speicherbedarf der Systemmatrix mühelos halbieren. Zudem ergeben sich für die Berechnung mit der GPU zwei wegweisende Vorteile. Auf der einen Seite können hierdurch, trotz der heutzutage begrenzten VRAM-Kapazitäten, deutlich komplexere Systeme evaluiert werden. Auf der anderen Seite berichtet Nvidia, dass der Datentyp *single* den Durchsatz an FLOPS bei CUDA signifikant steigern kann.

Abschließend steht noch die praktische Validierung der simulierten Ergebnisse aus. Obwohl bei den Untersuchungen eine hohe numerische Genauigkeit zugrunde gelegt wurde, müssen die Resultate mit denen aus der experimentellen Physik abgeglichen werden. Im Idealfall könnte eine *Toolbox* entwickelt werden, die vor der eigentlichen Analyse aufgerufen und ausgeführt wird.

Publikationsverzeichnis

[PS1] W. Plotnikov, B. L. Inci und D. Schulz, „Investigation of conformal Mappings for the Approximation of Faber Polynomial based Propagators," in *2022 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization (NEMO)*, IEEE, 2022, S. 1–3. Adresse: https://doi.org/10.1109/NEMO51452.2022.10038962

[PS2] W. Plotnikov, B. L. Inci und D. Schulz, „Evaluation of the Faber polynomial based Expansion for different Contours in a lossy System," in *2025 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization (NEMO)*, IEEE, 2025, S. 1–3. Adresse: https://doi.org/10.1109/NEMO62710.2025.11215553

[PS3] W. Plotnikov, T. Murawski und D. Schulz, „Local Propagators utilizing Faber Polynomial based Expansions," in *2022 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization (NEMO)*, IEEE, 2022, S. 1–3. Adresse: https://doi.org/10.1109/NEMO51452.2022.10038518

[PS4] W. Plotnikov und D. Schulz, „Performance Analysis of Faber Polynomial based local Propagators for Photonics," *Optical and Quantum Electronics*, Jg. 57, Nr. 3, S. 170, 2025. Adresse: https://doi.org/10.1007/s11082-025-08072-9

[PS5] W. Plotnikov und D. Schulz, „Optimization of the Faber Polynomial based Propagator through Parallelization," in *2024 49th International Conference on Infrared, Millimeter, and Terahertz Waves (IRMMW-THz)*, IEEE, 2024, S. 1–2. Adresse: https://doi.org/10.1109/IRMMW-THz60956.2024.10697858

[PS6] W. Plotnikov, R. Pommer und D. Schulz, „Faber Polynomial based Local Propagators for Laser Applications," in *2024 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization (NEMO)*, IEEE, 2024, S. 1–3. Adresse: https://doi.org/10.1109/NEMO59437.2024.11395755

[PS7] W. Plotnikov und D. Schulz, „Parallelization of Faber Polynomial Based Propagators for Laser Applications," *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, Jg. 38, Nr. 4, e70091, 2025. Adresse: https://doi.org/10.1002/jnm.70091

[PS8] W. Plotnikov, B. L. Inci und D. Schulz, „Optimization of Faber Polynomial based Propagators via Proper Orthogonal Decomposition," *Communications on Applied Mathematics and Computation*, S. 1–27, 2026. Adresse: https://doi.org/10.1007/s42967-026-00573-y

Literaturverzeichnis

[1] A. Bühl, *Die virtuelle Gesellschaft des 21. Jahrhunderts: sozialer Wandel im digitalen Zeitalter*. Springer-Verlag, 2013.

[2] A. Y. Pawar, D. D. Sonawane, K. B. Erande und D. V. Derle, „Terahertz technology and its applications," *Drug invention today*, Jg. 5, Nr. 2, S. 157–163, 2013.

[3] Z. Chen u. a., „Terahertz wireless communications for 2030 and beyond: A cutting-edge frontier," *IEEE Communications Magazine*, Jg. 59, Nr. 11, S. 66–72, 2021.

[4] J. C. Maxwell, „VIII. A dynamical theory of the electromagnetic field," *Philosophical transactions of the Royal Society of London*, Nr. 155, S. 459–512, 1865.

[5] A. J. Schwab, *Begriffswelt der Feldtheorie: Elektromagnetische Felder, Maxwell-Gleichungen, Gradient, Rotation, Divergenz*. Springer-Verlag, 2019.

[6] R. W. Boyd, A. L. Gaeta und E. Giese, „Nonlinear optics," in *Springer Handbook of Atomic, Molecular, and Optical Physics*, Springer, 2008, S. 1097–1110.

[7] J. Hebling, K.-L. Yeh, M. C. Hoffmann und K. A. Nelson, „High-power THz generation, THz nonlinear optics, and THz nonlinear spectroscopy," *IEEE Journal of Selected Topics in Quantum Electronics*, Jg. 14, Nr. 2, S. 345–353, 2008.

[8] U. S. Inan und R. A. Marshall, *Numerical electromagnetics: the FDTD method*. Cambridge University Press, 2011.

[9] A. Taflove, S. C. Hagness und M. Piket-May, „Computational electromagnetics: the finite-difference time-domain method," *The Electrical Engineering Handbook*, Jg. 3, Nr. 629-670, S. 15, 2005.

[10] K. Yee, „Numerical solution of initial boundary value problems involving Maxwell's equations in isotropic media," *IEEE Transactions on antennas and propagation*, Jg. 14, Nr. 3, S. 302–307, 1966.

[11] R. Courant, K. Friedrichs und H. Lewy, „Über die partiellen Differenzengleichungen der mathematischen Physik," *Mathematische annalen*, Jg. 100, Nr. 1, S. 32–74, 1928.

[12] N. Crouseilles, L. Einkemmer und J. Massot, „Exponential methods for solving hyperbolic problems with application to collisionless kinetic equations," *Journal of Computational Physics*, Jg. 420, S. 109 688, 2020.

[13] C. Moler und C. Van Loan, „Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later," *SIAM review*, Jg. 45, Nr. 1, S. 3–49, 2003.

[14] G. Faber, „Über polynomische Entwicklungen," *Mathematische Annalen*, Jg. 57, Nr. 3, S. 389–408, 1903.

[15] H. Kleene, „Effiziente numerische Zeitbereichsverfahren für die elektromagnetische Analyse von Komponenten der Photonik," 2021.

[16] H. Kleene, T. Luong und D. Schulz, „Faber Polynomial Based Approximations of Nonlinear Integrators for Electrodynamics," *IEEE Journal on Multiscale and Multiphysics Computational Techniques*, Jg. 6, S. 41–49, 2021.

[17] H Kleene und D Schulz, „On the evaluation of sources in highly accurate time domain simulations on the basis of faber polynomials," in *2018 Progress in Electromagnetics Research Symposium (PIERS-Toyama)*, IEEE, 2018, S. 352–356.

[18] A. G. Borisov und S. V. Shabanov, „Wave packet propagation by the Faber polynomial approximation in electrodynamics of passive media," *Journal of Computational Physics*, Jg. 216, Nr. 1, S. 391–402, 2006.

[19] H. Kleene und D. Schulz, „Time domain solution of Maxwell's equations using Faber polynomials," *IEEE Transactions on Antennas and Propagation*, Jg. 66, Nr. 11, S. 6202–6208, 2018.

[20] H. Fahs, „Investigation on polynomial integrators for time-domain electromagnetics using a high-order discontinuous Galerkin method," *Applied Mathematical Modelling*, Jg. 36, Nr. 11, S. 5466–5481, 2012.

[21] D. Dai, Y. Shi, S. He, L. Wosinski und L. Thylen, „Gain enhancement in a hybrid plasmonic nano-waveguide with a low-index or high-index gain medium," *Optics Express*, Jg. 19, Nr. 14, S. 12 925–12 936, 2011.

[22] Q. Wang und B. Arash, „A review on applications of carbon nanotubes and graphenes as nano-resonator sensors," *Computational Materials Science*, Jg. 82, S. 350–360, 2014.

[23] X. Yuan, Y. Liang, X. Hu, Y. Xu, Y. Chen und R. Kosonen, „Waste heat recoveries in data centers: A review," *Renewable and Sustainable Energy Reviews*, Jg. 188, S. 113 777, 2023.

[24] F. Meng, A. van Bezooijen und R. Mahmoudi, „A mismatch detector for adaptive antenna impedance matching," in *2006 European Microwave Conference*, IEEE, 2006, S. 1457–1460.

[25] R. Pandey, S. Tekumalla und M. Gupta, „Effect of defects on electromagnetic interference shielding effectiveness of magnesium," *Journal of Materials Science: Materials in Electronics*, Jg. 29, Nr. 11, S. 9728–9739, 2018.

[26] J. Jahns, *Photonik: Grundlagen, Komponenten und Systeme*. Walter de Gruyter, 2009.

[27] G. A. Reider, *Photonik: eine Einführung in die Grundlagen*. Springer-Verlag, 2013.

[28] S. Zhang, C. Menoni, V. Gruzdev und E. Chowdhury, „Ultrafast laser material damage simulation—a new look at an old problem," *Nanomaterials*, Jg. 12, Nr. 8, S. 1259, 2022.

[29] N. Bloembergen, „Laser-induced electric breakdown in solids," *IEEE Journal of Quantum Electronics*, Jg. 10, Nr. 3, S. 375–386, 2003.

