Transregio 391

TRR 391 Working Paper #11

May 2026

Fast Factor Extraction for Mixed Type Financial Data

Fabian Schmidt[a], Matei Demetrescu[a]

Suggested citation:
Schmidt, F. and Demestrescu, M. (2026). "Fast Factor Extraction for Mixed Type Financial Data". *TRR 391 Working Paper #11*. DOI: 10.17877/DE290R-26653.

Version 1.0, May 2026

[a] Department of Statistics, TU Dortmund University
Corresponding author: mdeme@statistik.tu-dortmund.de

Fast Factor Extraction for Mixed Type Financial Data*

Fabian Schmidt[a], Matei Demetrescu[a]

[a] Department of Statistics, TU Dortmund University

May 20, 2026

Abstract

Empirical research has access to ever larger datasets of mixed data types. The sheer amount of available data may often lead to the use of techniques such as dimensionality reduction or shrinkage. To reduce dimensionality, it is not uncommon to assume that the data are driven by a small number of common factors. For metric, or continuous, data, factor extraction may be conducted by means of standard principal component analysis [PCA]. PCA is, however, not directly applicable to count or binary data. Mixed data types may be dealt with via generalized linear models driven analogously by latent factors, and we discuss fast implementations of maximum likelihood estimators based on specific alternating least squares regressions. We demonstrate their practical applicability in simulations and in an application to factor-based forecasts of excess returns.

Keywords: factor models; mixed data types; alternating least squares.

JEL classifications: C55 (Large Data Sets: Modeling and Analysis); C61 (Optimization Techniques); G17 (Financial Forecasting and Simulation).

*The authors gratefully acknowledge financial support from the German Research Foundation (DFG) within TRR 391: Spatio-temporal Statistics for the Transition of Energy and Transport (520388526). Address correspondence to: Matei Demetrescu, TU Dortmund University, Department of Statistics, Vogelpothsweg 78, 44227 Dortmund, Germany. Email: mdeme@statistik.tu-dortmund.de.

1 Motivation

As data collection becomes less costly, both financially and in terms of the necessary effort, empirical research has increasing access to large datasets in many fields, *e.g.* finance, social sciences, medicine, or physics. Consider, for instance, the literature on equity premium prediction which produced a large variety of presumed predictors; cf. Goyal *et al.* (2024) and the references therein for an overview. Even if many of them are already aggregating many sources of information, these predictors typically exhibit only low signal-to-noise ratios (see Campbell (2008) and Phillips and Lee (2013), among others). Also, there are often more predictors than there are observations in time. Such issues may be side-stepped by applying dimensional reduction techniques (or, in machine learning lingo, suitable feature engineering), where one constructs a few new variables that are intended to capture the essential information contained in the set of (many) observable variables. For example, in macroeconomic forecasting, this approach was popularized by the seminal work of Stock and Watson (2002a,b), *inter alia*, with subsequent work often focusing on continuous predictors; cf. the FRED-MD database of McCracken and Ng (2016) or the datasets of Welch and Goyal (2008) and Goyal *et al.* (2024) which are frequently applied in macroeconomics and finance. Furthermore, such increasingly complex data sets often contain more than just metric data. Count data is indeed becoming more and more common in corporate finance, cf. Cohn *et al.* (2022, p. 529), while technical indicators are making up an important part of stock market analysis.

Although Principal Component Analysis [PCA] is a highly convenient method for dimensionality reduction in continuous datasets, Collins *et al.* (2001) point out that it is not appropriate for non-continuous data. As an alternative, they argue that the (quasi-)Gaussian assumption commonly used to justify PCA is merely a special case within the exponential family of distributions, which also includes, among others, the Binomial and the Poisson distribution, which are suited for binary or count data. More generally, the exponential family framework allows for dimensional reduction by assuming that the respective natural parameter of variable i's distribution is parameterized by $\boldsymbol{\lambda}_i \mathbf{F}_t$ where the first entries in both vectors correspond to an intercept (Lu *et al.*, 2018, pp. 194 f.). Then, the resulting negative data likelihood is minimized with respect to both the latent variables and loadings. Collins *et al.* (2001) solve this optimization problem by applying coordinate descent to minimize the loss with respect to loadings $\boldsymbol{\lambda}_i$ and factors $\mathbf{F}_t$ in an alternating fashion. Murphy (2012, p. 947) interprets this approach as a degenerate EM algorithm with point estimates of $\mathbf{F}_t$ being calculated in the E step. Meanwhile, Welling *et al.* (2008) interpret it as a variational EM algorithm and find that it is susceptible to overfitting and numerical instabilities in their application, since the number of latent variables grows proportionally to the size of the dataset. Alternative extraction approaches rely on different variational EM algorithms, such as the method proposed by Khan *et al.* (2010), which preserves posterior uncertainty over the latent variables rather than treating them as fixed parameters. More re-

cently, Bhamidipaty *et al.* (2025) discuss Bregman divergence-based optimization techniques, whereas Liu *et al.* (2018) propose a non-iterative approach utilizing a (shrinked) eigendecomposition of a specific bias- and noise-reduced covariance matrix estimator that is motivated by their single-particle imaging application.

Although exponential PCA can be used as a tool to extract latent factors (Lu *et al.*, 2018, p. 208), it does not appear to be in widespread use, possibly due to the limited number of implementations (Bhamidipaty *et al.*, 2025, p. 1). To the best of our knowledge, studies that simultaneously investigate multiple data types remain rare, despite growing data availability making them of particular interest. Although Collins *et al.* (2001) mention the potential for an approach that considers different distributions for different variables, they limit their subsequent analysis to a Poisson-only example. In this regard, Creal *et al.* (2014) pose one of the very few exceptions, with their observation-driven factor model applied to credit risk analysis simultaneously employing an ordered logit model and distinguishing (un-)constrained continuous variables via suitable distributional assumptions.

Therefore, we propose a simplified algorithm based on widely known regression techniques to make the framework more accessible by being easy to replicate in practice and by showcasing its potential in a finance application. Our goal is to provide a numerical tool that enables practitioners to utilize information from multiple data types in a theoretically justified way and that allows for straightforward extensions to other data types. We build on Collins *et al.* (2001), modeling different data types using data type-specific distributional assumptions within the exponential family of distributions, and thus extend classical factor extraction beyond its (often employed) Gaussian motivation by embedding it into a regression framework built around generalized linear models [GLM] instead of ordinary least squares [OLS]. The algorithm iteratively updates factors and loadings in an alternating fashion (see, among others, Breitung and Eickmeier (2016) for an earlier alternating scheme using OLS) until convergence, combining flexibility with computational efficiency. This construction allows each data type to be modeled according to an appropriate distribution while maintaining a common latent factor representation via the natural parameters.

To ensure robustness and scalability, we introduce three complementary variants: a Newton method-based scheme that leverages second-order information; a gradient descent version that reduces computational costs by avoiding repeated matrix inversions; and a hybrid variant that performs a single Newton refinement after gradient descent has terminated. This hybrid method proves particularly effective, preserving most gains in terms of computational costs from gradient descent while benefiting from accuracy improvements provided by Newton updates. All these steps are conducted as linear regressions. Not least, we discuss how to properly select the number of factors to extract.

We proceed as follows. In section 2, we specify our framework and the assumptions that come with it. Subsequently, we describe in section 3, how our framework extracts factors when

the dataset contains multiple data types. This approach is then put to a test in an extensive Monte Carlo simulation study, see section 4. Afterwards, section 5 demonstrates the empirical applicability via an exploratory analysis of a dataset combining continuous equity premium predictors from Goyal *et al.* (2024), technical indicators from Neely *et al.* (2014), and count variables such as monthly IPO numbers. Finally, section 6 concludes. Additional results concerning our simulation study and empirical application are provided in an online supplementary appendix.

2 The Mixed Data Model

Our mixed-type data set consists of N observed variables observed over T periods: n_1 continuous variables, gathered in $\mathbf{x}_{n_1,t}$, n_2 count variables $\mathbf{x}_{n_2,t}$, and n_3 dichotomous ones, $\mathbf{x}_{n_3,t}$. They are all influenced by r latent factors $\mathbf{F}_t$, where $r \ll N$. The impact of the factors is captured by the loading matrix $\mathbf{\Lambda}$, which is partitioned as

$$\mathbf{\Lambda} = \begin{pmatrix} \mathbf{\Lambda}_{n_1} \\ \mathbf{\Lambda}_{n_2} \\ \mathbf{\Lambda}_{n_3} \end{pmatrix},$$

corresponding to the three types of data we consider. Here, $\mathbf{\Lambda}_{n_i}$ is of dimension $n_i \times r$ and denotes the partition of the loading matrix belonging to exactly one of the aforementioned data types, the latent factors only control the time-varying expectations of the observable variables. In specific, $\mathbf{\Lambda}_{n_1}$ corresponds to continuous variables, s.t.

$$\begin{aligned} \mathbf{x}_{n_1,t} &= \mathbf{\Lambda}_{n_1}\mathbf{F}_t + \mathbf{e}_t \\ \mathrm{E}\left[\mathbf{x}_{n_1,t}|\mathbf{F}_t\right] &= \mathbf{\Lambda}_{n_1}\mathbf{F}_t. \end{aligned} \tag{1}$$

Here, an additional idiosyncratic error $\mathbf{e}_t$ obscures the effect of the factors, and the distribution of our observable continuous variables depends on the distribution of the zero-mean variables $\mathbf{e}$ as well.

Turning to observable count variables, we model them by means of Poisson regressions. Again, the factors just impact the conditional expectation where the required non-negativity of the modeled mean is ensured by using the natural logarithm as the so-called link function. Thus, each variable $i = n_1 + 1, \ldots, n_1 + n_2$ satisfies

$$\begin{aligned} x_{i,t}|\mathbf{F}_t &\sim Poisson(\tilde{\mu}_{i,t}) \\ \mathrm{E}\left[x_{i,t}|\mathbf{F}_t\right] &= e^{\boldsymbol{\lambda}_i\mathbf{F}_t} = \tilde{\mu}_{i,t} \\ \Pr\left[x_{i,t} = x|\mathbf{F}_t\right] &= \frac{\exp\left[\boldsymbol{\lambda}_i\mathbf{F}_t\right]^x e^{-\exp[\boldsymbol{\lambda}_i\mathbf{F}_t]}}{x!} \end{aligned} \tag{2}$$

where $\boldsymbol{\lambda}_i$ represents the i-th row of $\boldsymbol{\Lambda}_{n_2}$. Stacking the conditional expectations across i then yields $\tilde{\boldsymbol{\mu}}_t = \mathrm{E}\left[\mathbf{x}_{n_2,t}|\mathbf{F}_t\right] = \exp\left[\boldsymbol{\Lambda}_{n_2}\mathbf{F}_t\right]$ which can be solved for $\boldsymbol{\Lambda}_{n_2}\mathbf{F}_t = \ln\left(\mathrm{E}\left[\mathbf{x}_{n_2,t}|\mathbf{F}_t\right]\right)$ (Murphy, 2012, pp. 290 ff.). At this point, note there are no explicit idiosyncratic disturbances in (2) because the realizations stem from individual Poisson distributions (parameterized by the latent factor structure modeling the mean), which automatically adds idiosyncratic variation.

Finally, the dichotomous variables are modeled along the same lines as the count variables, but using the classical Bernoulli model for binary data. In combination with the logistic function serving as link function, observable dichotomous variables $\mathbf{x}_i, i = n_1 + n_2 + 1, \ldots, N$ satisfy

$$\begin{aligned} x_{i,t}|\mathbf{F}_t &\sim Bernoulli(\breve{\mu}_{i,t}) \\ \mathrm{E}\left[x_{i,t}\,|\,\mathbf{F}_t\right] &= 1/\left(1+\exp\left[-\boldsymbol{\lambda}_i\mathbf{F}_t\right]\right) = \breve{\mu}_{i,t} \\ \Pr\left[x_{i,t}=x\,|\,\mathbf{F}_t\right] &= \breve{\mu}_{i,t}^{x}\left(1-\breve{\mu}_{i,t}\right)^{1-x} = \begin{cases} \breve{\mu}_i, & \text{if } x_{i,t}=1 \\ 1-\breve{\mu}_i, & \text{if } x_{i,t}=0 \end{cases}. \end{aligned} \tag{3}$$

Consequently, stacking the mean representations for all i then yields $\breve{\boldsymbol{\mu}}_t = 1/\left(1+\exp\left[-\boldsymbol{\Lambda}_{n_3}\mathbf{F}_t\right]\right)$ which can be solved for $\boldsymbol{\Lambda}_{n_3}\mathbf{F}_t = \ln\left(\breve{\boldsymbol{\mu}}_t \oslash \left(1-\breve{\boldsymbol{\mu}}_t\right)\right)$ where $\oslash$ denotes element-wise division (Hastie *et al.*, 2009, pp. 119 ff.).