[30] AMD. „AMD Instinct MI200-Serie Beschleuniger." Zugriff am 10. Juni 2025. Adresse: `https://www.amd.com/de/products/accelerators/instinct/mi200.html`

[31] NVIDIA. „NVIDIA RTX PRO 6000 Blackwell Workstation-Edition." Zugriff am 10. Juni 2025. Adresse: `https://www.nvidia.com/de-de/products/workstations/professional-desktop-gpus/rtx-pro-6000/`

[32] H. Kleene und D. Schulz, „Concept of a complex envelope faber polynomial approach for the solution of Maxwell's equations," in *2018 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization (NEMO)*, IEEE, 2018, S. 1–3.

[33] H. Kleene und D. Schulz, „Complex envelope Faber polynomial method for the solution of Maxwell's equations," *Optical and Quantum Electronics*, Jg. 51, Nr. 12, S. 381, 2019.

[34] H. Kleene und D. Schulz, „Assessment of a time domain beam propagation algorithm based on Faber polynomial expansions," in *2019 IEEE MTT-S International Conference on Numerical Electromagnetic and Multiphysics Modeling and Optimization (NEMO)*, IEEE, 2019, S. 1–3.

[35] H. Kleene und D. Schulz, „Explicit Wideband Time-Domain Beam Propagation Algorithm Based on Faber Polynomials," *IEEE Journal on Multiscale and Multiphysics Computational Techniques*, Jg. 4, S. 282–289, 2019.

[36] P. Suetin, „Series in Faber polynomials and several generalizations," *Journal of Soviet Mathematics*, Jg. 5, Nr. 4, S. 502–551, 1976.

[37] D. Sarkar, *FDTD analysis of guided electromagnetic wave interaction with time-modulated dielectric medium*. Springer, 2022.

[38] S. D. Gedney, *Introduction to the finite-difference time-domain (FDTD) method for electromagnetics*. Morgan & Claypool Publishers, 2011, Bd. 27.

[39] M Wnuk, G Ro, M Bugaj u. a., „The analysis of microstrip antennas using the FDTD method," *WIT Transactions on Modelling and Simulation*, Jg. 41, 2005.

[40] P. Kosmas, A. P. Feresidis und G. Goussetis, „Periodic FDTD analysis of a 2-D leaky-wave planar antenna based on dipole frequency selective surfaces," *IEEE transactions on antennas and propagation*, Jg. 55, Nr. 7, S. 2006–2012, 2007.

[41] G. Lehner, *Electromagnetic field theory for engineers and physicists*. Springer Science & Business Media, 2010.

[42] J. D. Joannopoulos, S. G. Johnson, J. N. Winn und R. D. Meade, „Molding the flow of light," *Princet. Univ. Press. Princeton, NJ [ua]*, Jg. 12, 2008.

[43] P. Drude, „Zur elektronentheorie der metalle," *Annalen der physik*, Jg. 306, Nr. 3, S. 566–613, 1900.

[44] D. Sarid und W. A. Challener, *Modern introduction to surface plasmons: theory, Mathematica modeling, and applications*. Cambridge university press, 2010.

[45] B. Schulkin, L. Sztancsik und J. F. Federici, „Analytical solution for photonic band-gap crystals using Drude conductivity," *American journal of physics*, Jg. 72, Nr. 8, S. 1051–1054, 2004.

[46] P.-f. Yang, J.-q. Yao und X. Wei, „Thz modulator based on the drude model," *Optoelectronics Letters*, Jg. 7, Nr. 1, S. 19–21, 2011.

[47] S. A. Maier u. a., *Plasmonics: fundamentals and applications*. Springer, 2007, Bd. 1.

[48] J. B. J. baron de Fourier, *Théorie analytique de la chaleur*. Firmin Didot, 1822.

[49] L. Nickelson, „Plasmonics," in *Electromagnetic Theory and Plasmonics for Engineers*, Springer, 2018, S. 611–695.

[50] G. Adam, O. Hittmair, G. Adam und O. Hittmair, „Das Debye-Modell," *Wärmetheorie*, S. 284–288, 1992.

[51] P. Mantas, „Dielectric response of materials: extension to the Debye model," *Journal of the European Ceramic Society*, Jg. 19, Nr. 12, S. 2079–2086, 1999.

[52] B. C. Truong, H. D. Tuan, H. H. Kha und H. T. Nguyen, „Debye parameter extraction for characterizing interaction of terahertz radiation with human skin tissue," *IEEE Transactions on Biomedical Engineering*, Jg. 60, Nr. 6, S. 1528–1537, 2013.

[53] R. A. Alahnomi u. a., „Review of recent microwave planar resonator-based sensors: Techniques of complex permittivity extraction, applications, open challenges and future research directions," *Sensors*, Jg. 21, Nr. 7, S. 2267, 2021.

[54] R. Gerhardt, „Impedance and dielectric spectroscopy revisited: distinguishing localized relaxation from long-range conductivity," *Journal of Physics and Chemistry of Solids*, Jg. 55, Nr. 12, S. 1491–1506, 1994.

[55] K. Wolf, G Briegleb und H. Stuart, „Kerr-Effekt, Lichtzerstreuung und Molekülstruktur," *Zeitschrift für Physikalische Chemie*, Jg. 6, Nr. 1, S. 163–209, 1929.

[56] J. Behringer, „Über den Zusammenhang von Raman-Streuung, Absorption und Fluoreszenz (Resonanz-Raman-Effekt)," *Zeitschrift für Elektrochemie, Berichte der Bunsengesellschaft für physikalische Chemie*, Jg. 62, Nr. 5, S. 544–567, 1958.

[57] Z. Lin u. a., „Using photonic crystal microrings to mitigate Raman-Kerr effects competition for soliton microcomb generation," *Journal of Lightwave Technology*, Jg. 42, Nr. 1, S. 268–275, 2023.

[58] U Keller, „Ultrafast all-solid-state laser technology," *Applied Physics B*, Jg. 58, Nr. 5, S. 347–363, 1994.

[59] S. M. Sze, Y. Li und K. K. Ng, *Physics of semiconductor devices*. John wiley & sons, 2021.

[60] W. T. Lotshaw, D. McMorrow, C. Kalpouzos und G. A. Kenney-Wallace, „Femtosecond dynamics of the optical Kerr effect in liquid nitrobenzene and chlorobenzene," *Chemical physics letters*, Jg. 136, Nr. 3-4, S. 323–328, 1987.

[61] C. Li, „Nonlinear optics," *Principles and Applications*, 2017.

[62] A. E. Siegman, *Lasers*. University science books, 1986.

[63] S.-L. Chua, Y. Chong, A. D. Stone, M. Soljačić und J. Bravo-Abad, „Low-threshold lasing action in photonic crystal slabs enabled by Fano resonances," *Optics express*, Jg. 19, Nr. 2, S. 1539–1562, 2011.

[64] A. Cerjan, A. Oskooi, S.-L. Chua und S. G. Johnson, „Modeling lasers and saturable absorbers via multilevel atomic media in the Meep FDTD software: Theory and implementation," *arXiv preprint arXiv:2007.09329*, 2020.

[65] P. Bermel, E. Lidorikis, Y. Fink und J. D. Joannopoulos, „Active materials embedded in photonic crystals and coupled to electromagnetic radiation," *Physical Review B—Condensed Matter and Materials Physics*, Jg. 73, Nr. 16, S. 165 125, 2006.

[66] A. Cerjan, Y. Chong, L. Ge und A. D. Stone, „Steady-state ab initio laser theory for n-level lasers," *Optics express*, Jg. 20, Nr. 1, S. 474–488, 2011.

[67] F. Teixeira, „A summary review on 25 years of progress and future challenges in FDTD and FETD techniques," *The Applied Computational Electromagnetics Society Journal (ACES)*, S. 1–14, 2010.

[68] H. R. Schwarz und N. Köckler, *Numerische mathematik*. Springer, 2006.

[69] R. J. LeVeque, *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*. SIAM, 2007.

[70] F. Gustrau, *Electromagnetic Design: Theorie und Simulation elektromagnetischer Felder*. Carl Hanser Verlag GmbH Co KG, 2023.

[71] F. Teixeira u. a., „Finite-difference time-domain methods," *Nature Reviews Methods Primers*, Jg. 3, Nr. 1, S. 75, 2023.

[72] D. B. Davidson, *Computational electromagnetics for RF and microwave engineering*. Cambridge University Press, 2010.

[73] N. R. Desai, K. K. Kesari und A. Agarwal, „Pathophysiology of cell phone radiation: oxidative stress and carcinogenesis with focus on male reproductive system," *Reproductive Biology and Endocrinology*, Jg. 7, S. 1–9, 2009.

[74] P. B. Wong, G. L. Tyler, J. E. Baron, E. M. Gurrola und R. A. Simpson, „A three-wave FDTD approach to surface scattering with applications to remote sensing of geophysical surfaces," *IEEE Transactions on Antennas and Propagation*, Jg. 44, Nr. 4, S. 504–514, 1996.

[75] C.-W. Huang, A. Elsherbeni, J. Chen und C. Smith, „FDTD characterization of meander line antennas for RF and wireless communications," *Progress In Electromagnetics Research*, Jg. 24, S. 185–199, 1999.

[76] X. Yuan, D. Borup, J. W. Wiskin, M. Berggren, R. Eidens und S. A. Johnson, „Formulation and validation of Berenger's PML absorbing boundary for the FDTD simulation of acoustic scattering," *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, Jg. 44, Nr. 4, S. 816–822, 1997.

[77] R. M. Joseph und A. Taflove, „FDTD Maxwell's equations models for nonlinear electrodynamics and optics," *IEEE Transactions on Antennas and Propagation*, Jg. 45, Nr. 3, S. 364–374, 1997.