3 Factor Extraction Algorithm for Mixed Data Types

3.1 Background

To get started with our approach, we first review the well-understood case of continuous variables. Given only continuous data, the goal of factor extraction is to choose $\hat{\boldsymbol{\Lambda}}$ and $\hat{\mathbf{F}}_t$ such that a k-dimensional representation of the observable data $x_{it},\ i = 1, \ldots, n_1,\ t = 1, \ldots, T$, where $k \ll n_1$, minimizes the reconstruction error. Assuming the factor number r is known and setting $k = r$ lets us defer the practical choice of k until after presenting the factor extraction framework. In more detail, the minimization problem now reads as

$$V(k) = \min_{\boldsymbol{\Lambda}_{n_1}^k,\,\mathbf{F}^k} \frac{1}{n_1}\sum_{i=1}^{n_1}\sum_{t=1}^{T}\left(x_{i,t} - \boldsymbol{\lambda}_i^k\mathbf{F}_t^k\right)^2 \tag{4}$$

where the superscript k denoting the number of extracted factors will be suppressed from now on to simplify the notation. At this point, a closer inspection of the quadratic target function $V(k)$ reveals that the estimation problem can be interpreted as a regression problem for each $i = 1, \ldots, n_1$ where $\boldsymbol{\lambda}_i$ denotes the vector of coefficients. But there is a crucial difference from the standard regression case: not only the parameters are unknown, but so are the regressors. To solve (up to a rotation) the resulting identification problem, an additional constraint is imposed,

typically

$$\mathbf{\Lambda}_{n_1}'\mathbf{\Lambda}_{n_1}/n_1 = \mathbf{I}_k \tag{5}$$

with $\mathbf{I}_k$ denoting the identity matrix of dimension k. Afterwards, factors and loadings can be extracted jointly by translating the minimization problem (4) into an eigenvalue problem that can be solved in one step using PCA (Bai and Ng, 2002, pp. 197 f.), e.g. by using a singular value decomposition [SVD] of the standardized data matrix. This is in fact the standard way to solve the minimum problem in (4), which however requires continuous variables to begin with.

Alternatively, $V(k)$ could be minimized using iterative optimization methods based on gradient descent. A particular block-wise implementation is motivated by the observation that the loadings (or the factors) could be extracted by regressing the observable variables on the factors (or the loadings), should the latter be (hypothetically) known. Maintaining the assumption that there are only continuous observable variables and assuming for the sake of the argument that $\mathbf{F}_t$ were known, the following target function must be minimized to obtain an estimate of the loading row $\boldsymbol{\lambda}_i$

$$\ell_i\left(\boldsymbol{\lambda}_i; \mathbf{F}, \mathbf{x}_i\right) = \sum_{t=1}^{T}\left(x_{i,t}^2 - 2x_{i,t}\boldsymbol{\lambda}_i\mathbf{F}_t + \mathbf{F}_t'\boldsymbol{\lambda}_i'\boldsymbol{\lambda}_i\mathbf{F}_t\right), \quad i = 1, \ldots, n_1 \tag{6}$$

resulting in the gradient and the Hessian

$$S\left(\boldsymbol{\lambda}_i\right) = -2\sum_{t=1}^{T}\left(x_{i,t} - \boldsymbol{\lambda}_i\mathbf{F}_t\right)\mathbf{F}_t', \qquad H\left(\boldsymbol{\lambda}_i\right) = 2\sum_{t=1}^{T}\mathbf{F}_t\mathbf{F}_t'. \tag{7}$$

Now, let $\mathbf{X}_{n_i}$ denote the $n_i \times T$ matrix of observations stacking the $1 \times T$ rows $\mathbf{x}_i$ and let $\mathbf{F}$ denote the $k \times T$ matrix of $k \times 1$ factor vectors $\mathbf{F}_t$. Stacking and solving the first order conditions then yields

$$\hat{\mathbf{\Lambda}}_{n_1} = \mathbf{X}_{n_1}\mathbf{F}'\left(\mathbf{F}\mathbf{F}'\right)^{-1}, \tag{8}$$

i.e. $\hat{\mathbf{\Lambda}}_{n_1}$ simply stacks OLS estimates of the loadings when $\mathbf{F}$ is known.

Alternatively, the estimator in (8) can be written as an update in the style of Newton–Raphson. Although this representation is trivial for continuous variables, it will be quite useful later on. This is because the Newton-Raphson update mirrors the structure of the optimization problem when dealing with generalized linear models used in the following for discrete data. The resulting update expression reads as

$$\hat{\boldsymbol{\lambda}}_i = \boldsymbol{\lambda}_i - S\left(\boldsymbol{\lambda}_i\right) \cdot \left(H\left(\boldsymbol{\lambda}_i\right)\right)^{-1} = \boldsymbol{\lambda}_i + \left(\mathbf{x}_i - \boldsymbol{\lambda}_i\mathbf{F}\right)\mathbf{F}'\left(\mathbf{F}\mathbf{F}'\right)^{-1} \tag{9}$$

for each $i = 1, \ldots, n_1$.

If, on the other hand, the loadings were known, estimates could be obtained straightforwardly via

$$\hat{\mathbf{F}}_t = \left(\mathbf{\Lambda}_{n_1}'\mathbf{\Lambda}_{n_1}\right)^{-1}\mathbf{\Lambda}_{n_1}'\mathbf{x}_{n_1,t} = \mathbf{\Lambda}_{n_1}'\mathbf{x}_{n_1,t}/n_1, \quad t = 1,\ldots,T. \tag{10}$$

Here, the last equality is satisfied due to the aforementioned normalization constraint put on $\mathbf{\Lambda}$ and (10) gives rise to a second updating equation in the fashion of (9).

The fact that estimates of both of our two unknowns, $\mathbf{\Lambda}$ and $\mathbf{F}$, can be obtained via one of the aforementioned regression approaches (as long as the other item is known) motivates our iterative approach to extract factors and loadings by running these last two regressions in an alternating fashion; cf., *inter alia*, Breitung and Eickmeier (2016) or He *et al.* (2023) for earlier examples of alternating least squares-based approaches to factor models. Assuming that suitable initial estimates $\hat{\mathbf{\Lambda}}^{(s)}$ and $\hat{\mathbf{F}}^{(s)}$ of both $\mathbf{\Lambda}$ and $\mathbf{F}$ are available, updated estimates can then be obtained by alternating equations (8) and (10) after having replaced $\boldsymbol{\lambda}$ and $\mathbf{F}$ by $\hat{\boldsymbol{\lambda}}_i^{(s)}$ and $\hat{\mathbf{F}}^{(s)}$. This can be seen as an iterative "block" Gauss-Seidel approach. It should be pointed out that such alternating optimization is numerically inefficient compared to SVD-based PCA, however SVD is only meaningful for continuous data.

Subsequently, the updated loadings are orthonormalized to satisfy the constraint in (5). One approach to impose (5) is to apply a compact singular value decomposition to $\hat{\mathbf{\Lambda}}_{n_1}^{(s+1)} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}'$ before choosing $\hat{\mathbf{\Lambda}}_{n_1}^{(s+1)} = \sqrt{n_1}\mathbf{U}$ which satisfies the constraint. This then requires an adjustment of $\hat{\mathbf{F}}_{n_1}^{(s)} = \mathbf{\Sigma}\mathbf{V}'\hat{\mathbf{F}}_{n_1}^{(s)}/\sqrt{n_1}$ to preserve the product $\hat{\mathbf{\Lambda}}^{(s+1)}\mathbf{F}_{n_1}^{(s)}$. Afterwards, the resulting $\hat{\mathbf{\Lambda}}_{n_1}^{(s+1)}$ and $\hat{\mathbf{F}}_{n_1}^{(s)}$ replace $\mathbf{\Lambda}_{n_1}$ (and $\mathbf{F}$ in the case of a Newton-Raphson update equation) in (10) to obtain new factor estimates $\mathbf{F}^{(s+1)}$. All new estimates are subsequently used to start another update cycle and such update cycles are repeated until the fit no longer improves considerably.[1,2]

3.2 Adding Binary and Count Data

Continuous data is not the only data type we want to work with. For the loadings of count variables, $\boldsymbol{\lambda}_{n_2,i}$, the target function changes from the quadratic loss in (6) to the negative log likelihood of a Poisson model with log link function. Hence,

$$\ell_i\left(\boldsymbol{\lambda}_i;\mathbf{F},\mathbf{x}_i\right) = -\sum_{t=1}^{T}\left(x_{i,t}\boldsymbol{\lambda}_i\mathbf{F}_t - e^{\boldsymbol{\lambda}_i\mathbf{F}_t} - \ln\left(x_{i,t}!\right)\right), \quad i = n_1+1,\ldots,n_1+n_2 \tag{11}$$

[1]Preliminary simulations indicated that it is beneficial to enforce the normalization condition in every update step, as this reduces the risk of overfitting to one particular observable variable whose loadings might be very large.

[2]Instead of squeezing an orthonormalization in between the alternating updates (8) and (10), one could also use a Lagrange optimization-based update of the loadings and consider the identification restriction in (5) as the constraint; see He *et al.* (2023). This appears to be more involved, though, and we do not further pursue this avenue.

has to be separately minimized for each i. Consequently, the score and Hessian evaluated at the current state of the loading and factor estimates are

$$S(\boldsymbol{\lambda}_i) = -\sum_{t=1}^{T} \left(x_{i,t} - e^{\boldsymbol{\lambda}_i^{(s)} \mathbf{F}_t^{(s)}} \right) \left(\mathbf{F}_t^{(s)} \right)', \qquad H(\boldsymbol{\lambda}_i) = \sum_{t=1}^{T} e^{\boldsymbol{\lambda}_i^{(s)} \mathbf{F}_t^{(s)}} \cdot \mathbf{F}_t^{(s)} \left(\mathbf{F}_t^{(s)} \right)'.$$

These nonlinear equations, unlike (7) in the continuous case, cannot be solved analytically even when either $\mathbf{F}_t$ or $\boldsymbol{\lambda}_i$ are given. The natural choice in this context is the afore-mentioned Newton-Raphson algorithm, which yields row-wise updates of $\hat{\boldsymbol{\Lambda}}_{n_2}$ via

$$\hat{\boldsymbol{\lambda}}_i^{(s+1)} = \hat{\boldsymbol{\lambda}}_i^{(s)} + \left(\mathbf{x}_i - \tilde{\boldsymbol{\mu}}_i^{(s)} \right) \left(\hat{\mathbf{F}}^{(s)} \right)' \left(\hat{\mathbf{F}}^{(s)} \operatorname{diag} \left(\tilde{\boldsymbol{\mu}}_i^{(s)} \right) \left(\hat{\mathbf{F}}^{(s)} \right)' \right)^{-1} \tag{12}$$

where $\tilde{\boldsymbol{\mu}}_i^{(s)} = \exp \left[\hat{\boldsymbol{\lambda}}_i^{(s)} \hat{\mathbf{F}}^{(s)} \right]$ (Murphy, 2012, pp. 291 ff.). In standard Poisson regression setups (where $\mathbf{F}_t$ are observed), the Newton-Raphson step in equation (12) would be repeated until $\hat{\boldsymbol{\lambda}}_i$ no longer changes substantially from one iteration to the next. In our context, this would overfit $\hat{\boldsymbol{\lambda}}_i$ to the current factor estimate, which is not yet accurate either. For this reason, we perform only one update step before moving on to updating the factors given the newly improved estimates of the loadings.