[78] H. Chen, J. M. McMahon, M. A. Ratner und G. C. Schatz, „Classical electrodynamics coupled to quantum mechanics for calculation of molecular optical properties: a RT-TDDFT/FDTD approach," *The Journal of Physical Chemistry C*, Jg. 114, Nr. 34, S. 14 384–14 392, 2010.

[79] S. G. Garcıa, A. R. Bretones, B. G. Olmedo und R. G. Martın, „Finite difference time domain methods," *Time Domain Techniques in Computational Electromagnetics*, 2003.

[80] J. Geiser, *Computational Engineering*. Springer, 2018, Bd. 1.

[81] A Elsherbeni und V Demir, „The finite-difference," in *Time-domain Method for Electromagnetics with MATLAB Simulations*, SciTech Publishing, 2009, S. 231–231.

[82] G. Pelosi, „The finite-element method, Part I: RL Courant [historical corner]," *IEEE Antennas and Propagation Magazine*, Jg. 49, Nr. 2, S. 180–182, 2007.

[83] R. W. Clough, „Original formulation of the finite element method," *Finite elements in analysis and design*, Jg. 7, Nr. 2, S. 89–101, 1990.

[84] A. Prior, „Applications of implicit and explicit finite element techniques to metal forming," *Journal of Materials Processing Technology*, Jg. 45, Nr. 1-4, S. 649–656, 1994.

[85] D. Jiao und J.-M. Jin, „An effective algorithm for implementing perfectly matched layers in time-domain finite-element simulation of open-region EM problems," *IEEE transactions on antennas and propagation*, Jg. 50, Nr. 11, S. 1615–1623, 2002.

[86] K. Sagiyama, S. Govindjee und P.-O. Persson, „An efficient time-domain perfectly matched layers formulation for elastodynamics on spherical domains," *International Journal for Numerical Methods in Engineering*, Jg. 100, Nr. 6, S. 419–441, 2014.

[87] U. Basu, „Explicit finite element perfectly matched layer for transient three-dimensional elastic waves," *International journal for numerical methods in engineering*, Jg. 77, Nr. 2, S. 151–176, 2009.

[88] M. W. Evans, F. H. Harlow und E. Bromberg, „The Particle-In-Cell Method for Hydrodynamic Calculations," Los Alamos Scientific Laboratory, Techn. Ber., 1957.

[89] V. Shankar, W. F. Hall und A. H. Mohammadian, „A time-domain differential solver for electromagnetic scattering problems," *Proceedings of the IEEE*, Jg. 77, Nr. 5, S. 709–721, 2002.

[90] N. K. Madsen und R. W. Ziolkowski, „Numerical solution of Maxwell's equations in the time domain using irregular nonorthogonal grids," *Wave Motion*, Jg. 10, Nr. 6, S. 583–596, 1988.

[91] S. M. Rao, *Time domain electromagnetics*. Elsevier, 1999.

[92] C Fumeaux, D Baumann, G Almpanis, E. Li und R. Vahldieck, „Finite-Volume Time-Domain method for electromagnetic modelling: Strengths, limitations and challenges," *International Journal of Microwave and Optical Technology*, Jg. 3, Nr. 3, S. 318–328, 2008.

[93] D. Halliday, R. Resnick und J. Walker, *Fundamentals of physics*. John Wiley & Sons, 2013.

[94] F. M. L. M. M. Darwish, *The finite volume method in computational fluid dynamics: An advanced introduction with OpenFOAM and Matlab*. Springer, 2016.

[95] C. Fumeaux, K. Sankaran und R. Vahldieck, „Spherical perfectly matched absorber for finite-volume 3-D domain truncation," *IEEE transactions on microwave theory and techniques*, Jg. 55, Nr. 12, S. 2773–2781, 2007.

[96] D. J. Hoppe, *Impedance boundary conditions in electromagnetics*. CRC Press, 2018.

[97] Z.-g. Liu, „Fabry-Perot resonator antenna," *Journal of Infrared, Millimeter, and Terahertz Waves*, Jg. 31, Nr. 4, S. 391–403, 2010.

[98] Z. Mousavirazi, M. M. M. Ali, H. N. Gheisanab und T. A. Denidni, „Analysis and design of ultra-wideband PRGW hybrid coupler using PEC/PMC waveguide model," *Scientific reports*, Jg. 12, Nr. 1, S. 14 214, 2022.

[99] C. A. Balanis, *Advanced engineering electromagnetics*. John Wiley & Sons, 2012.

[100] J. Webb, „Absorbing boundary conditions for the finite-element analysis of planar devices," *IEEE transactions on microwave theory and techniques*, Jg. 38, Nr. 9, S. 1328–1332, 2002.

[101] X. Yuan, D. Borup, J. W. Wiskin, M. Berggren, R. Eidens und S. A. Johnson, „Formulation and validation of Berenger's PML absorbing boundary for the FDTD simulation of acoustic scattering," *IEEE transactions on ultrasonics, ferroelectrics, and frequency control*, Jg. 44, Nr. 4, S. 816–822, 1997.

[102] B. Engquist und A. Majda, „Absorbing boundary conditions for the numerical simulation of waves," *Mathematics of computation*, Jg. 31, Nr. 139, S. 629–651, 1977.

[103] G. Mur, „Absorbing boundary conditions for the finite-difference approximation of the time-domain electromagnetic-field equations," *IEEE transactions on Electromagnetic Compatibility*, Nr. 4, S. 377–382, 1981.

[104] G. Zheng, A. Kishk, A. W. Glisson und A. Yakovlev, „Implementation of Mur's absorbing boundaries with periodic structures to speed up the design process using finite-difference time-domain method," *Progress In Electromagnetics Research*, Jg. 58, S. 101–114, 2006.

[105] J.-P. Bérenger, „A historical review of the absorbing boundary conditions for electromagnetics," in *Forum for Electromagnetic Research Methods and Application Technologies*, Bd. 9, 2015, S. 1–28.

[106] J.-F. Mao, „Twofold Mur's first-order ABC in the FDTD method," *IEEE transactions on microwave theory and techniques*, Jg. 46, Nr. 3, S. 299–301, 2002.

[107] J. Cole und D Zhu, „Improved version of the second-order Mur absorbing boundary condition based on a nonstandard finite difference model," *Applied Computational Electromagnetics Society Journal (ACES)*, S. 375–381, 2009.

[108] J.-P. Berenger, „A perfectly matched layer for the absorption of electromagnetic waves," *Journal of computational physics*, Jg. 114, Nr. 2, S. 185–200, 1994.

[109] W. C. Chew und W. H. Weedon, „A 3D perfectly matched medium from modified Maxwell's equations with stretched coordinates," *Microwave and optical technology letters*, Jg. 7, Nr. 13, S. 599–604, 1994.

[110] C. M. Rappaport, „Perfectly matched absorbing boundary conditions based on anisotropic lossy mapping of space," *IEEE Microwave and Guided Wave Letters*, Jg. 5, Nr. 3, S. 90–92, 2002.

[111] Z. S. Sacks, D. M. Kingsland, R. Lee und J.-F. Lee, „A perfectly matched anisotropic absorber for use as an absorbing boundary condition," *IEEE transactions on Antennas and Propagation*, Jg. 43, Nr. 12, S. 1460–1463, 1995.

[112] R. W. Ziolkowski, „Time-derivative Lorentz materials and their utilization as electromagnetic absorbers," *Physical Review E*, Jg. 55, Nr. 6, S. 7696, 1997.

[113] S. D. Gedney, „An anisotropic PML absorbing media for the FDTD simulation of fields in lossy and dispersive media," *Electromagnetics*, Jg. 16, Nr. 4, S. 399–415, 1996.

[114] J.-P. Bérenger, „Application of the CFS PML to the absorption of evanescent waves in waveguides," *IEEE Microwave and Wireless Components Letters*, Jg. 12, Nr. 6, S. 218–220, 2002.

[115] M. Kuzuoglu und R. Mittra, „Frequency dependence of the constitutive parameters of causal perfectly matched anisotropic absorbers," *IEEE Microwave and Guided wave letters*, Jg. 6, Nr. 12, S. 447–449, 2002.

[116] J.-P. Berenger, „Perfectly matched layer for the FDTD solution of wave-structure interaction problems," *IEEE Transactions on antennas and propagation*, Jg. 44, Nr. 1, S. 110–117, 2002.

[117] B. D. Gvozdic und D. Z. Djurdjevic, „Performance advantages of CPML over UPML absorbing boundary conditions in FDTD algorithm," *Journal of Electrical Engineering*, Jg. 68, Nr. 1, S. 47, 2017.

[118] R. A. Horn und C. R. Johnson, *Matrix analysis*. Cambridge university press, 2012.

[119] D. A. Bini, S. Dendievel, G. Latouche und B. Meini, „Computing the exponential of large block-triangular block-Toeplitz matrices encountered in fluid queues," *Linear Algebra and its Applications*, Jg. 502, S. 387–419, 2016.

[120] L. Bergamaschi und M. Vianello, „Efficient computation of the exponential operator for large, sparse, symmetric matrices," *Numerical linear algebra with applications*, Jg. 7, Nr. 1, S. 27–45, 2000.

[121] J. Liesen und Z. Strakos, *Krylov subspace methods: principles and analysis*. Numerical Mathematics und Scie, 2013.

[122] W. E. Arnoldi, „The principle of minimized iterations in the solution of the matrix eigenvalue problem," *Quarterly of applied mathematics*, Jg. 9, Nr. 1, S. 17–29, 1951.

[123] C. Lanczos, „An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," 1950.

[124] K. Busch, J. Niegemann, M. Pototschnig und L. Tkeshelashvili, „A Krylov-subspace based solver for the linear and nonlinear Maxwell equations," *physica status solidi (b)*, Jg. 244, Nr. 10, S. 3479–3496, 2007.

[125] P. Bader, S. Blanes und F. Casas, „Computing the matrix exponential with an optimized Taylor polynomial approximation," *Mathematics*, Jg. 7, Nr. 12, S. 1174, 2019.

[126] N. J. Higham, „The scaling and squaring method for the matrix exponential revisited," *SIAM Journal on Matrix Analysis and Applications*, Jg. 26, Nr. 4, S. 1179–1193, 2005.

[127] P. Novati, „Solving linear initial value problems by Faber polynomials," *Numerical linear algebra with applications*, Jg. 10, Nr. 3, S. 247–270, 2003.

[128] W. Huisinga, L. Pesce, R. Kosloff und P. Saalfrank, „Faber and Newton polynomial integrators for open-system density matrix propagation," *The Journal of chemical physics*, Jg. 110, Nr. 12, S. 5538–5547, 1999.

[129] J. L. Ullman, „Studies in Faber polynomials. I," *Transactions of the American Mathematical Society*, Jg. 94, Nr. 3, S. 515–528, 1960.

[130] S. Cohn-Vossen, „Kürzeste Wege und Totalkrümmung auf Flächen," *Compositio mathematica*, Jg. 2, S. 69–133, 1935.

[131] P. K. Suetin, „Fundamental properties of Faber polynomials," *Russian Mathematical Surveys*, Jg. 19, Nr. 4, S. 121, 1964.

[132] B. Riemann, „Grundlagen für eine allgemeine Theorie der Functionen einer veränderlichen complexen Grösse. Inaugural dissertation, Göttingen, 1851," *Zweiter unveränderter Abdruck, Göttinger*, 1867.

[133] A. Bultheel, *Laurent series and their Padé approximations*. Birkhäuser, 2012, Bd. 27.

[134] G. Starke und R. S. Varga, „A hybrid Arnoldi-Faber iterative method for nonsymmetric systems of linear equations," *Numerische Mathematik*, Jg. 64, S. 213–240, 1993.

[135] D. Gaier, *Lectures on complex approximation*. Springer, 1987, Bd. 188.

[136] S. Ellacott, „A survey of Faber methods in numerical approximation," *Computers & Mathematics with Applications*, Jg. 12, Nr. 5-6, S. 1103–1107, 1986.

[137] L. Bieberbach und L. Bieberbach, „Die Cauchysche Integralformel," *Lehrbuch der Funktionentheorie: Band I: Elemente der Funktionentheorie*, S. 123–171, 1921.

[138] N. S. Nise, *Control systems engineering*. John Wiley & Sons, 2019.

[139] T. Driscoll, *Schwarz-Christoffel mapping*. Cambridge University Press, 2002.

[140] S. A. Gershgorin, „Uber die abgrenzung der eigenwerte einer matrix," Nr. 6, S. 749–754, 1931.

[141] H. De Raedt, K. Michielsen, J. S. Kole und M. T. Figge, „Solving the Maxwell equations by the Chebyshev method: A one-step finite-difference time-domain algorithm," *IEEE Transactions on Antennas and Propagation*, Jg. 51, Nr. 11, S. 3155–3160, 2003.

[142] H. Nyquist, „Certain topics in telegraph transmission theory," *Transactions of the American Institute of Electrical Engineers*, Jg. 47, Nr. 2, S. 617–644, 1928.

[143] M. Hochbruck, A. Ostermann und J. Schweitzer, „Exponential Rosenbrock-type methods," *SIAM Journal on Numerical Analysis*, Jg. 47, Nr. 1, S. 786–803, 2009.

[144] E. B. Christoffel, „Ueber einige allgemeine Eigenschaften der Minimumsflächen.," *Walter de Gruyter, Berlin/New York Berlin, New York*, 1867.

[145] H. A. Schwarz, „Ueber einige Abbildungsaufgaben.," *Walter de Gruyter, Berlin/New York Berlin, New York*, 1869.

[146] E. M. Stein und R. Shakarchi, *Complex analysis*. Princeton University Press, 2010, Bd. 2.

[147] E. Costamagna, „On the numerical inversion of the Schwarz-Christoffel conformal transformation," *IEEE transactions on microwave theory and techniques*, Jg. 35, Nr. 1, S. 35–40, 1987.

[148] M. A. Chaudhry, *An extended Schwarz-Christoffel transformation for analysis and design of high capacity integrated circuit lines and resistors*. University of California, Irvine, 1989.

[149] H. B. Palmer, „The capacitance of a parallel-plate capacitor by the Schwartz-Christoffel transformation," *Electrical Engineering*, Jg. 56, Nr. 3, S. 363–368, 1937.

[150] G. Savin, „Spannungserhöhung am Rande von Löchern," *Berlin: Verlag Technik*, 1956.

[151] S. C. Sanday und S. Sanday, „Evaluation of the Schwarz–Christoffel mapping function for special polygons," *SIAM Journal on Applied Mathematics*, Jg. 18, Nr. 4, S. 815–817, 1970.

[152] L. N. Trefethen, „Numerical computation of the Schwarz–Christoffel transformation," *SIAM Journal on Scientific and Statistical Computing*, Jg. 1, Nr. 1, S. 82–102, 1980.

[153] Z. Nehari, *Conformal mapping*. Courier Corporation, 2012.

[154] P. Henrici, *Applied and computational complex analysis, Volume 3: Discrete Fourier analysis, Cauchy integrals, construction of conformal maps, univalent functions*. John Wiley & Sons, 1993, Bd. 41.

[155] T. A. Driscoll, *Domain decomposition methods for conformal mapping and eigenvalue problems*. Cornell University, 1996.

[156] L. N. Trefethen, „SCPACK user's guide," *Numerical Analysis Report*, S. 89–2, 1989.

[157] T. A. Driscoll, „Algorithm 756: A MATLAB toolbox for Schwarz-Christoffel mapping," *ACM Transactions on Mathematical Software (TOMS)*, Jg. 22, Nr. 2, S. 168–186, 1996.

[158] N. Papamichael und N. Stylianopoulos, *Numerical conformal mapping: Domain decomposition and the mapping of quadrilaterals*. World Scientific, 2010.

[159] R. Menikoff und C. Zemach, „Methods for numerical conformal mapping," *Journal of Computational Physics*, Jg. 36, Nr. 3, S. 366–410, 1980.

[160] IEEE, „IEEE Standard for Floating-Point Arithmetic," *IEEE Std 754-2019 (Revision of IEEE 754-2008)*, S. 1–84, 2019.

[161] Mathworks. „Floating-Point Numbers." Zugriff am 18. Dezember 2024. Adresse: `https://de.mathworks.com/help/matlab/matlab_prog/floating-point-numbers.html`

[162] T. A. Driscoll und S. A. Vavasis, „Numerical conformal mapping using cross-ratios and Delaunay triangulation," *SIAM Journal on Scientific Computing*, Jg. 19, Nr. 6, S. 1783–1803, 1998.

[163] J. Curtiss, „Faber polynomials and the Faber series," *The American Mathematical Monthly*, Jg. 78, Nr. 6, S. 577–596, 1971.

[164] A.-L. Cauchy, *Oeuvres complètes d'Augustin Cauchy*. Gauthier-Villars, 1893, Bd. 1.

[165] F. P. Preparata und M. I. Shamos, *Computational geometry: an introduction*. Springer Science & Business Media, 2012.

[166] U. Meyer-Baese und U Meyer-Baese, *Digital signal processing with field programmable gate arrays*. Springer, 2007, Bd. 65.

[167] J. Geiser, *Multicomponent and Multiscale Systems*. Springer, 2016.

[168] M. J. Gander, L. Halpern und F. Nataf, „Optimal Schwarz waveform relaxation for the one dimensional wave equation," *SIAM Journal on Numerical Analysis*, Jg. 41, Nr. 5, S. 1643–1681, 2003.

[169] H. A. Schwarz, „Ueber einen Grenzübergang durch alternirendes Verfahren," 1870.

[170] M. L. Crow und M. Ilić, „The waveform relaxation method for systems of differential/algebraic equations," *Mathematical and computer modelling*, Jg. 19, Nr. 12, S. 67–84, 1994.

[171] M. Y. Ben Belgacem F., „The mortar element method for three dimensional finite elements," eng, *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, Jg. 31, Nr. 2, S. 289–302, 1997. Adresse: `http://eudml.org/doc/193838`

[172] C. Farhat und F.-X. Roux, „A method of finite element tearing and interconnecting and its parallel solution algorithm," *International journal for numerical methods in engineering*, Jg. 32, Nr. 6, S. 1205–1227, 1991.

[173] C. Farhat, P.-S. Chen und J. Mandel, „A scalable Lagrange multiplier based domain decomposition method for time-dependent problems," *International Journal for Numerical Methods in Engineering*, Jg. 38, Nr. 22, S. 3831–3853, 1995.

[174] J.-L. Lions, „Résolution d'EDP par un schéma en temps «pararéel» A"parareal"in time discretization of PDE's," *Academie des Sciences Paris Comptes Rendus Serie Sciences Mathematiques*, Jg. 332, Nr. 7, S. 661–668, 2001.

[175] A. Hindmarsh, P. Gresho und D. Griffiths, „The stability of explicit Euler time-integration for certain finite difference approximations of the multi-dimensional advection–diffusion equation," *International journal for numerical methods in fluids*, Jg. 4, Nr. 9, S. 853–897, 1984.

[176] J. C. Butcher, „A history of Runge-Kutta methods," *Applied numerical mathematics*, Jg. 20, Nr. 3, S. 247–260, 1996.