Remark 1 A convenient side effect of using a Poisson regression framework is that a violation of the Poisson assumption, *e.g.* the property that the mean equals the variance, just reduces efficiency as long as the mean is correctly specified. Hence, a negative binomial or a zero-inflated Poisson model can be more efficient, but default Poisson-based estimates remain unbiased (Cohn *et al.*, 2022, Takeaways 8 and 12). □

Finally, replacing the Poisson log likelihood by the Bernoulli one suffices to add dichotomous variables as a third data type. This results in the negative log likelihood

$$\ell_i \left(\boldsymbol{\lambda}_i; \mathbf{F}, \mathbf{x}_i \right) = -\sum_{t=1}^{T} \left(\ln \left(\breve{\mu}_{i,t} \right) x_{i,t} + \ln \left(1 - \breve{\mu}_{i,t} \right) \left(1 - x_{i,t} \right) \right), \quad i = n_1 + n_2 + 1, \ldots, N \tag{13}$$

where $\breve{\mu}_{i,t} = 1/\left(1 + e^{-\boldsymbol{\lambda}_i \mathbf{F}_t} \right)$ due to using the logit link function. Consequently, the gradient and the Hessian change to

$$S(\boldsymbol{\lambda}_i) = -\sum_{t=1}^{T} \left(x_{i,t} - \breve{\mu}_{i,t}^{(s)} \right) \left(\mathbf{F}_t^{(s)} \right)', \qquad H(\boldsymbol{\lambda}_i) = \sum_{t=1}^{T} \breve{\mu}_{i,t}^{(s)} \left(1 - \breve{\mu}_{i,t}^{(s)} \right) \cdot \mathbf{F}_t^{(s)} \left(\mathbf{F}_t^{(s)} \right)'$$

leading to a third set of Newton-Raphson loading updates

$$\hat{\boldsymbol{\lambda}}_i^{(s+1)} = \hat{\boldsymbol{\lambda}}_i^{(s)} + \left(\mathbf{x}_i - \breve{\boldsymbol{\mu}}_i^{(s)} \right) \left(\hat{\mathbf{F}}^{(s)} \right)' \left(\hat{\mathbf{F}}^{(s)} \operatorname{diag} \left(\breve{\boldsymbol{\mu}}_i^{(s)} \odot \left(1 - \breve{\boldsymbol{\mu}}_i^{(s)} \right) \right) \left(\hat{\mathbf{F}}^{(s)} \right)' \right)^{-1} \tag{14}$$

where $\odot$ denotes the Hadamard product (Hastie *et al.*, 2009, pp. 120 f.).

These newly obtained estimates $\hat{\mathbf{\Lambda}}^{(s+1)}$ are then used to update the factors via another Newton-Raphson-type update, applied separately at each time t,

$$\begin{aligned}\hat{\mathbf{F}}_t^{(s+1)} &= \hat{\mathbf{F}}_t^{(s)} + \left(\left(\hat{\mathbf{\Lambda}}^{(s+1)}\right)' \mathbf{W}_t^{(s+1/2)} \hat{\mathbf{\Lambda}}^{(s+1)}\right)^{-1} \cdot \left(\hat{\mathbf{\Lambda}}^{(s+1)}\right)' \begin{pmatrix} 2\left(\mathbf{x}_{n_1,t} - \hat{\mathbf{\Lambda}}_{n_1}^{(s+1)} \hat{\mathbf{F}}_t^{(s)}\right) \\ \mathbf{x}_{n_2,t} - \tilde{\boldsymbol{\mu}}_t^{(s+1/2)} \\ \mathbf{x}_{n_3,t} - \breve{\boldsymbol{\mu}}_t^{(s+1/2)} \end{pmatrix}, \\ \mathbf{W}_t^{(s+1/2)} &= \operatorname{diag}\left(\left[2\boldsymbol{\iota}_{n_1}' : \left(\tilde{\boldsymbol{\mu}}_t^{(s+1/2)}\right)' : \left(\breve{\boldsymbol{\mu}}_t^{(s+1/2)} \odot \left(1 - \breve{\boldsymbol{\mu}}_t^{(s+1/2)}\right)\right)'\right]\right)\end{aligned} \tag{15}$$

where $\tilde{\boldsymbol{\mu}}_t^{(s+1/2)} = \exp\left[\hat{\mathbf{\Lambda}}_{n_2}^{(s+1)} \hat{\mathbf{F}}_t^{(s)}\right]$, $\breve{\boldsymbol{\mu}}_t^{(s+1/2)} = \left(1 + \exp\left[-\hat{\mathbf{\Lambda}}_{n_3}^{(s+1)} \hat{\mathbf{F}}_t^{(s)}\right]\right)^{-1}$, and $\boldsymbol{\iota}_{n_1}$ denotes an $n_1 \times 1$ vector of 1s.

However, the algorithm is still incomplete. We must address non-zero unconditional means, their effect on orthonormalizing the estimated parameters, how to initialize the procedure, and when to stop updating the parameter estimates.

3.3 Intercept and Normalization Condition

Starting with the intercept and ortho-normalization issue, preliminary simulations highlighted the importance of explicitly including an intercept in the Poisson/Logit regression steps to enable factor extraction. Unlike continuous variables, which, as is common in the factor model literature, are standardized before factor extraction, the binary and count variables cannot be centered meaningfully. If one did nonetheless, the data would lose its non-negativity and/or binarity. Consequently, the non-zero means of some observable variables need to be captured by an intercept to maintain 0-centricity of the factors.

If an intercept was left out while the unconditional mean of some count (dichotomous) variables was unequal to $1 = e^0$ ($0.5 = 1/(1 + \exp[0])$), the factor extraction would become numerically unstable as the 0-centered factors cannot adequately capture unconditional non-zero means. As can be seen in Figure 1, the algorithm occasionally compensates for the numerical instability by extracting a factor that fluctuates around 1 or -1. This shows that, although there was no explicit intercept, the algorithm still extracted an intercept-like factor; cf. $\hat{F}_1$ in Figure 1. However, this behavior is not guaranteed and effectively reduces the number of non-intercept factors from k to $k - 1$. In addition, the intercept-like values $\hat{F}_1$ were estimated. As these parameters would otherwise have been known, leaving out an intercept creates an additional T parameters that increase computational costs and the risk of numerical instability. Therefore, we strongly recommend including an intercept directly.

This recommendation affects the choice of the orthonormalization constraint. While enforcing either $\mathbf{\Lambda}'\mathbf{\Lambda}/N = \mathbf{I}_k$ or

$$\mathbf{F}\mathbf{F}'/T = \mathbf{I}_k \tag{16}$$

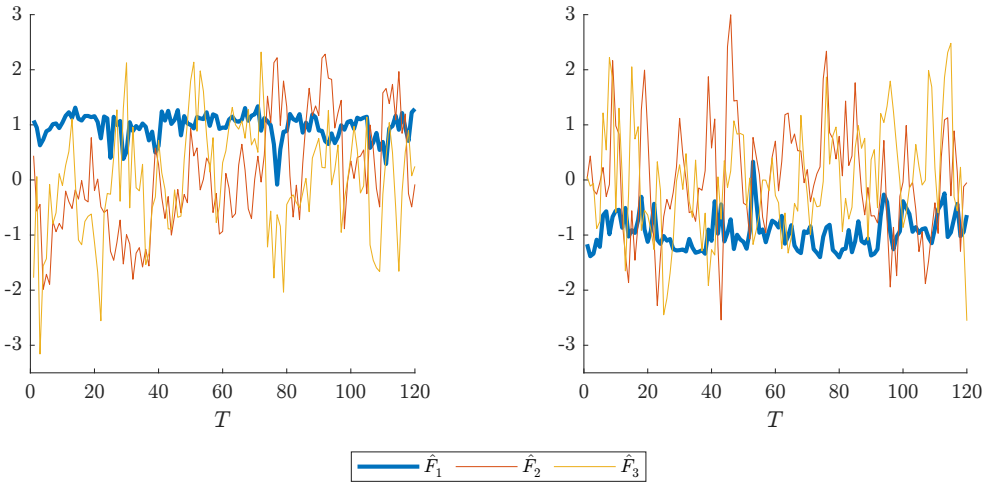

Figure 1: Two examples of factor extractions where the simulated data satisfied $r = 3$, $T = 120$, $N = 120$, and $n_i = 40$ for $i = 1, \ldots, 3$. $k = 3$ factors have been extracted using the Newton-Raphson extraction variant without including an intercept while using equation (16) as the normalization condition enforced on the estimated factors $\hat{\mathbf{F}}$.

is equivalent for continuous data, one has to be careful once an intercept is included. Because the first column of $\hat{\mathbf{\Lambda}}$ contains regression intercepts (since the first entry of $\hat{\mathbf{F}}_t$ is 1 to match standard regression notation), orthonormalizing this column is inappropriate and orthogonalizing it relative to the other (meaningfully orthonormalizable) columns is needlessly complicated. Therefore, we follow Lu *et al.* (2018, Sec. 2.2) and impose the orthonormality constraint in (16) to preserve the intercepts. Since the first column of $\hat{\mathbf{F}}$ is already correctly scaled and the remaining columns only need to be orthonormalized relative to the first, the identification restriction can be enforced efficiently, for instance, via a compact singular value decomposition.

Consequently, the factor update (leaving out the vector's first entry as this corresponds to the intercept fixed to 1) becomes

$$\begin{aligned}
\hat{\mathbf{F}}^{(s+1)}_{[2:k,t]} &= \hat{\mathbf{F}}^{(s)}_{[2:k,t]} + \left(\left(\hat{\mathbf{\Lambda}}^{(s)}_{[*,2:k]} \right)' \mathbf{W}^{(s)}_t \hat{\mathbf{\Lambda}}^{(s)}_{[*,2:k]} \right)^{-1} \cdot \left(\hat{\mathbf{\Lambda}}^{(s)}_{[*,2:k]} \right)' \begin{pmatrix} 2 \left(\mathbf{x}_{n_1,t} - \hat{\mathbf{\Lambda}}^{(s)}_{n_1} \hat{\mathbf{F}}^{(s)}_t \right) \\ \mathbf{x}_{n_2,t} - \tilde{\boldsymbol{\mu}}^{(s)}_t \\ \mathbf{x}_{n_3,t} - \breve{\boldsymbol{\mu}}^{(s)}_t \end{pmatrix}, \\
\mathbf{W}^{(s)}_t &= \operatorname{diag} \left(\left[2\boldsymbol{\iota}'_{n_1} : \left(\tilde{\boldsymbol{\mu}}^{(s)}_t \right)' : \left(\breve{\boldsymbol{\mu}}^{(s)}_t \odot \left(1 - \breve{\boldsymbol{\mu}}^{(s)}_t \right) \right)' \right] \right)
\end{aligned} \tag{17}$$

where $\hat{\mathbf{F}}_{[2:k,t]}$ $\left(\hat{\mathbf{\Lambda}}_{[*,2:k]} \right)$ denotes rows (columns) $2, \ldots, k$ of $\hat{\mathbf{F}}_t$ $\left(\hat{\mathbf{\Lambda}} \right)$, $\tilde{\boldsymbol{\mu}}^{(s)}_t = \exp \left[\hat{\mathbf{\Lambda}}^{(s)}_{n_2} \hat{\mathbf{F}}^{(s)}_t \right]$ and $\breve{\boldsymbol{\mu}}^{(s)}_t = \left(1 + \exp \left[-\hat{\mathbf{\Lambda}}^{(s)}_{n_3} \hat{\mathbf{F}}^{(s)}_t \right] \right)^{-1}$. Subsequent to centering and orthonormalizing $\hat{\mathbf{F}}^{(s+1)}$ in line with (16) (adjusting $\hat{\mathbf{\Lambda}}^{(s)}$ accordingly), the continuous variable loadings are then updated via

$$\hat{\mathbf{\Lambda}}^{(s+1)}_{[1:n_1,2:k]} = \hat{\mathbf{\Lambda}}^{(s)}_{[1:n_1,2:k]} + \left(\mathbf{X}_{n_1} - \hat{\mathbf{\Lambda}}^{(s)}_{n_1} \hat{\mathbf{F}}^{(s+1)} \right) \left(\hat{\mathbf{F}}^{(s+1)}_{[2:k,*]} \right)' \left(\hat{\mathbf{F}}^{(s+1)}_{[2:k,*]} \left(\hat{\mathbf{F}}^{(s+1)}_{[2:k,*]} \right)' \right)^{-1}. \tag{18}$$

Here, the intercept column of $\hat{\mathbf{\Lambda}}_{n_1}$ can be omitted because the continuous variables were standardized beforehand, so the initial intercept loadings (see section 3.4 below) do not need updating. Turning to the count variable ($i \in \{n_1 + 1, \ldots, n_1 + n_2\}$) loadings, their update reads as

$$\hat{\boldsymbol{\lambda}}_i^{(s+1)} = \hat{\boldsymbol{\lambda}}_i^{(s)} + \left(\mathbf{x}_i - \tilde{\boldsymbol{\mu}}_i^{(s+1/2)}\right)\left(\hat{\mathbf{F}}^{(s+1)}\right)' \left(\hat{\mathbf{F}}^{(s+1)} \operatorname{diag}\left(\tilde{\boldsymbol{\mu}}_i^{(s+1/2)}\right)\left(\hat{\mathbf{F}}^{(s+1)}\right)'\right)^{-1}, \tag{19}$$

while the binary variable ($i \in \{n_1 + n_2 + 1, \ldots, N\}$) loading update is

$$\begin{aligned}\hat{\boldsymbol{\lambda}}_i^{(s+1)} = \hat{\boldsymbol{\lambda}}_i^{(s)} &+ \left(\mathbf{x}_i - \breve{\boldsymbol{\mu}}_i^{(s+1/2)}\right)\left(\hat{\mathbf{F}}^{(s+1)}\right)' \\ &\cdot \left(\hat{\mathbf{F}}^{(s+1)} \operatorname{diag}\left(\breve{\boldsymbol{\mu}}_i^{(s+1/2)} \odot \left(1 - \breve{\boldsymbol{\mu}}_t^{(s+1/2)}\right)\right)\left(\hat{\mathbf{F}}^{(s+1)}\right)'\right)^{-1},\end{aligned} \tag{20}$$

where both $\tilde{\boldsymbol{\mu}}^{(s+1/2)}$ and $\breve{\boldsymbol{\mu}}^{(s+1/2)}$ already employ the updated factors.