[177] Y. Maday, „The parareal in time algorithm," 2008.

[178] S. Lie, „Allgemeine Theorie der partiellen Differentialgleichungen erster Ordnung," *Mathematische Annalen*, Jg. 9, Nr. 2, S. 245–296, 1875.

[179] H. F. Trotter, „On the product of semi-groups of operators," *Proceedings of the American Mathematical Society*, Jg. 10, Nr. 4, S. 545–551, 1959.

[180] G. Strang, „On the construction and comparison of difference schemes," *SIAM journal on numerical analysis*, Jg. 5, Nr. 3, S. 506–517, 1968.

[181] M. Vovchanskyi, „A quick probability-oriented Introduction to Operator Splitting Methods," *Theory of Stochastic Processes*, Jg. 28 (44), Nr. 1, S. 50–110, 2024.

[182] J. H. Ferziger, M. Perić und R. L. Street, *Numerische strömungsmechanik*. Springer, 2008, Bd. 1.

[183] P. Monk und E. Suli, „Error estimates for Yee's method on non-uniform grids," *IEEE Transactions on magnetics*, Jg. 30, Nr. 5, S. 3200–3203, 1994.

[184] L. Liu, X. Li und F. Q. Hu, „Nonuniform-time-step explicit Runge–Kutta scheme for high-order finite difference method," *Computers & Fluids*, Jg. 105, S. 166–178, 2014.

[185] M. J. Berger und J. Oliger, „Adaptive mesh refinement for hyperbolic partial differential equations," *Journal of computational Physics*, Jg. 53, Nr. 3, S. 484–512, 1984.

[186] G. Kim, E. Arvas, V. Demir und A. Elsherbeni, „A novel nonuniform subgridding scheme for FDTD using an optimal interpolation technique," *Progress In Electromagnetics Research B*, Jg. 44, S. 137–161, 2012.

[187] M Brio, J. Moloney und A. Zakharian, „FDTD based second-order accurate local mesh refinement method for Maxwell's equations in two space dimensions," 2004.

[188] D. M. Pederson und L. L. Raja, „A stable finite-difference time-domain scheme for local time-stepping on an adaptive mesh," *Journal of Computational Physics*, Jg. 394, S. 456–476, 2019.

[189] M. Grote. „High-Order Explicit Local Time-Stepping Methods For Wave Propagation." Zugriff am 14. Februar 2025. Adresse: `https://team.inria.fr/magique3d/files/2016/03/talk_grote.pdf`

[190] B. Donderici und F. L. Teixeira, „Improved FDTD subgridding algorithms via digital filtering and domain overriding,“ *IEEE Transactions on Antennas and Propagation*, Jg. 53, Nr. 9, S. 2938–2951, 2005.

[191] K. J. Ebeling, *Integrierte Optoelektronik: Wellenleiteroptik. Photonik. Halbleiter*. Springer-Verlag, 2013.

[192] P. Monk und E. Süli, „A convergence analysis of Yee's scheme on nonuniform grids,“ *SIAM Journal on Numerical Analysis*, Jg. 31, Nr. 2, S. 393–412, 1994.

[193] Mathworks. „LAPACK in MATLAB.“ Zugriff am 13. Januar 2025. Adresse: `https://de.mathworks.com/help/matlab/math/lapack-in-matlab.html`

[194] T. Dortmund. „LiDO3.“ Zugriff am 13. Januar 2025. Adresse: `https://lido.itmc.tu-dortmund.de/lido3/`

[195] T. Dortmund. „LiDO3 Manual.“ Zugriff am 13. Januar 2025. Adresse: `https://lido.itmc.tu-dortmund.de/storages/lido-itmc/r/LiDO3_first_contact_handout.pdf`

[196] Mathworks. „Parallel Computing Toolbox.“ Zugriff am 14. Januar 2025. Adresse: `https://de.mathworks.com/products/parallel-computing.html`

[197] Mathworks. „Run Code on Parallel Pools.“ Zugriff am 14. Januar 2025. Adresse: `https://de.mathworks.com/help/parallel-computing/run-code-on-parallel-pools.html`

[198] D. R. Butenhof, *Programming with POSIX threads*. Addison-Wesley Professional, 1993.

[199] M. Kalin, „Modern C Up and Running,“ 2022.

[200] Mathworks. „Optimization Code Generation for Real-Time Applications.“ Zugriff am 14. Januar 2025. Adresse: `https://de.mathworks.com/help/optim/ug/real-time-code-generation.html`

[201] Mathworks. „GPU-Berechnungen.“ Zugriff am 16. Januar 2025. Adresse: `https://de.mathworks.com/solutions/gpu-computing.html`

[202] J. Crisp, *Introduction to fiber optics*. Elsevier, 2005.

[203] K. O. Hill, Y. Fujii, D. C. Johnson und B. S. Kawasaki, „Photosensitivity in optical fiber waveguides: Application to reflection filter fabrication,“ *Applied physics letters*, Jg. 32, Nr. 10, S. 647–647, 1978.

[204] Nvidida. „CUDA Toolkit v11.7.1.“ Zugriff am 20. Juni 2025. Adresse: `https://docs.nvidia.com/cuda/archive/11.7.1/cuda-c-best-practices-guide/index.html?utm_source=chatgpt.com`

[205] Nvidida. „CUDA C++ Programming Guide.“ Zugriff am 20. Juni 2025. Adresse: `https://docs.nvidia.com/cuda/cuda-c-programming-guide/`

[206] Y. Lee, K. G. Shin und H. S. Chwa, „Thermal-Aware Scheduling for Integrated CPUs–GPU Platforms,“ *ACM Transactions on Embedded Computing Systems (TECS)*, Jg. 18, Nr. 5s, S. 1–25, 2019.

[207] Mathworks. „Sparse.“ Zugriff am 19. Dezember 2024. Adresse: `https://de.mathworks.com/help/matlab/ref/sparse.html`

[208] P. Benner, W. Schilders, S. Grivet-Talocia, A. Quarteroni, G. Rozza und L. Miguel Silveira, *Model order reduction: volume 3 applications*. De Gruyter, 2020.

[209] L. Fortuna, G. Nunnari und A. Gallo, *Model order reduction techniques with applications in electrical engineering*. Springer Science & Business Media, 2012.

[210] A. Chatterjee, *An introduction to the proper orthogonal decomposition*. JSTOR, 2000, S. 808–817.

[211] V. Klema und A. Laub, „The singular value decomposition: Its computation and some applications," *IEEE Transactions on automatic control*, Jg. 25, Nr. 2, S. 164–176, 1980.

[212] G. W. Stewart, „On the early history of the singular value decomposition," *SIAM review*, Jg. 35, Nr. 4, S. 551–566, 1993.

[213] F. Chinesta, A. Huerta, G. Rozza und K. Willcox, „Model order reduction," *Encyclopedia of computational mechanics*, S. 1–36, 2016.

[214] G. H. Golub und C. Reinsch, „Singular value decomposition and least squares solutions," in *Handbook for Automatic Computation: Volume II: Linear Algebra*, Springer, 1971, S. 134–151.

[215] N. Kreimer und M. D. Sacchi, „A tensor higher-order singular value decomposition for prestack seismic data noise reduction and interpolation," *Geophysics*, Jg. 77, Nr. 3, S. V113–V122, 2012.

[216] R. A. Sadek, „SVD based image processing applications: state of the art, contributions and research challenges," *arXiv preprint arXiv:1211.7102*, 2012.

[217] B. Bermeitinger, T. Hrycej und S. Handschuh, „Singular value decomposition and neural networks," in *Artificial Neural Networks and Machine Learning–ICANN 2019: Deep Learning: 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17–19, 2019, Proceedings, Part II 28*, Springer, 2019, S. 153–164.

[218] L. N. Trefethen und D. Bau, *Numerical linear algebra*. SIAM, 2022.

[219] I. Markovsky, *Low rank approximation: algorithms, implementation, applications*. Springer, 2012, Bd. 906.

[220] C. Eckart und G. Young, „The approximation of one matrix by another of lower rank," *Psychometrika*, Jg. 1, Nr. 3, S. 211–218, 1936.

[221] D. F. Gleich, „Better than best low-rank approximation with the singular value decomposition," *arXiv e-prints*, arXiv–2402, 2024.

[222] A. Van der Sluis, *Condition numbers and equilibration of matrices*. Springer-Verlag Berlin/Heidelberg, 1969, Bd. 14, S. 14–23.

[223] R. Kiehl, „Die de Rham Kohomologie algebraischer Mannigfaltigkeiten über einem bewerteten Körper," *Publications Mathématiques de l'IHÉS*, Jg. 33, S. 5–20, 1967.

[224] P. Holmes, *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge university press, 2012.

[225] Z. Luo und G. Chen, *Proper orthogonal decomposition methods for partial differential equations*. Academic Press, 2018.

[226] L. Sirovich, „Turbulence and the dynamics of coherent structures. I. Coherent structures," *Quarterly of applied mathematics*, Jg. 45, Nr. 3, S. 561–571, 1987.

[227] R. Everson und L. Sirovich, *Karhunen–Loeve procedure for gappy data*. Optica Publishing Group, 1995, Bd. 12, S. 1657–1664.

[228] M. Turk und A. Pentland, „Eigenfaces for recognition," *Journal of cognitive neuroscience*, Jg. 3, Nr. 1, S. 71–86, 1991.

[229] E. Bouhoubeiny und P. Druault, „Note on the POD-based time interpolation from successive PIV images," *Comptes Rendus Mécanique*, Jg. 337, Nr. 11-12, S. 776–780, 2009.

[230] Z. Luo, J. Gao und Z. Xie, *Reduced-order finite difference extrapolation model based on proper orthogonal decomposition for two-dimensional shallow water equations including sediment concentration*, 2015.

[231] S. Chaturantabut und D. C. Sorensen, „Discrete empirical interpolation for nonlinear model reduction," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, IEEE, 2009, S. 4316–4321.

[232] M. Hinze, J. N. Kutz, O. Mula und K. Urban, *Model Order Reduction and Applications*. Springer, 2021.

[233] Z. Luo und J. Gao, „A POD reduced-order finite difference time-domain extrapolating scheme for the 2D Maxwell equations in a lossy medium," *Journal of Mathematical Analysis and Applications*, Jg. 444, Nr. 1, S. 433–451, 2016.

[234] M. Gavish und D. L. Donoho, „The optimal hard threshold for singular values is $4/\sqrt{3}$," *IEEE Transactions on Information Theory*, Jg. 60, Nr. 8, S. 5040–5053, 2014.

[235] F. Bai und Y. Wang, „Reduced-order modeling based on hybrid snapshot simulation," *International Journal of Computational Methods*, Jg. 18, Nr. 01, S. 2 050 029, 2021.

[236] F. Bai und Y. Wang, „DEIM reduced order model constructed by hybrid snapshot simulation," *SN Applied Sciences*, Jg. 2, Nr. 12, S. 2165, 2020.

[237] M. Brand, „Incremental singular value decomposition of uncertain data with missing values," in *Computer Vision—ECCV 2002: 7th European Conference on Computer Vision Copenhagen, Denmark, May 28–31, 2002 Proceedings, Part I 7*, Springer, 2002, S. 707–720.

[238] Y. Zhang, „An answer to an open question in the incremental SVD," *arXiv preprint arXiv:2204.05398*, 2022.

[239] M. W. Berry, *Large-scale sparse singular value computations*. SAGE Publications Sage UK: London, England, 1992, Bd. 6, S. 13–49.

[240] F. Bai, „Reduced Order Modeling Based on Hybrid Snapshot Simulation," *University of South Carolina*, 2021.

[241] X. Fu und J. N. Kutz, „Adaptive dimensionality-reduction for time-stepping in differential and partial differential equations," *Numerical Mathematics: Theory, Methods and Applications*, Jg. 10, Nr. 4, S. 872–894, 2017.

[242] M.-L. Rapún und J. M. Vega, „Reduced order models based on local POD plus Galerkin projection," *Journal of Computational Physics*, Jg. 229, Nr. 8, S. 3046–3063, 2010.

[243] K. Wang u. a., „Advances in optical fiber sensors based on multimode interference (MMI): a review," *IEEE Sensors Journal*, Jg. 21, Nr. 1, S. 132–142, 2020.

[244] L. B. Soldano und E. C. Pennings, „Optical multi-mode interference devices based on self-imaging: principles and applications,“ *Journal of lightwave technology*, Jg. 13, Nr. 4, S. 615–627, 2002.

[245] E. Zürich. „Guided waves in optical waveguides.“ Zugriff am 24. August 2025. Adresse: `https://people.ee.ethz.ch/~fyuriy/oe/oe_optcom_chapters/Optoelectronics_2010_Ch03.pdf`

[246] J. L. Greathouse und M. Daga, „Efficient sparse matrix-vector multiplication on GPUs using the CSR storage format,“ in *SC'14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, IEEE, 2014, S. 769–780.

[247] G Singh, R. Yadav und V Janyani, „Multimode interference (MMI) coupler based all optical switch: Design, Applications & Performance Analysis,“ *International Journal of Recent Trends in Engineering*, Jg. 1, Nr. 3, S. 115–119, 2009.

Anhang A: Herleitungen

Da nur wenige analytische Abbildungsfunktionen $\psi(\mathfrak{w})$ bekannt sind und in der einschlägigen Literatur nach bestem Wissen keine detaillierte Beschreibung zur numerischen Approximation der Entwicklungskoeffizienten c_m zu finden ist, wird die Herleitung im Folgenden schrittweise erläutert. Somit kann die Abbildung vom Einheitskreis auf beliebige Polygone erfolgen.

A.1 Entwicklungskoeffizienten zur SC-Transformation

Den Ausgangspunkt für die Entwicklungskoeffizienten repräsentiert die Gleichung

$$c_m(\Delta t_s) = \frac{1}{2\pi j} \int\limits_{|\mathfrak{w}|=\varrho} \frac{f(\psi(\mathfrak{w}))}{\mathfrak{w}^{m+1}} d\mathfrak{w}. \tag{A.1}$$

Wenn wie im Rahmen der Arbeit der Matrixexponential-Ansatz $\exp(\Delta t\mathcal{H})$ als Operator $f(\psi(\mathfrak{w}))$ verwendet wird, resultiert

$$c_m(\Delta t_s) = \frac{1}{2\pi j} \int\limits_{|\mathfrak{w}|=\varrho} \frac{\exp(\Delta t_s \psi(\mathfrak{w}))}{\mathfrak{w}^{m+1}} d\mathfrak{w}. \tag{A.2}$$

Als Beispiel für die inverse Abbildungsfunktion $\psi(\mathfrak{w})$ dient ein beliebiges Polygon aus dem Abschnitt 5.2, solange die Taylorentwicklung zur Approximation des SC Integrals bei $\mathfrak{w}^{-3}$ terminiert:

$$\psi(\mathfrak{w}) = \gamma\mathfrak{w} + \gamma_0 + \gamma_1\mathfrak{w}^{-1} + \gamma_2\mathfrak{w}^{-2} + \gamma_3\mathfrak{w}^{-3}. \tag{A.3}$$

Das Einsetzen der Gleichung (A.3) in die Gleichung (A.2) führt zu

$$c_m(\Delta t_s) = \frac{1}{2\pi j} \int\limits_{|\mathfrak{w}|=\varrho} \frac{\exp\big(\Delta t_s(\gamma\mathfrak{w} + \gamma_0 + \gamma_1\mathfrak{w}^{-1} + \gamma_2\mathfrak{w}^{-2} + \gamma_3\mathfrak{w}^{-3})\big)}{\mathfrak{w}^{m+1}} d\mathfrak{w}. \tag{A.4}$$

Bevor das Integral mit der Cauchy-Integralformel gelöst wird, ist es ratsam die Exponentialfunktion als Exponentialreihe auszudrücken:

$$c_m(\Delta t_s) = \frac{1}{2\pi j} \int\limits_{|\mathfrak{w}|=\varrho} \sum_{i=0}^{\infty} \left(\frac{{\Delta t_s}^i}{i!} (\gamma\mathfrak{w} + \gamma_0 + \gamma_1\mathfrak{w}^{-1} + \gamma_2\mathfrak{w}^{-2} + \gamma_3\mathfrak{w}^{-3})^i \right) \frac{1}{\mathfrak{w}^{m+1}} d\mathfrak{w}. \tag{A.5}$$

Nun liegt es nahe, die Cauchy-Integralformel für die Polstelle der Ordnung $m+1$ bei $\mathfrak{w} = 0$ auszuwerten. Die Schwierigkeit, die sich hierbei ergibt ist, dass die Monome eine negative Ordnung haben, wodurch bei Anwendung der Cauchy-Integralformel $\lim\limits_{\mathfrak{w}\to 0} \mathfrak{w}^{-i} = \infty$ folgen wird. Deshalb wird vorgeschlagen, die Ausdrücke auf einen Nenner zu bringen:

$$c_m(\Delta t_s) = \frac{1}{2\pi j} \int\limits_{|\mathfrak{w}|=\varrho} \sum_{i=0}^{\infty} \left(\frac{{\Delta t_s}^i}{i!} (\gamma\mathfrak{w}^4 + \gamma_0\mathfrak{w}^3 + \gamma_1\mathfrak{w}^2 + \gamma_2\mathfrak{w}^1 + \gamma_3)^i \frac{1}{\mathfrak{w}^{m+1+3i}} \right) d\mathfrak{w}. \tag{A.6}$$

Somit kann die Cauchy-Integralformel angewandt werden, ohne dass divergierende Terme auftreten:

$$c_m(\Delta t_s) = \lim_{\mathfrak{w}\to 0} \sum_{i=0}^{\infty} \left[\frac{1}{(m+3i)!} \cdot \frac{d^{m+3i}}{d\mathfrak{w}^{m+3i}} \left(\frac{{\Delta t_s}^i}{i!} (\gamma \mathfrak{w}^4 + \gamma_0 \mathfrak{w}^3 + \gamma_1 \mathfrak{w}^2 + \gamma_2 \mathfrak{w}^1 + \gamma_3)^i \right) \right]. \quad \text{(A.7)}$$

An dieser Stelle ist es wichtig zu beachten, dass innerhalb der Summe nur das Monom mit der Ordnung $m + 3i$ zur Kumulation beiträgt. Alle Monome mit einer geringeren Ordnung entfallen durch die Ableitung. Alle Monome mit einer höheren Ordnung konvergieren aufgrund des Limes gegen Null. Somit ist es von großer Bedeutung die Koeffizienten des i-fach potenzierten Polynoms zu bestimmen. Mit Hilfe der Cauchy-Produktformel, welche eine diskrete Faltung darstellt, ist dies ohne großen Aufwand direkt möglich. Zur Veranschaulichung werden die γ-Koeffizienten des 3-Ecks aus dem Abschnitt 5.2 entnommen. In der Tabelle A.1 sind in der Zeile $i = 1$ die ursprünglichen Koeffizienten der Monome zu sehen. Es gilt die Zahlen in den durchgezogenen Rahmen bei $m = 0$ zu addieren, bei $m = 1$ sind es die Zahlen im gestrichelten Rahmen. Bei jedem höherem m müsste entsprechend der Wert in der Spalte $m + 3i$ aus der Zeile i in der Summe berücksichtigt werden. Die Tabelle kann bis zur Grenze der Maschinengenauigkeit der CPU erweitert werden, wodurch c_m mit hoher Präzision approximiert werden kann. Zuletzt sei noch erwähnt, dass der Vorfaktor $\frac{1}{(m+3i)!}$ ebenfalls entfällt, da das relevante Monom genau $m + 3i$ mal abgeleitet wird, wodurch sich beide Terme gegenseitig kürzen.