3.4 Starting Values and Termination Conditions

Up to this point, only the intermediate steps of the algorithm have been examined. But to get it up and running, appropriate initial values and termination criteria have to be found to complete the algorithm.

Despite the fact that there are different data types in our dataset, we base the initial factors and loadings on PCA estimates obtained from *all* observable variables (standardized) regardless of their data type. This is only intended to deliver fast initialization of the algorithm. Afterwards, we append 1s as intercept entries to the factor matrix to account for the possible non-centrality of count and binary variables. Next, we initialize intercept loadings by appending log-transformed sample means (of count variables) and sample log-odds (of binary variables) to the PCA estimate of $\mathbf{\Lambda}$. In case of continuous variables, we simply append zeros, because these variables have been standardized in advance as this does not render them uninformative while improving comparability across variables. It is worth emphasizing: although this PCA approach ignores the methodological issues that led us to propose our framework in the first place, the resulting estimates just serve as starting values and can be obtained quickly while incorporating information from all variables.

Alternatively, PCA-based starting values could be built around the continuous data only, again enriched by log-transformed sample means and sample log-odds. But this might overemphasize the drivers of the continuous variables. If counts and binaries are wholly or partly influenced by other factors, that information may be missed during the first iterations and recovering it may take longer. Finally, starting from a fully random initialization can be numerically unstable according to preliminary simulations. Because the method only guaranties convergence to a local stationary point, random starts often cause non-convergence when the reachable local

optimum is too far from the true underlying factors. Therefore, we deem it appropriate (and support this claim in our subsequent simulation study) to resort to starting values that stem from PCA applied to all data.

Now, it remains to determine how and when the algorithm stops. This is complicated by the mixed nature of the data, since we must track an overall assessment of the goodness-of-fit. As our extraction framework is motivated by generalized linear models that optimize the corresponding log likelihoods, we resort to a McFadden Pseudo R^2 type of metric that allows us to aggregate convergence information for all three types of data. To be precise, we track

$$R^2_{\text{adj}} = 1 - \frac{\sum_{i=1}^{N} \ell_i \left(\hat{\boldsymbol{\lambda}}_i, \hat{\mathbf{F}}; \mathbf{x}_i \right) - k(N+T)}{\sum_{i=1}^{N} \ell_i \left(f(\bar{\mathbf{x}}_i), \boldsymbol{\iota}_T'; \mathbf{x}_i \right)} \tag{21}$$

in each iteration where $f(\bar{\mathbf{x}}_i)$ denotes the respective link function. Given these values, the algorithm terminates under the following conditions (implemented specifications are in parentheses):

1. **Negligible Changes Over Time:** Once successive R^2_{adj} changes relative to its current maximum fall below a prespecified tolerance (less than 1 %) for a prespecified number of iterations (5).

2. **Persistent Divergence:** Once R^2_{adj} falls short of its current maximum by more than a threshold level (1 %) for several consecutive iterations (5). This signals a divergence from optimality, potentially caused by weak identification in matrix factorization. In our alternating regression framework, weak identification may push factors in opposite directions until they become so large that the loadings can no longer offset them, which usually reflects overfitting.

3. **Non-Finite or Invalid Fit Values:** Once R^2_{adj} goes to infinity or becomes `NaN`, which typically occurs when N and T are small and matrix inversion becomes numerically unstable, the algorithm terminates because the fit is no longer reliable and Boolean operators may behave erratically when R^2_{adj} is not a real number.

4. **Iteration limit:** Once the number of update cycles reaches an upper limit (50).

3.5 Improvements of Computational Efficiency

Unlike the situation with continuous variables only, our algorithm cannot jointly update $\hat{\boldsymbol{\Lambda}}$ and $\hat{\mathbf{F}}$ because the Newton-Raphson step requires inverting the Hessians. With matrix inversion being already costly on its own, the count and binary components exacerbate this since each observable variable has its own weights. As a result, each loading and each time-t factor vector requires a separate matrix inversion. Updating $\hat{\boldsymbol{\Lambda}}$ and $\hat{\mathbf{F}}$ therefore reduces to `for` loops whose runtime grows with N and T.

One may avoid these matrix inversions by relying solely on the gradient using gradient descent updates instead of Newton-Raphson, *i.e.* updating some parameter vector $\boldsymbol{\beta}$ via

$$\hat{\boldsymbol{\beta}}_i^{(s+1)} = \hat{\boldsymbol{\beta}}_i^{(s)} - \eta \cdot S\left(\hat{\boldsymbol{\beta}}_i^{(s)}\right)$$

where η denotes the learning rate (Hastie *et al.*, 2009, p. 396). In contrast to the updates in equation (17), such updates no longer require variable- or time-specific matrix inversion. Hence, the factor updates can be concatenated horizontally across t, which yields

$$\hat{\mathbf{F}}_{[2:k,*]}^{(s+1)} = \hat{\mathbf{F}}_{[2:k,*]}^{(s)} + \boldsymbol{\eta}_F \odot \left(\hat{\boldsymbol{\Lambda}}_{[*,2:k]}^{(s)}\right)' \begin{pmatrix} 2\left(\mathbf{X}_{n_1} - \hat{\boldsymbol{\Lambda}}_{n_1}^{(s)}\hat{\mathbf{F}}^{(s)}\right) \\ \mathbf{X}_{n_2} - e^{\hat{\boldsymbol{\Lambda}}_{n_2}^{(s)}\hat{\mathbf{F}}^{(s)}} \\ \mathbf{X}_{n_3} - 1/\left(1 + e^{-\hat{\boldsymbol{\Lambda}}_{n_3}^{(s)}\hat{\mathbf{F}}^{(s)}}\right) \end{pmatrix}. \tag{22}$$

After similarly modifying the loading updates in equations (18), (19), and (20), they can be vertically concatenated across i, resulting in

$$\hat{\boldsymbol{\Lambda}}^{(s+1)} = \hat{\boldsymbol{\Lambda}}^{(s)} + \boldsymbol{\eta}_\Lambda \odot \left(\begin{pmatrix} 2\left(\mathbf{X}_{n_1} - \hat{\boldsymbol{\Lambda}}_{n_1}^{(s)}\hat{\mathbf{F}}^{(s+1)}\right) \\ \mathbf{X}_{n_2} - e^{\hat{\boldsymbol{\Lambda}}_{n_2}^{(s)}\hat{\mathbf{F}}^{(s+1)}} \\ \mathbf{X}_{n_3} - 1/\left(1 + e^{-\hat{\boldsymbol{\Lambda}}_{n_3}^{(s)}\hat{\mathbf{F}}^{(s+1)}}\right) \end{pmatrix} \left(\hat{\mathbf{F}}^{(s+1)}\right)' \odot \mathbf{M} \right) \tag{23}$$

where a conformable matrix $\mathbf{M}_{[i,1]} = 0$ for $i = 1, \ldots, n_1$ and 1 elsewhere, preventing updates to the continuous-variable intercepts as those are fixed at 0 since the continuous variables have been standardized *ex-ante*. Hence, both $\hat{\mathbf{F}}$ and $\hat{\boldsymbol{\Lambda}}$ can be updated computationally efficiently given some conformable learning rates $\boldsymbol{\eta}_F, \boldsymbol{\eta}_\Lambda$. Moreover, since $\hat{\mathbf{F}}\hat{\mathbf{F}}' = T\mathbf{I}_k$ by the constraint in (16), (23) can be interpreted as building on linear regression as well.

As there are still many parameters, one learning rate is insufficient. Therefore, we resort to parameter-specific learning rates using the AdaGrad framework proposed by Duchi *et al.* (2011). This learning rate technique tracks the gradient (squared element-wise) over the iterations so that $\boldsymbol{\eta}_\cdot = \eta / \left(\mathbf{G}^{(s+1)}\right)^{\circ\frac{1}{2}}$ where $\eta = 0.5$ (in our case) and $\mathbf{G}^{(s+1)} = \mathbf{G}^{(s)} + \left[S\left(\boldsymbol{\beta}^{(s)}\right)\right]^{\circ 2}$ with $\circ\frac{1}{2}$ and $\circ 2$ denoting the Hadamard root and power, respectively. Since each operation is executed element-wise, the computational complexity of finding $\boldsymbol{\eta}_\cdot$ remains limited while large updates are toned down, reducing the risk of overfitting at the cost of potentially slow learning in later iterations.

Compared to the Newton method introduced above, gradient descent may require more iterations because it neglects second-order information that the Newton-Raphson algorithm takes into account through the Hessian. Since this may partially offset the benefits of reduced computational complexity, investigating computation times relative to the number of iterations will

be the second key aspect (next to how accurately factors are extracted) of our simulation study later on.

Remark 2 Updating estimates to newly available data is another aspect that may raise computational costs, in particular when done in real time. One straightforward approach to minimize the effort is to use the current estimates as starting values once new data becomes available. Especially if updates are required in real time, *i.e.* only one new observation becomes available at each time step, it is likely that it would suffice to perform just one iteration to update the parameters given their current estimates as the additional data represents only a small addition to the overall dataset. We leave this question to future research. □

3.6 Hybrid Gradient Descent-Newton Raphson Extraction

We note that both the gradient descent and Newton-Raphson approaches have their respective advantages and disadvantages, as gradient descent tends to update much faster while Newton-Raphson steps are often more accurate. We therefore additionally consider a combination of the two updating schemes. This hybrid variant is based on the idea of improving an already consistent initial estimator by another consistent estimator outperforming the first with respect to the research question. The fundamental idea has been discussed, *inter alia*, in Kreiss (1985) (w.r.t. ARMA coefficient estimation) or Welsh and Ronchetti (2002) (w.r.t. outlier-robust regression analysis), and amounts, in our setup, to performing exactly one Newton-Raphson update cycle once the gradient descent updates are terminated; see also the summary of the extraction framework in Algorithm 1 in the appendix. Thereby, this variant preserves most of the reduction in computational costs while also benefiting from the second-order information that the Newton-Raphson step leverages.

3.7 Factor Number Selection

Finally, the implementation in both the simulation study and our empirical illustration requires choosing the factor number k. As we focus on extracting factors that describe the dataset at hand, our first proposal is the adjusted Pseudo-R^2 in (21) as it captures how well the factor model describes the behavior of the observable models compared to the simplest model possible, a single intercept factor.

In addition, we consider BIC-like information criteria of the form

$$\mathcal{L}\left(k, \hat{\mathbf{\Lambda}}^k, \hat{\mathbf{F}}^k\right) = \mathcal{P}(k)/2 - \ell\left(\hat{\mathbf{\Lambda}}^k, \hat{\mathbf{F}}^k; \mathbf{X}\right)$$

where $\ell(\cdot)$ denotes the log likelihood of the problem evaluated at the estimates and $\mathcal{P}(\cdot)$ the penalty term; cf. Zhao *et al.* (2025, Eq. 3). We apply the factor extraction-specific penalty terms of Bai and Ng (2002), (i) $k\left(\frac{N+T}{NT}\right)\ln\left(\frac{NT}{N+T}\right)$, (ii) $k\left(\frac{N+T}{NT}\right)\ln\left(\min(N,T)\right)$, (iii) $k\left(\frac{\ln(\min(N,T))}{\min(N,T)}\right)$ and

combine them with $\sum_{i=1}^{N} \ell_i \left(\hat{\boldsymbol{\lambda}}, \hat{\mathbf{F}}; \mathbf{x}_i\right) / (NT)$. Because the precise value of $\ell(\cdot)$ is of secondary importance, computational speed may be prioritized over numerical precision at this point, allowing us to obtain factor and loading estimates via gradient descent.

In the following, we compare the performance of R^2_{adj} and our three resulting BIC-type criteria, BN_1, BN_2, and BN_3, to that of the ER and GR criteria of Ahn and Horenstein (2013), which we employ to determine the number of factors in the competing PCA-based estimation (restricted to continuous variables).

4 Simulation Study

We study the performance of our three factor extraction variants, comparing them to standard PCA applied only to the continuous variables as a (not entirely) naïve benchmark. The controlled environment helps to investigate how well the extracted factors recover the space spanned by the true factors and how computationally costly each variant is. Moreover, we examine which of our factor number selection criteria works best. To simplify the notation, the gradient descent-based variant from section 3.5 will be denoted as *GD*, the Newton-Raphson-based variant from section 3.2 as *NR*, and the hybrid approach from section 3.6 as *GD-NR*.