Tabelle A.1: Relevanten γ-Koeffizienten zur Berechnung der Entwicklungskoeffizienten c_m.

	Koeffizient									
Potenz	γ_0	γ_1	γ_2	γ_3	γ_4	γ_5	γ_6	γ_7	γ_8	$\cdots$
$i = 0$	1									
$i = 1$	-0,02	-0,03	0,24	-0,49	0,29					
$i = 2$	2e-4	8e-4	-7e-3	7e-2	0,08	-0,24	0,35	-0,27	0,09	
$\vdots$										

Anhang B: Veranschaulichung

Zum besseren Verständnis der lokalen Operatoren wird der Ablauf anhand eines einfachen Beispiels illustriert. Dabei wird Schritt für Schritt verdeutlicht, wie die Extraktion, Expansion und Assemblierung durchgeführt werden.

B.1 Demonstration des Ablaufs der lokalen Operatoren

Schritt 1: Die Grundlage der Demonstration bildet die Diskretisierung nach Yee, wie die Abbildung B.1 illustriert. Zunächst wird ein 1D-Rechengebiet gezeigt, welches abwechselnd fünf $\vec{E}$- und vier $\vec{H}$-Feldkomponenten aufweist. Aufgrund der homogenen Diskretisierung sind die Diskretisierungspunkte äquidistant. Die Elemente innerhalb der zugehörigen dünnbesetzten Systemmatrix veranschaulichen die charakteristische Stufenstruktur. Im entsprechenden Zustandsvektors sind diese Elemente zugeordnet.

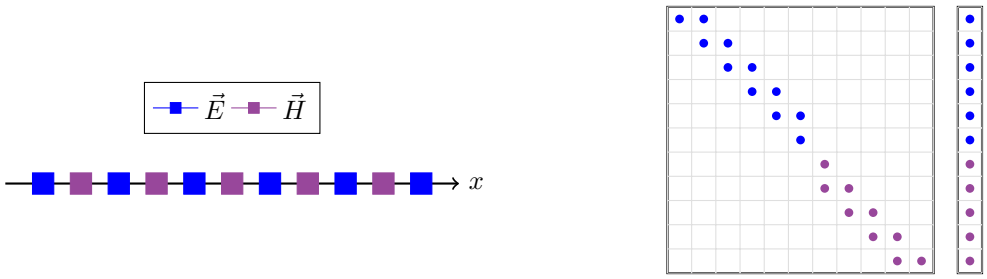


Abbildung B.1: Schritt 1 der Demonstration der lokalen Operatoren.

Schritt 2: Der Einfachheit halber wird im Folgenden ausschließlich auf die Elemente der $\vec{E}$-Feldkomponente eingegangen. Demnach ist bei der Matrix und dem Vektor in der Abbildung B.2 ein vergrößerter Ausschnitt der Abbildung B.1 zu sehen.

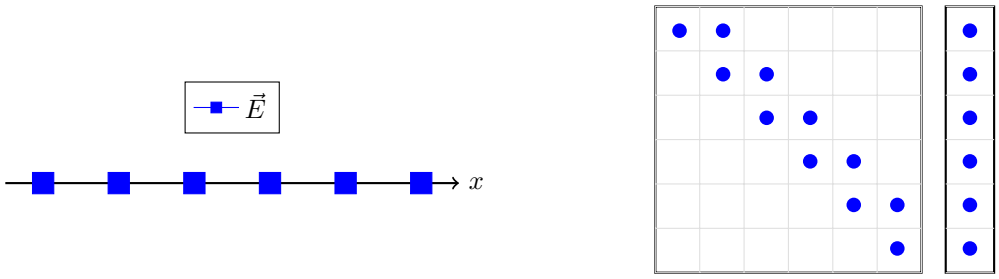


Abbildung B.2: Schritt 2 der Demonstration der lokalen Operatoren.

Schritt 3: Einige Anwendungen erfordern beispielsweise an Ecken und Kanten eine höhere lokale Auflösung als die zuvor global definierte. Daher entstehen zwei Subbereiche, von denen einer eine grobe und der andere eine feinere Auflösung besitzt. Im Rahmen der lokalen Operatoren wird eine Extraktion angestrebt, bei der die Submatrizen separat evaluiert werden, die mit den jeweiligen

Elementen des Gebiets verknüpft sind. Während die Anordnung der Diskretisierungspunkte keine Herausforderung darstellt, gilt dies nicht für die Extraktion der Submatrizen. Intuitiv würden aus der 6×6 Matrix jeweils zwei 3×3 Matrizen für die nachfolgenden Teilevaluierungen entnommen werden. Die Schwierigkeit dieses Vorgehens liegt an der Missachtung der bestehenden Kopplung genau an der Schnittstelle zwischen dem groben und feinen Gebiet. Infolgedessen käme es zu zwei vollständig entkoppelten Teilevaluierungen, die fehlerhafte Ergebnisse zur Folge hätten. Die Erweiterung der feinen Submatrix zu einer 4×3 Matrix repräsentiert keinen zulässigen Ansatz, da stets quadratische Ordnungen aus Konsistenzgründen benötigt werden. Der Sachverhalt wird in der Abbildung B.3 mit $N_x = 6$ Diskretisierungspunkten veranschaulicht.

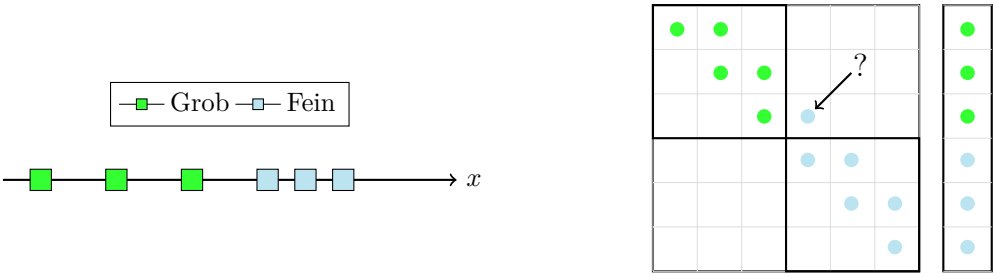


Abbildung B.3: Schritt 3 der Demonstration der lokalen Operatoren.

Schritt 4: Um einen mathematisch sauberen Ablauf zu gewährleisten, werden anstatt der 3×3 Submatrizen jeweils eine zusätzliche Spalte und Zeile berücksichtigt, sodass Submatrizen mit der Ordnung 4×4 extrahiert werden. Hiermit ergibt sich eine Überlappung zwischen den Gebieten, wodurch die Kopplung unmittelbar mit einbezogen wird, wie in der Abbildung B.4 zu erkennen ist. Analog dazu werden im Zustandsvektor die Elemente an der Schnittstelle der Überlappung zugewiesen, aufgrund der Expansion in das jeweils andere Gebiet.

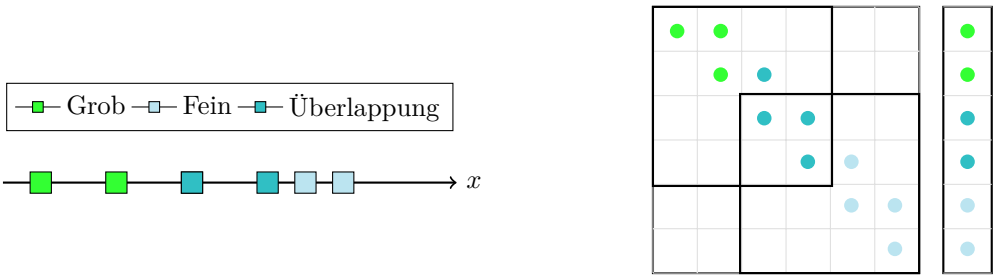


Abbildung B.4: Schritt 4 der Demonstration der lokalen Operatoren.

Schritt 5: Nach der korrekten Zuweisung der Elemente kann die eigentliche Extraktion vorgenommen werden. Der zeitabhängige Propagator wird für das feine und grobe Subgebiet unabhängig voneinander evaluiert. Zur Klarheit wird für beide Teilevaluierungen weiterhin die gesamte Systemmatrix illustriert, wobei nur die Elemente innerhalb der Untermatrix tatsächlich betrachtet werden. Die bisher nicht berücksichtigten Elemente werden im Subzustandsvektor mit Null angegeben, wie die Abbildung B.5 darlegt.