4.1 Simulation Design

Motivated by the stock index data application that we have in mind, the latent factors are assumed to satisfy stationary AR(1) processes. Hence,

$$\mathbf{F}_t = \text{diag}(\boldsymbol{\rho}) \cdot \mathbf{F}_{t-1} + \mathbf{w}_t, \quad \boldsymbol{\Psi}(L)\mathbf{w}_t = \mathbf{v}_t \tag{24}$$

where $\mathbf{F}_t$ is a $r \times 1$ vector, $\text{diag}(\boldsymbol{\rho}) = \text{diag}\left(\rho_1, \ldots, \rho_r\right)$, and ρ_i is fixed and bounded away from 1, *i.e.* $|\rho| < 1$. Furthermore, let $\boldsymbol{\Psi}(L)$ be a stable finite-order VAR(p) model that allows for mild degrees of time and cross-sectional dependence of the innovations $\mathbf{v}_t$.

In general, we simulate the data according to the data-generating process [DGP] in equations (1)–(3), varying the sample size in both N and T. Throughout all subsequent DGPs, we consider $n_i = \{20, 40, 80\}$ for all three data types, *i.e.,* in total $N = \{60, 120, 240\}$ observable variables and $T = \{60, 120, 240\}$ observations in time.

Then, all $r = 4$ factors follow stationary AR(1) processes, cf. (24), with coefficients drawn at random from $\mathcal{U}(0, 0.8)$ throughout simulation designs. Turning to the intercepts and loadings, we assume, without loss of generality, that the continuous variables have mean zero. The intercepts for the count variables are drawn from $\mathcal{U}(2, 8)$, and those for the binary variables are drawn from $\mathcal{U}(0.2, 0.8)$. Then, the loadings of the continuous variables are i.i.d. $\mathcal{N}(0.5, 1)$, while the loadings for both count and binary variables are i.i.d. $\mathcal{U}(-0.5, 1.5)$-distributed, which reduces the number

of extreme mean values induced by the link functions. In addition, the explicit idiosyncratic errors $\mathbf{e}$ in (1) are i.i.d. multivariate standard normally distributed.

In particular, our simulation study investigates the following DGPs:

DGP1.1 (baseline model): We consider the case where the innovation vector $\mathbf{w}_t$ in (24) is generated as an i.i.d. sequence of multivariate normally-distributed random vectors with mean zero vector and covariance matrix

$$\Sigma = \begin{pmatrix} 0.25 & \phi & 0 \\ \phi & 0.25 & \vdots \\ 0 & \cdots & \ddots \end{pmatrix}$$

as our homoskedastic baseline DGP setting $\phi = 0$.

DGP1.2 (Continuous variable-only starting values): Next, we assess the importance of using all data types to initialize the estimates by simulating data from DGP1.1. This time, however, we obtain starting values by performing PCA only on the observed continuous variables. The resulting initial factor scores are then combined with the following initialization of the loading matrix,

$$\hat{\mathbf{\Lambda}}^{(0)} = \begin{pmatrix} \mathbf{0} & \hat{\mathbf{\Lambda}}_{n_1} \\ \ln\left(\bar{\mathbf{X}}_{n_2}\right) & \mathbf{0} \\ -\ln\left(\left(1 - \bar{\mathbf{X}}_{n_3}\right) \oslash \bar{\mathbf{X}}_{n_3}\right) & \mathbf{0} \end{pmatrix},$$

constructed from the estimated continuous variable-only PCA loadings, together with unconditional mean estimates for the count and binary observed variables.

DGP2 (conditional heteroskedasticity): In the second DGP, we extend the baseline setup assuming that $\mathbf{w}_t$ follows a GARCH(1, 1) process. We keep the unconditional correlation structure of DGP1, and re-scale the innovations using standard deviations obtained from the variance process

$$\sigma_t^2 = (1 - a_1 - b_1) + a_1 \varepsilon_{t-1}^2 + b_1 \sigma_{t-1}^2.$$

Since $\mathrm{E}\left[\varepsilon_t^2\right] = \frac{(1-a_1-b_1)}{1-a_1-b_1} = 1$ if $a_1, b_1 \geq 0$ and $a_1 + b_1 < 1$, our choice of $a_1 = 0.1, b_1 = 0.85$ results in a stationary GARCH(1,1) process where $\mathrm{E}\left[w_t^2\right] = 0.25$ as before. Otherwise, the setup from DGP1.1 remains unchanged.

DGP3 (Non-AR factors): In DGP3, we consider a cross-sectional setup instead of a time series setup, setting the AR(1) coefficients to 0 while keeping the parameterization from DGP1.1, otherwise.

DGP4.1 (Sparse loadings): In DGP4, the loading matrix differs. While retaining the distributional assumptions regarding loadings and intercepts, we assume that the first factor is the only one that influences all observable variables (on average). Subsequently, the second factor impacts the continuous variables, whereas the third influences the count, and the fourth the binary variables with the other loadings set to 0 accordingly. Other than that, we keep the baseline setup from DGP1.1 and extract starting values from all observable variables.

DGP4.2 (Sparse loadings and starting value choice): To deepen the understanding of our starting value choice, we simulate data as in DGP4.1. However, we base our starting values on continuous variable-only PCA estimates and unconditional sample averages as in DGP1.2, now.

DGP4.3 (Sparse loadings when r is known): To disentangle the impact of using a partitioned loading matrix on estimating r from its impact on factor estimation, we repeat DGP4.1 while fixing $k = r = 4$.

DGP5 (Over-dispersed counts): Next, DGP5 examines a violation of our modeling assumptions by considering over-dispersed count data. Specifically, we abandon the Poisson assumption in the data-generating process and instead assume that counts follow a negative binomial distribution with an overdispersion parameter drawn from $\mathcal{U}(0, 1)$. However, we still treat the resulting observations as if they were Poisson. This setup allows us to demonstrate the robustness of our algorithm under mild forms of model misspecification.

DGP6 (Imbalanced panel): Finally, DGP6 investigates an imbalanced design by considering variable sets of size $N = 140$ where $\mathbf{n} = (n_1, n_2, n_3) = (20, 40, 80)$, $\mathbf{n} = (40, 80, 20)$ or $\mathbf{n} = (80, 20, 40)$ in place of the equal-sized (across data types) sets that have been used so far. In all other respects, it retains the parameterization of DGP1.1.

All simulations are performed in MATLAB 2024b using an *AMD Ryzen 9 7950X3D* processor and MATLAB's Threefry 4x64 random number generator. The maximum number of factors is set to 6, and the number of Monte Carlo replications for each DGP-N-T combination is 5000.

4.2 Distance Measures of Subspaces

Our exercise aims to establish how well the algorithm recovers the true factor space from the data and whether including additional variables pays off compared to using only continuous data. Knowing the true factors in the simulation study allows us to examine the difference between the two subspaces by comparing the orthonormalized column spaces of $\hat{\mathbf{F}}$ and $\mathbf{F}$; for this, we however require a suitable distance measure.

First, we adopt the following metric which compares two linear spaces spanned by the estimated factors and the original DGP, respectively, using

$$\mathcal{D}(\mathbf{Q}_1, \mathbf{Q}_2) = \left(1 - \frac{1}{\max(k,l)} \operatorname{tr}\left(\mathbf{Q}_1\mathbf{Q}_1'\mathbf{Q}_2\mathbf{Q}_2'\right)\right)^{1/2}$$

where $(\mathbf{Q}_1)_{T\times k}$ and $(\mathbf{Q}_2)_{T\times l}$ denote column-wise orthogonal matrices; see He *et al.* (2023) and the references therein. The required column-wise orthogonality can be ensured by means of a (modified) Gram-Schmidt-based orthonormalization. Now, this metric satisfies $0 \leq \mathcal{D}(\mathbf{Q}_1, \mathbf{Q}_2) \leq 1$ by construction, as it quantifies the distance between the two column spaces spanned by $\mathbf{Q}_1 = \hat{\mathbf{F}}', \mathbf{Q}_2 = \mathbf{F}'$. If $\mathcal{D}(\mathbf{Q}_1, \mathbf{Q}_2) = 0$, the spanned spaces would be the same whereas they would be orthogonal if $\mathcal{D}(\mathbf{Q}_1, \mathbf{Q}_2) = 1$.

Our second measure, proposed by Ye and Lim (2016), serves as a robustness check of the results of $\mathcal{D}$. Using principal angles between two orthonormal bases $\mathbf{Q}_1$ and $\mathbf{Q}_2$, it is similarly applicable when the two corresponding subspaces $\mathbf{Q}_1^s, \mathbf{Q}_2^s$ are of different dimensions. Let σ_i^{svd} denote the singular values of $\mathbf{Q}_1'\mathbf{Q}_2$. Then, the $\min(k,l)$ principal angles between the subspaces are readily available via $\theta_i = \cos^{-1}(\sigma_i^{\text{svd}}(\mathbf{Q}_1'\mathbf{Q}_2))$, $i = 1, \ldots, \min(k,l)$ (Zhu and Knyazev, 2013, Theorem 2.1). Subsequently, the so-called Grassmann distance between the two subspaces can be computed as

$$\mathcal{D}^{\text{gr}(\infty,\infty)}(\mathbf{Q}_1, \mathbf{Q}_2) = \frac{\left(|k-l|\pi^2/4 + \sum_{i=1}^{\min(k,l)} \theta_i^2\right)^{1/2}}{\left(\max(k,l) \cdot \pi^2/4\right)^{1/2}}.$$

In the numerator, the first summand captures the difference in dimensions, whereas the latter summand captures the difference between the two subspaces in terms of the $\min(k,l)$ available principal angles between the two subspaces. Thus, the sum can be viewed both as the distance from $\mathbf{Q}_1^s$ to the most distant k-dimensional subspace contained in $\mathbf{Q}_2^s$, and as the distance from $\mathbf{Q}_2^s$ to the most distant l-dimensional subspace containing $\mathbf{Q}_1^s$. The denominator then rescales $\mathcal{D}^{\text{gr}}(\mathbf{Q}_1, \mathbf{Q}_2)$ to lie in $[0,1]$.[3]

4.3 Discussion of Simulation Study Results

For each simulated dataset, factor extraction begins by determining the number of factors that we want to obtain. Examining the DGP1.1 results in Table 1, we find that all estimators of r improve with the sample size (both in N and/or T). Moreover, both R^2_{adj} and BN_3 tend to detect slightly too many factors, whereas ER and GR detect too few. That said, note that the PCA criteria use only one third as many observations as the other criteria because they consider only continuous variables. Hence, there are less observations to sustain a "large" number of factors, so the smaller k values are not surprising. However, with a growing number of observations, the PCA criteria also accurately recover the true number of factors. We therefore report results

[3]For more details regarding this metric, we refer the interested reader to Ye and Lim (2016, pp. 1188–1191).

N	T	R^2_{adj}	BN_1	BN_2	BN_3	ER	GR
60	60	4.27 (0.54)	3.95 (0.26)	3.85 (0.37)	4.23 (0.51)	2.56 (1.22)	2.87 (1.18)
	120	4.23 (0.52)	4.01 (0.17)	3.99 (0.18)	4.05 (0.25)	2.88 (1.22)	3.20 (1.09)
	240	4.14 (0.46)	3.99 (0.21)	3.99 (0.20)	4.00 (0.23)	3.07 (1.19)	3.39 (1.02)
120	60	4.19 (0.48)	4.00 (0.11)	3.99 (0.13)	4.03 (0.19)	3.46 (1.04)	3.71 (0.77)
	120	4.19 (0.48)	4.01 (0.12)	4.01 (0.10)	4.09 (0.33)	3.86 (0.57)	3.95 (0.34)
	240	4.13 (0.41)	4.00 (0.15)	4.00 (0.14)	4.03 (0.22)	3.95 (0.34)	3.98 (0.17)
240	60	4.12 (0.37)	4.00 (0.09)	4.00 (0.09)	4.01 (0.12)	3.95 (0.37)	3.98 (0.19)
	120	4.17 (0.43)	4.01 (0.12)	4.01 (0.11)	4.04 (0.21)	4.00 (0.04)	4.00 (0.00)
	240	4.13 (0.39)	4.01 (0.16)	4.01 (0.14)	4.07 (0.28)	4.00 (0.00)	4.00 (0.00)

Table 1: Average number of extracted factors using selection criteria R^2_{adj}, BN_1, BN_2, BN_3, ER, and GR (Standard deviations in parentheses). Data simulated from DGP1.1.

N	T	*GD*				*NR*				*GD-NR*		*PCA*
		R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	sec.	R^2_{adj}
60	60	0.446 (0.102)	15.05 (1.90)	14.94 (1.93)	0.0054 (0.0037)	0.450 (0.105)	7.01 (0.10)	5.12 (1.46)	0.0111 (0.0058)	0.449 (0.104)	0.0067 (0.0035)	0.189 (0.060)
	120	0.467 (0.089)	14.84 (1.74)	14.76 (1.77)	0.0085 (0.0025)	0.475 (0.091)	7.08 (0.28)	5.33 (1.33)	0.0167 (0.0042)	0.474 (0.091)	0.0108 (0.0030)	0.208 (0.059)
	240	0.475 (0.083)	14.78 (1.70)	14.73 (1.73)	0.0148 (0.0021)	0.485 (0.085)	7.23 (0.48)	5.61 (1.29)	0.0284 (0.0033)	0.482 (0.150)	0.0185 (0.0022)	0.219 (0.057)
120	60	0.455 (0.093)	14.85 (1.56)	14.76 (1.59)	0.0081 (0.0019)	0.461 (0.095)	7.00 (0.08)	5.36 (1.32)	0.0144 (0.0034)	0.461 (0.094)	0.0100 (0.0027)	0.242 (0.051)
	120	0.476 (0.084)	14.69 (1.49)	14.64 (1.51)	0.0152 (0.0019)	0.483 (0.086)	7.07 (0.27)	5.73 (1.20)	0.0242 (0.0027)	0.483 (0.085)	0.0183 (0.0021)	0.263 (0.041)
	240	0.487 (0.076)	14.60 (1.40)	14.58 (1.40)	0.0286 (0.0030)	0.496 (0.078)	7.18 (0.42)	6.05 (1.15)	0.0437 (0.0042)	0.465 (2.083)	0.0342 (0.0033)	0.272 (0.039)
240	60	0.459 (0.092)	14.69 (1.44)	14.63 (1.44)	0.0143 (0.0018)	0.465 (0.094)	7.00 (0.09)	5.73 (1.21)	0.0233 (0.0025)	0.464 (0.094)	0.0173 (0.0020)	0.269 (0.040)
	120	0.479 (0.078)	14.56 (1.34)	14.53 (1.34)	0.0283 (0.0029)	0.487 (0.080)	7.05 (0.23)	6.10 (1.05)	0.0398 (0.0033)	0.486 (0.080)	0.0335 (0.0032)	0.282 (0.036)
	240	0.491 (0.072)	14.58 (1.28)	14.57 (1.28)	0.0603 (0.0052)	0.501 (0.074)	7.16 (0.40)	6.39 (1.01)	0.0783 (0.0064)	0.499 (0.074)	0.0695 (0.0055)	0.292 (0.035)

Table 2: Average R^2_{adj}, total/R^2_{adj}-optimal iteration counts $\mathcal{I}/\mathcal{I}^*$, and computation times in seconds (Standard deviations in parentheses). Data simulated from DGP1.1.

where the factor numbers k are based on GR (for PCA) and BN_2 (for our extraction framework), as these balanced the risk of selecting too few or too many factors best in our simulation study.

Table 2 then shows that all methods improve on the intercept-only model with PCA doing so the least. However, the pseudo-R^2 difference cannot be interpreted well beyond the ordering of the values, which is why we focus on the two aforementioned distance measures $\mathcal{D}_F$ and $\mathcal{D}_F^{\text{gr}}$. As can be seen in Figures 2 and 6, NR extracts factor spaces that are closest to the true ones, with GD-NR performing almost as well throughout all N, T combinations. Unsurprisingly, performance improves with sample size, partially, due to the factor number being estimated more accurately, but also due to the algorithm profiting from more information; consider, for example, $T = 240$ where the fit improves in N although $k = 4$ for all N. Moreover, we note that GD performs slightly worse, but the single NR iteration in GD-NR improves the GD fit enough to essentially match NR.

After establishing the "quality" of factor extraction, we focus on computational demand. Looking at the number of iterations that it takes to extract factors, we note that GD takes longer in terms of iterations, but because this variant does not require computationally costly matrix inversions, the computational time measured in seconds is nonetheless lower. With GD-NR coming in second in terms of computational time,[4] this variant appears to balance the accuracy improvement caused by second-order information and the computational efficiency of gradient descent. Finally, note that we do not report the termination causes because all three variants almost always stop for the same reason: the fit no longer improves. Other termination causes, such as the values of R^2_{adj} becoming invalid or the upper iteration limit being reached, occur rarely and typically only when the number of extracted factors k differs substantially from r.

Turning to PCA, the estimated factor spaces are farther from the true ones than the ones our alternating algorithm variants extract, indicating that using non-continuous variables provides useful additional information; partially so because the factor number is easier to determine (cf. the upper panels in Figures 2 and 6 where the spikes stem from k being too small), but also in cases where $k = r$ (cf. the lower-right panels in Figures 2 and 6).[5]

Looking at the results of DGP1.2, neglecting the information from count and binary variables in the starting value computation does not alter the findings qualitatively given a dense loading matrix; cf. the supplementary material, section S.1, Tables S.1, S.2 and Figures S.1, S.2. The main difference is that, on average, it takes our algorithm marginally longer to terminate. However, otherwise, NR, GD-NR, and GD (with the performance of the latter differing marginally less from the ones of the former two, now) still perform as before.

[4] We do not report the numbers of iterations for GD-NR as they coincide with those of GD plus the additional NR iteration.

[5] PCA is an order of magnitude faster than our algorithm as it utilizes highly efficient singular value decomposition algorithms that are computationally much less expensive than our alternating approach.

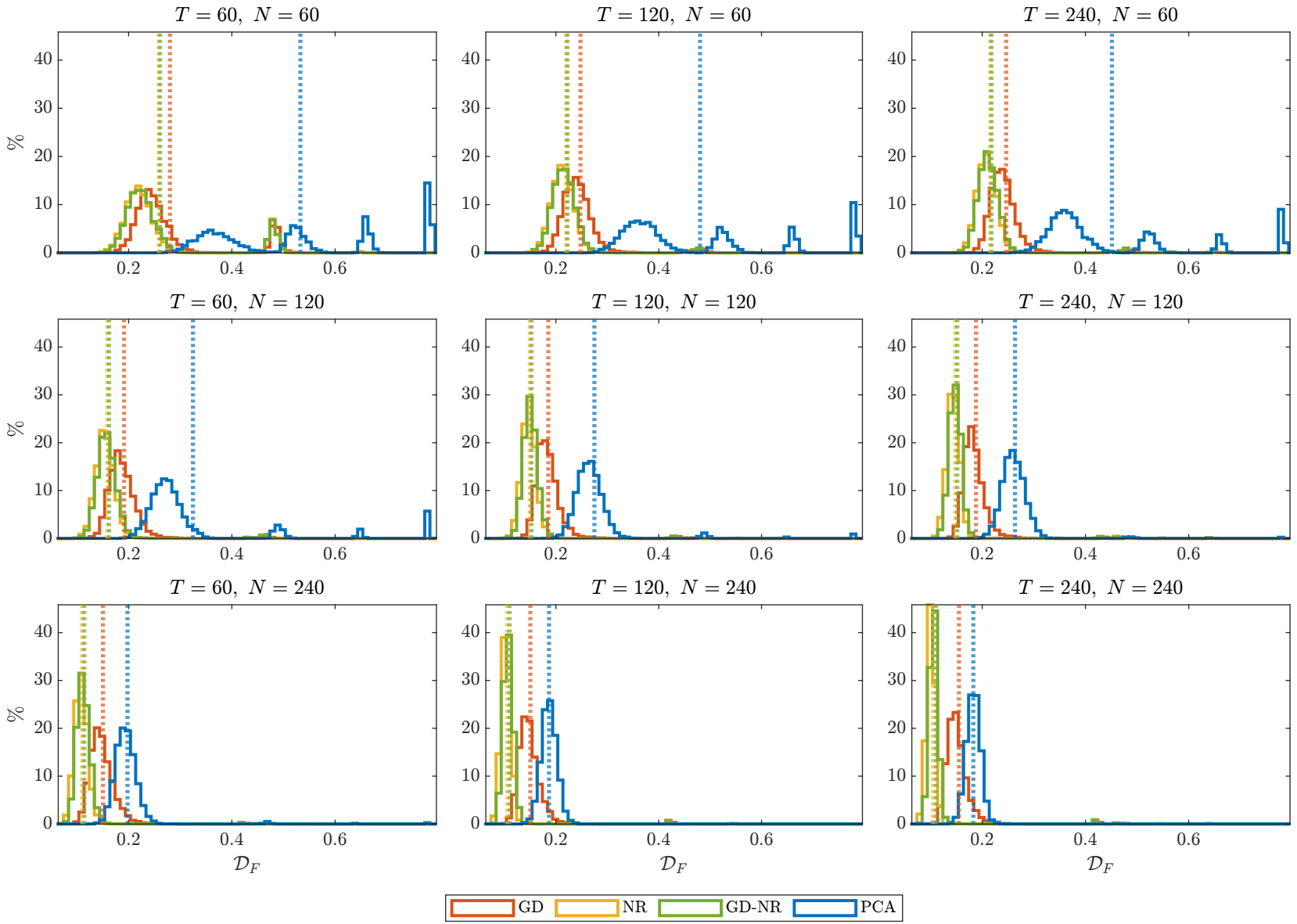

Figure 2: $\mathcal{D}_F$ histogram (average values as dotted lines). Data simulated from DGP1.1.

In DGP2, the additional variation caused by conditional heteroskedasticity reduces the accuracy of our factor number selection mechanisms. In particular, the two that we apply, namely BN_2 and GR, both indicate fewer factors for smaller samples with the effect disappearing in larger samples. However, even for $k = r$, the accuracy of the factor estimates decreases slightly relative to DGP1.1. But the qualitative results remain unchanged, nonetheless; see Tables S.3, S.4, and Figures S.3, S.4 in the supplementary appendix.

A reduced degree of dependence like in DGP3, where factors are i.i.d. instead of AR(1), causes smaller R^2_{adj} values throughout the methods, but other than that the results do not change relative to the baseline setup; see the detailed results in Tables S.5, S.6 and Figures S.5, S.6.

A partitioned loading matrix like the one in DGP4.1 substantially changes the results. As none of our variants explicitly considers this kind of sparsity, the number of factors r is incorrectly estimated most of the time; see Table S.7. This is not an issue with our algorithm, but one of not specifying sparsity correctly, which would affect other methods as well. Consequently, both distance measures $\mathcal{D}_F$ and $\mathcal{D}_F^{\text{gr}}$ indicate that the subspace spanned by the true factors is not well recovered; cf. Figures S.7 and S.8. Unfortunately, *PCA* even amplifies these issues because it is not possible to uncover factors 3 and 4 without the non-continuous data. Consequently, *PCA*'s performance also deteriorates; see Table S.8, in addition.

Naturally, neglecting non-continuous information in the starting values, as in DGP4.2, further exacerbates performance degradation, in particular for *GD*; cf. Tables S.9, S.10 and Figures S.9, S.10. It takes the algorithm longer to converge and the true factors are not recovered as precisely as in DGP4.1. This highlights the benefit of including as much information as possible in the initial values, as this helps the algorithm to start near a suitable local optimum.

The results in Figure 7 as well as in Table S.11 and Figure S.11 then show that the adverse effect of a sparse loading matrix can be partially attributed to factor number selection. Assuming that r is known and choosing $k = r$ accordingly improves both $\mathcal{D}_F$ and $\mathcal{D}_F^{\text{gr}}$ for *GD*, *NR*, and *GD-NR*. Hence, our factor extraction appears to perform satisfactorily despite not accounting for a possibly sparse loading matrix. Conversely, this is not true for *PCA* as there is only information about two factors in the continuous data. Naturally, extracting two more factors does not capture additional valuable information in this constellation.

Although fixing $k = r$ partially resolves the issue in terms of factor estimation accuracy, it comes at the cost of causing a few R^2_{adj} outliers in *GD-NR*, indicating potentially inadequate treatment of the data. Since the *GD*- and *NR*-R^2_{adj} values remain comparable to others in the respective Monte Carlo setup, the Newton-Raphson step may help to identify convergence of the algorithm to a suboptimal local optimum, be it due to the wrong number of factors being extracted or the particular dataset resulting in suboptimal performance. With *GD-NR* pseudo-R^2 tending to minus infinity and $\mathcal{D}_F, \mathcal{D}_F^{\text{gr}}$ being larger than usual in these few Monte Carlo samples, the final NR step might serve as an indicator that something is wrong with factor

extraction, which then requires the researcher to investigate the potential issue more closely in practice.