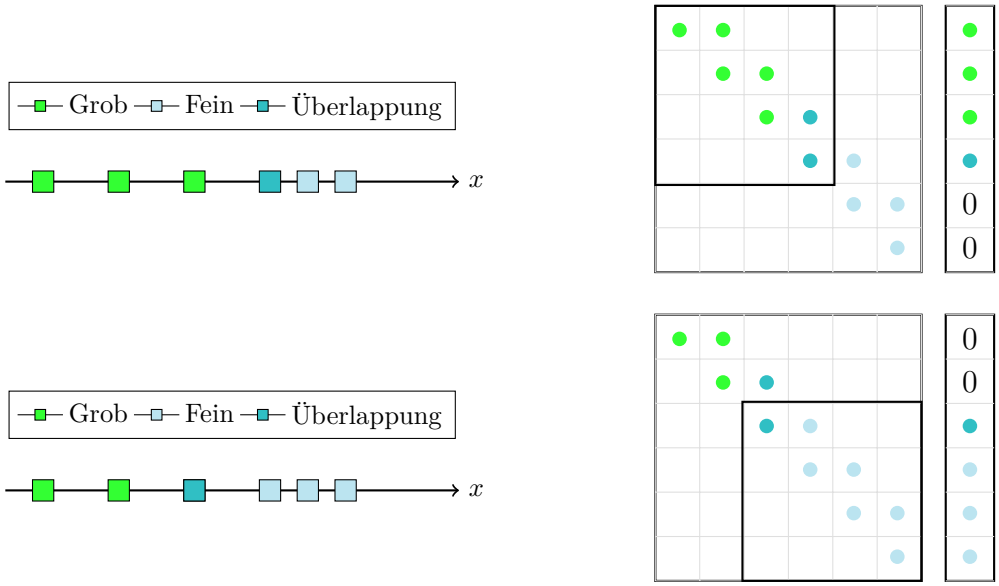


Abbildung B.5: Schritt 5 der Demonstration der lokalen Operatoren.

Schritt 6: In der Abbildung B.6 kann die erste Expansion im Zuge der Entwicklung mit der Faber-Methode nachvollzogen werden. Mit zunehmender Entwicklungsordnung m der Faber-Methode erhöht sich die Ordnung der Submatrizen jeweils um eine Zeile und eine Spalte. Hierbei sind insbesondere die Subzustandsvektoren ausschlaggebend, welche kontinuierlich weniger Nulleinträge enthalten. Der Rechenzeitgewinn durch die lokalen Operatoren resultiert infolge der unterschiedlichen Entwicklungsordnungen der Teilevaluierungen, wobei der Propagator im feinen Subgebiet mit $m_{\mathsf{f}} \rightarrow N_{pol}$ und im groben Subgebiet mit $m_{\mathsf{g}} \rightarrow N_{pol}$ ausgewertet wird. In diesem Beispiel wird $m_{\mathsf{f}} = m_{\mathsf{g}}$ angenommen, jedoch kommt die Laufzeitreduktion primär durch $m_{\mathsf{f}} > m_{\mathsf{g}}$ zur Geltung.

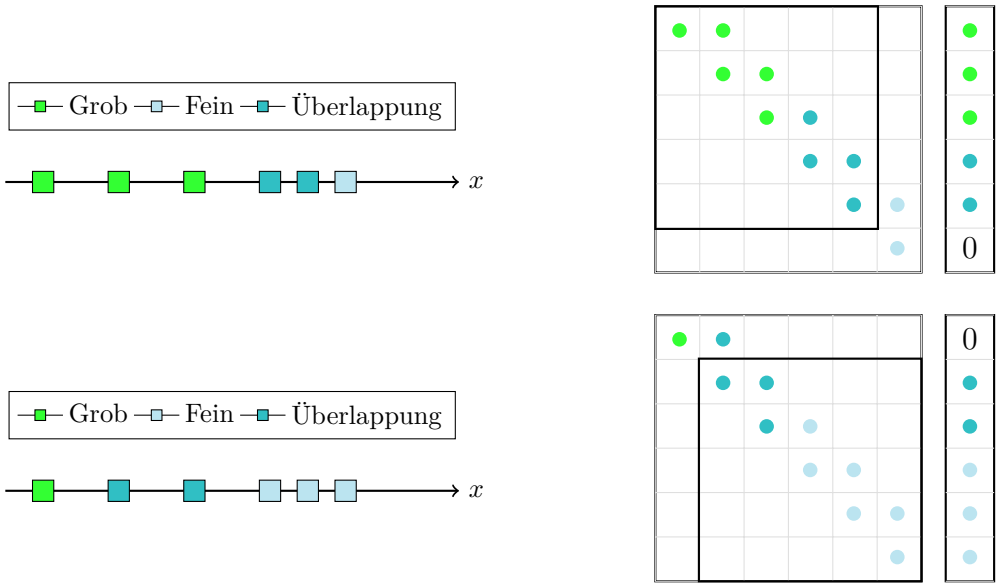


Abbildung B.6: Schritt 6 der Demonstration der lokalen Operatoren.

Schritt 7: Das Ausmaß der Expansion ergibt sich aus der Entwicklungsordnung c_m der Faber-Methode, welche wiederum von der Zeitschrittweite Δt abhängt. In der Abbildung B.7 gilt $m = m_{\mathsf{f}} = m_{\mathsf{g}} = N_{pol}$, was das Ende der Expansion markiert. Dementsprechend sind die Subzustandsvektoren voll besetzt. An dieser Stelle ist größte Limitierung der lokalen Operatoren ersichtlich. Wird eine Zeitschrittweite Δt vorgegeben, die ein $N_{pol} > N_x$ erfordert, erfolgt eine Expansion in Bereiche außerhalb des definierten Rechengebiets. Aus diesem Grund können unvorhersehbare Fehler auftreten. Dennoch lässt sich aufgrund dieser Tatsache die maximal erlaubte Zeitschrittweite exakt im Voraus vorhersagen, da die Entwicklungsordnung auf Basis von c_m mit relativ geringem Aufwand bestimmt werden kann.

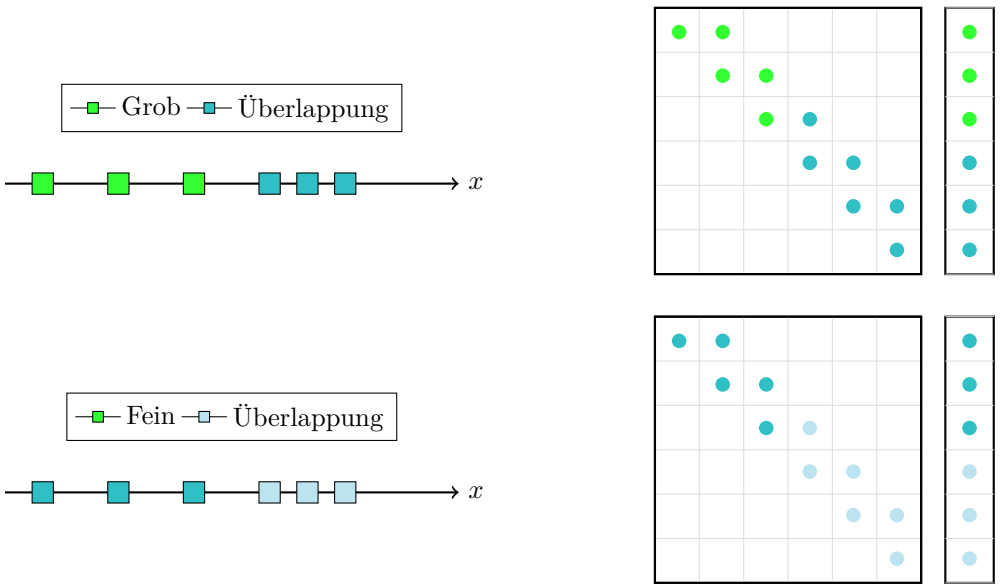


Abbildung B.7: Schritt 7 der Demonstration der lokalen Operatoren.

Schritt 8: Als letzter Schritt im Prozessablauf der lokalen Operatoren müssen die Ergebnisse der Teilevaluierungen assembliert werden. Dabei werden die Subzustandsvektoren einer vektorweisen Addition unterzogen. Das Ergebnis wird in einem neuen Subzustandsvektor gespeichert, der die Feldwerte der $\vec{E}$-Feldkomponente zum nächsten Zeitpunkt enthält. Der gesamte Ablauf für die $\vec{H}$-Feldkomponente wird analog bis zu diesem Schritt durchgeführt. Zur Erzeugung des Zustandsvektors werden die ermittelten Subzustandsvektoren der einzelnen Feldkomponente wieder zusammengeführt, indem sie hintereinander angehängt werden, wie in der Abbildung B.1.

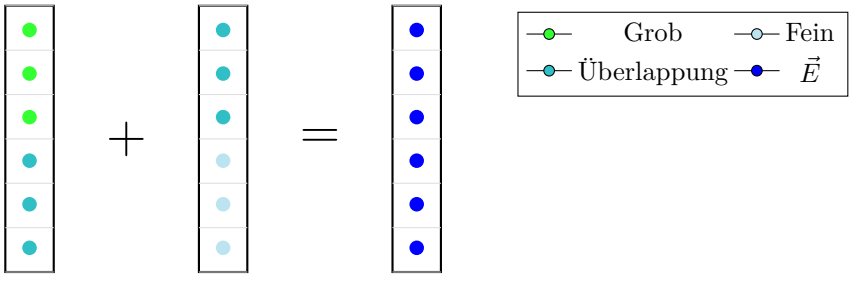


Abbildung B.8: Schritt 8 der Demonstration der lokalen Operatoren.

Ich widme diese Arbeit meiner Familie,
die mir stets mit ihrer Unterstützung, ihrem Glauben und Vertrauen zur Seite stand.
Insbesondere widme ich diese Arbeit meiner Frau,
die mir stets mit ihrer Liebe, Fürsorge und Geborgenheit Kraft gegeben hat,
selbst in den schwierigsten Phasen.
Ich liebe euch.