Thus, more sophisticated approaches might be necessary to deal with sparse loading structures. Two potential avenues for future research could be to adapt the two-step approach of Breitung and Eickmeier (2016) separately estimating some global and some "regional" factors (with "regional" amounting to "data type-specific" in DGP4) or to resort to shrinkage; for instance, extending our alternating framework by LASSO-type penalties.

As this is beyond the scope of this paper, we leave more sophisticated factor number selection techniques and factor extraction in sparse setups to future research. Instead, we continue our simulation study by considering the case where one of our distributional assumptions is incorrect. It turns out that the kind of misspecification DGP5 explores mainly impacts the factor number selection. Once the sample is sufficiently large and our information criteria yield $k \approx r$, $\mathcal{D}_F$ and $\mathcal{D}_F^{\text{gr}}$ exhibit estimation accuracy almost similar to what we observed in the baseline setup DGP1.1; see, for instance, the lower right panels in Figures S.12 and S.13 and the results in Tables S.12 and S.13 in the supplementary material. This observation supports Cohn *et al.* (2022) who pointed out that Poisson regression remains reliable even when the count data is not Poisson, but, for instance, negative binomially distributed.

Finally, imbalanced panels do not negatively affect the performance of our factor extraction framework. Although variables have different scales and therefore have varying impacts on factor updates and termination criterion, Tables S.14 and S.15 as well as Figures S.14 and S.15, summarizing the results obtained for DGP6, show that *NR* and *GD-NR* still outperform both *GD* and *PCA*, with the latter performing particularly poorly when the number of continuous variables available is small. Thus, each data type appears to contribute meaningfully to both the accuracy of factor number selection and factor extraction in our framework.

In summary, our three extraction variants estimate factors accurately, and including information that comes in non-continuous form pays off in terms of estimation accuracy. The algorithm even offers a certain degree of robustness to misspecification, as indicated by the results of DGPs 4 and 5. This robustness comes in handy in empirical applications where the data is certainly not as well behaved as we assumed in our baseline setup.

5 Empirical Application

We now illustrate the empirical applicability of our framework by extracting factors from a dataset of presumed equity premium predictors. As a fundamental problem in finance, the prediction of equity premia has received considerable attention over the last decades; consider, for instance, the early work of Fama (1970) and the seminal work of Campbell and Shiller (1988) or Fama and French (1988) which have been succeeded by numerous studies. As interest

continued, the list of presumed predictors kept growing; cf., for instance, the reviews of presumed predictors in Welch and Goyal (2008) and Goyal *et al.* (2024).

As the number of predictors has grown beyond what standard econometric tools can usually handle and the signal-to-noise ratios of the variables are often low, building indices from particular groups of variables in advance represents one approach to dimensional reduction; cf. the technical indicator or the illiquidity measure indices *tchi* and *lzrt*, respectively, in the dataset of Goyal *et al.* (2024). Alternatively, indices can be constructed by considering the broader context of the data and deriving factors, which then serve as indices, from the full dataset.

5.1 Data

Our dataset consists mainly of the (pseudo) out-of-sample version of the data in Goyal *et al.* (2024)[6] and we refer the interested reader to the variable definitions they provide, as well as to the references cited therein.

Since this dataset contains only continuous variables, we add 14 technical indicators from Neely *et al.* (2014). These indicators are built from monthly data to match the frequency of the continuous variables and have been repeatedly studied by, among others, Harvey *et al.* (2021). Moreover, they are part of the Goyal *et al.* (2024) dataset, but only as an index called *tchi*. Because we can now treat indicators as binary variables, we may also include the indicators directly, abbreviating the moving average indicators as MAI(s,l), the momentum indicators as MOI(s), and the on-balance volume indicators as OBV(s,l), with s,l denoting bandwidths or lags.

Next, we add count variables starting with the monthly number of US IPOs originally used in Ibbotson *et al.* (1994),[7] which exclude, among others, SPACs, direct listings, penny stocks, units, and closed-end funds. Furthermore, we include several count variables related to technical indicators. Using daily S&P 500 data taken from Yahoo Finance, we aggregate 50-(200-)day line crossings and exceedances on a monthly basis. In addition, we calculate daily Bollinger bands (20 days, $k = 2$; BB), and the Relative strength index ($n = 14$; RSI). As these items require a considerable number of observations, we resort to daily observations and aggregate them to monthly frequencies. First, the number of days where the closing price crosses the upper/lower bands is counted, and the number of days the closing price remains outside the interval. These numbers indicate bullish/bearish tendencies as well as market fluctuations. We do the same for the RSI, counting the bullish (RSI > 50) and bearish (RSI < 50) crossings, as well as the number of days the closing price remains in that interval. Moreover, the number of days the RSI grows beyond 70 (drops below 30) and how many days a month the price remains in that interval are

[6]The dataset of Goyal *et al.* (2024) is available from the website of Amit Goyal, https://sites.google.com/view/agoyal145/.

[7]The dataset is an updated version of the original data and is taken from Jay Ritter's website, https://site.warrington.ufl.edu/ritter/ipo-data/.

	GD vs. *NR*	*GD* vs. *GD-NR*	*NR* vs. *GD-NR*
$\mathcal{D}_F$	0.2082	0.1429	0.2026
$\mathcal{D}_F^{\text{gr}}$	0.1349	0.0916	0.1320

Table 3: Distances $\mathcal{D}_F$ and $\mathcal{D}_F^{\text{gr}}$ between the subspaces spanned by the extracted factors. k selected by *GD* and BN_2.

supposed to give an idea of the probability of a market reversal, since an RSI > 70 (< 30) is often interpreted as signaling an overbought (oversold) market.

In total, our dataset ranges from January 1980 to December 2019 to maximize the number of continuous variables, removing all continuous variables that are not available throughout the entire period. The resulting list of variables can be found in Table S.16 in the supplementary material, section S.2, together with a short description and the respective data source for each variable. Furthermore, the table lists AR(1) coefficient estimates that have been used to determine whether a continuous variable needs to be transformed for it to be stationary. Additionally, the respective transformation type (in the notation of McCracken and Ng (2016)), and post-transformation AR(1) coefficient estimates are collected in Table S.16. Lastly, the table provides sample averages and standard deviations for all variables to get an idea about each variable's location and scale properties.

5.2 Empirical Factors

Like in the simulation exercise, we apply *GD* for $k \in \{1, 2, \ldots, 6\}$ and select the number of factors using BN_2. This procedure recommends extracting $k = 5$ factors, and we follow this choice when applying both *NR* and *GD-NR*. Afterwards, *equamax* rotations are applied to each set of extracted factors to improve their interpretability.[8] Subsequent to the rotations, the results turn out to be qualitatively quite similar across the three extraction variants, with the distance between the extracted subspaces being small; cf. Table 3. Since the only notable qualitative difference is that the signs of the extracted factors change between extraction variants, we will discuss the results of *GD-NR* and provide the remaining results in section S.2 of the supplementary appendix.

Unlike in most factor applications, not all variables are standardized, so loadings are difficult to compare to each other across variables. For this reason, we adapt the approach of Ludvigson and Ng (2011) and regress every observable variable on each extracted factor before calculating the corresponding McFadden Pseudo-R^2. This is supposed to grasp the explanatory power each factor provides towards the respective observable variable.

[8]For the sake of interpretability, we re-order the extracted factors by data type as the algorithm does not enforce a specific order of the factors.

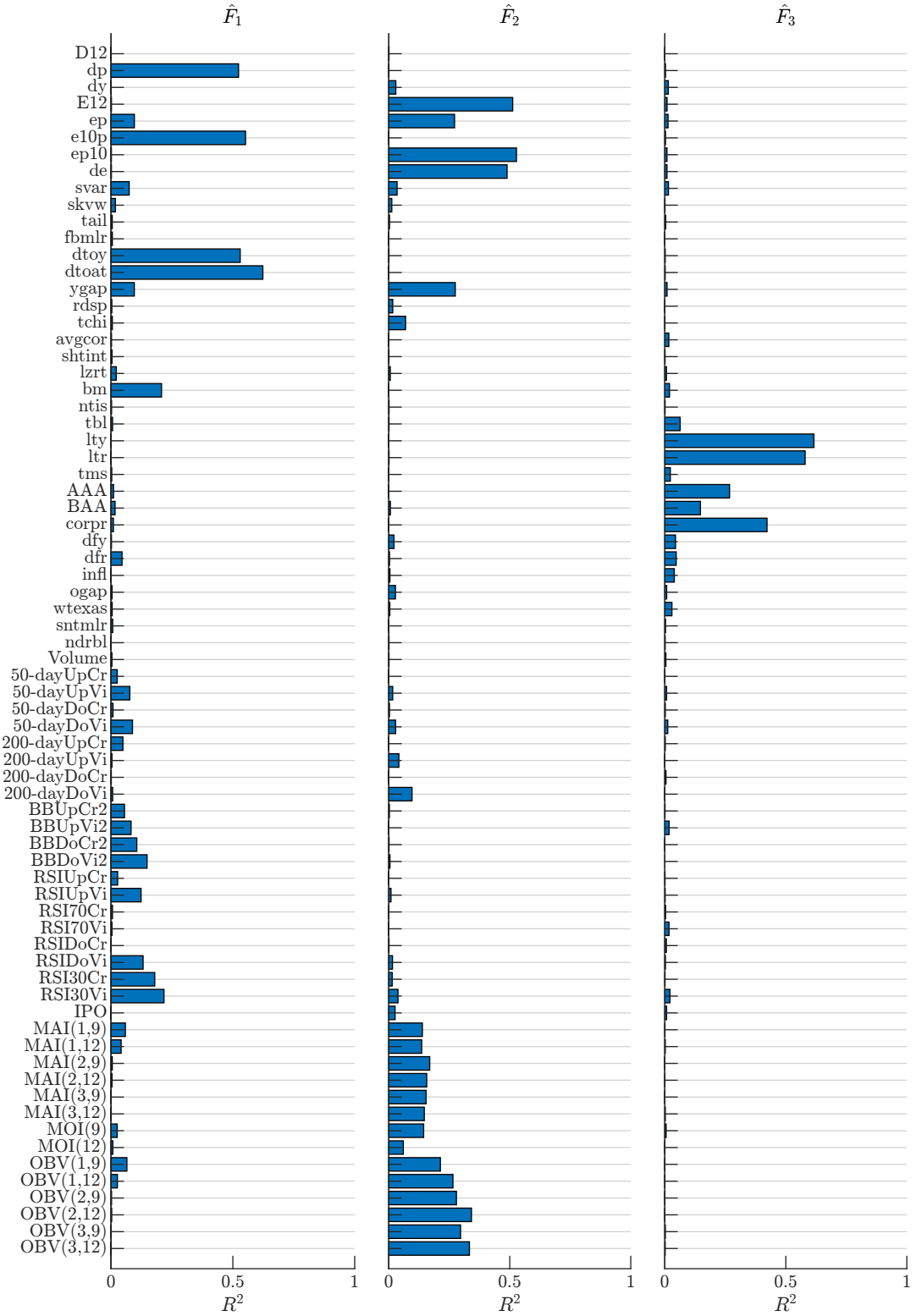

Figure 3: Univariate Pseudo-R^2 values of regressing X_i on $\hat{F}_i$. Factors extracted using *GD-NR*, and k selected by *GD* and BN_2.

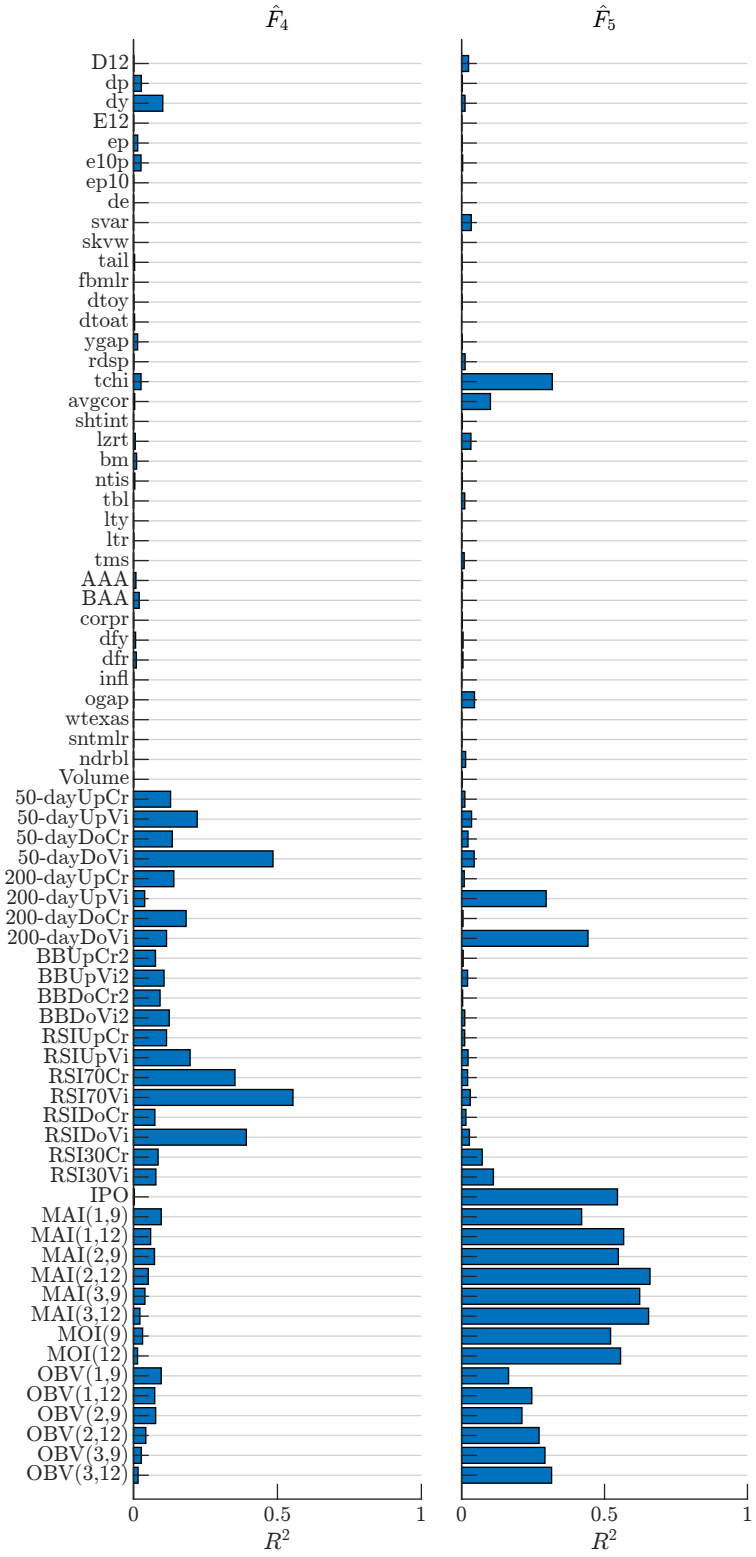

Figure 4: Univariate Pseudo-R^2 values of regressing X_i on $\hat{F}_i$. Factors extracted using *GD-NR*, and k selected by *GD* and BN_2.

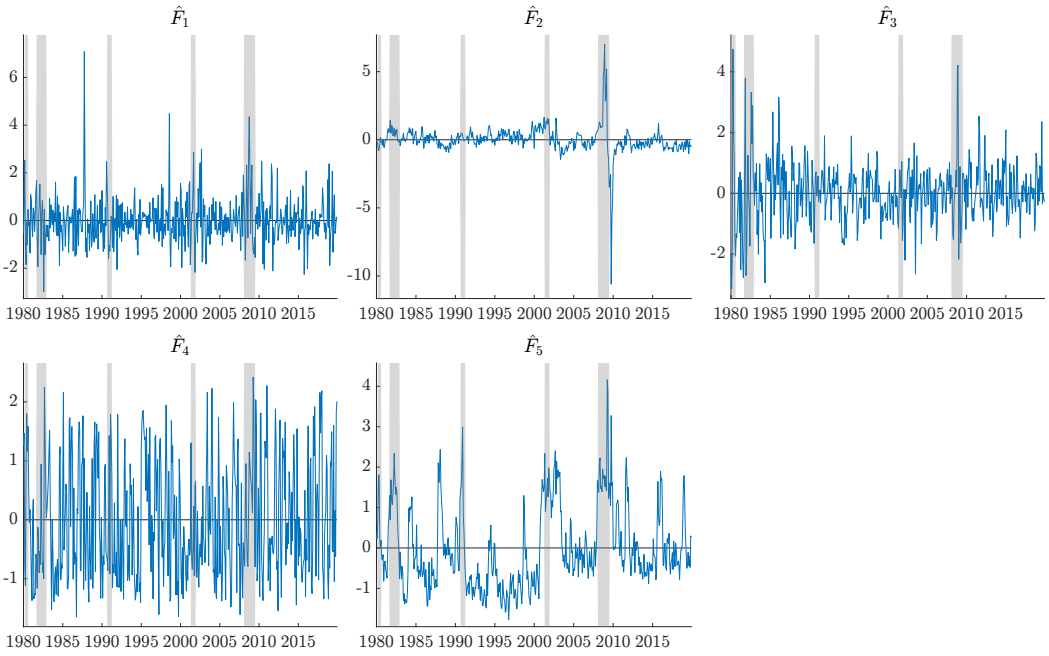

Figure 5: Extracted factor time series plot; Recessions periods (NBER recession indicator) highlighted in grey. Factors extracted using *GD-NR*, and k selected by *GD* and BN_2.

Figures 3 and 4 then show that each extracted factor links quite well to a particular group of variables. For instance, factor $\hat{F}_5$ accounts for a substantial share of the variation in the Neely *et al.* (2014) indicators, but contributes little to the variation in other variables, apart from two of our monthly aggregates. The two variables that count the number of bullish or bearish exceedances of the 200-day line are also considerably related to the fifth factor. But since 200 trading days amounts to roughly 10 months of data, which is between the 9 and 12 month windows considered by the MAI indicators, it is not surprising that these variables measure something quite similar to what the MAIs capture.

Nevertheless, this factor is comparable to the "technical indicator factor" in Neely *et al.* (2014), with our finding reiterating that macroeconomic variables and technical indicators can be seen as complementary and neglecting the latter would remove potentially valuable information from the dataset. However, other than in Neely *et al.*, the technical indicators do not relate evenly to the factor; indicated by the factor's explanatory power being weaker with respect to the on-balance volume indicators than with respect to the directly price-related indicators. Interestingly, *IPO* is also linked to factor 5, which might be due to market-cycle conditions raising/reducing IPO attractiveness over time.

Turning to our count variables, we note that they can be mainly attributed to a second factor. Factor 4 collects them with its explanatory power being particularly high with respect to the number of violations of the 50-day line and with respect to some RSI-based aggregates. Thus, this factor appears to capture shorter-horizon information — as seen in Figure 5, panels 4 and 5, which depict differing persistence levels of the extracted factors — than factor 5.

	GD vs. *PCA*	*NR* vs. *PCA*	*GD-NR* vs. *PCA*
$\mathcal{D}_F$	0.3897	0.2941	0.3976
$\mathcal{D}_F^{\text{gr}}$	0.2566	0.1930	0.2627

Table 4: Distances $\mathcal{D}_F$ and $\mathcal{D}_F^{\text{gr}}$ between the subspaces spanned by the extracted factors. k selected by *GD*, BN_2 and *PCA*, *GR*, respectively.

Some count variables even carry information that relates to continuous variables, as can be seen by analyzing the pseudo-R^2 values of factor 1 in Figure 3. Although this factor mainly impacts price variables like the dividend-price or the ten year earning average-price ratio, and the distance to the year/all time high, it also links to bearish count variables; in particular to the bearish RSI variables. In contrast to factor 4 where the bullish RSI variables exhibited higher R^2 values, it is now the overbought market crossings and days, as well as the bearish Bollinger band days that can be partially attributed to the factor. After examining the factor estimates in Figure 5, panel 1, this does not come as a surprise. $\hat{F}_1$ displays considerable peaks in times of financial market turmoil, *e.g.* in October 1987, in August 1998, or in 2002 and 2008, respectively.

Similarly, the second factor, which mainly collects earning information, also partially relates to binary price and volume information. However, keep in mind that this observation is rotation-dependent and note that continuous variables already provide the main information of the first three factors, as shown by the *PCA* factors being nearly identical to the respective *GD-NR* factors; see Figures 8 and 9.

Finally, factor 3 compresses bond market information in addition to the price, dividend, and earnings information compressed by the first two factors, respectively, with such patterns having been repeatedly identified in the literature. But unlike factor 1 whose extremes were mainly related to volatile markets, factor 3 explaining bond rate and yield changes can be attributed, at least partially, to the business cycle, with the factor's peaks occurring in times of recession; cf. Figure 5, panel 3. Turning back to factor 2, we note that it exhibits a similar behavior, reaching many of its peaks in times of recession. Although this is somehow obscured by the massive swing during the financial crisis/great recession in 2007–2009, factor 2 displays upward trends in the early 1980s, the 1990s, and 2000s all culminating in downswings during the subsequent recessions, *e.g.* the ones following the dot-com bust and the great financial crisis, respectively.

As mentioned earlier, there are *PCA* analogues for factors 1 to 3 extracted by *GD-NR* as suggested by the factor estimate paths and the pseudo-R^2 estimates in Figures 8 and 9. Although not as close to each other as the subspaces in Table 3, the subspaces spanned by the *GD-NR* factors and the *PCA* factors, respectively, are still fairly similar according to the subspace distances in Table 4. Once more, this supports the claim of complementarity between technical indicators and macroeconomic variables put forward by Neely *et al.* (2014, p. 1779).

5.3 Equity Premium Prediction

An in-sample and a pseudo-out-of-sample forecasting exercise complete our empirical illustration. In-sample, we utilize the factors from section 5.2, *i.e.* the extracted predictors are based on all available information, to predict the equity premium. This equity premium is calculated, as is standard in the literature, using the log S&P 500 returns (denoted as $r_{m,t}$) and the 3-month treasury bill rate serving as the proxy of the risk-free rate ($r_{f,t}$) as follows: $eq\_p_t = \ln(1 + r_{m,t}) - \ln(1 + r_{f,t})$. Afterwards, we compare the factors' predictive ability to an expanding window historic average that has also been used as a benchmark by, *inter alia*, Neely *et al.* (2014). Furthermore, AR(1)-based predictions of the equity premium serve as a second benchmark because they do not require any additional predictors. Letting $\overline{eq\_p}_t = \frac{1}{t}\sum_{j=1}^{t} eq\_p_j$ denote the historical average, the relative forecasting performance is assessed via

$$R^2_{FMSE} = 1 - \frac{\sum_{t=T^*}^{T-1}\left(eq\_p_{t+1} - \widehat{eq\_p}_{t+1}\right)^2}{\sum_{t=T^*}^{T-1}\left(eq\_p_{t+1} - \overline{eq\_p}_t\right)^2}, \tag{25}$$

cf. Campbell and Thompson (2008, p. 1515). For the in-sample results, set $T^* = 1$ and let $\widehat{eq\_p}_{t+1}$ denote the respective forecast of eq_p_{t+1} that can be found given the information available at time t. Then, $R^2_{FMSE} \in (-\infty, 1]$ indicates superior predictive performance of the factor-based forecast (benchmark) by values greater (smaller) than 0.

In contrast, our pseudo-out-of-sample exercise uses rolling windows to forecast the next observation. Here, our choice of 240 observations per rolling window ensures that there is enough data to extract meaningful factors (and factor numbers) while still leaving data to evaluate the predictive performance. Consequently, our prediction period begins in January 2000, *i.e.* $T^* = 240$, and continues until December 2019.

In addition, we examine whether the forecast performance differs between recession and expansion periods by constructing business-cycle versions of R^2_{FMSE}. Both in- and out-of-sample, only the residuals from either contraction (*rec*) or expansion (*exp*) periods are considered by pre-multiplying the forecast errors in equation (25) with an appropriate indicator $\mathbf{1}_t^d$ based on the NBER Recession Indicator available from the Federal Reserve Bank of St. Louis economic database. This then changes R^2_{FMSE} to

$$R^2_{FMSE,d} = 1 - \frac{\sum_{t=T^*}^{T-1}\mathbf{1}_t^d\left(eq\_p_{t+1} - \widehat{eq\_p}_{t+1}\right)^2}{\sum_{t=T^*}^{T-1}\mathbf{1}_t^d\left(eq\_p_{t+1} - \overline{eq\_p}_t\right)^2}, \quad d = exp,\ rec;$$

see, among others, Dai *et al.* (2025, p. 1133).

The results of our forecast exercise are presented in Table 5. As can be seen, *GD-NR* outperforms *GD* both in- and out-of-sample, while performing comparably to *NR*. Meanwhile, AR(1) forecasts, which rely solely on the equity premium's own history and ignore any additional available information, perform subpar throughout.

	GD	*NR*	*GD-NR*	*PCA*	*AR*
R^2_{IS}	0.0315	0.0368	0.0389	0.0381	0.0032
$R^2_{IS,rec}$	0.1098	0.1209	0.1206	0.1335	0.0362
$R^2_{IS,exp}$	0.0046	0.0079	0.0107	0.0054	-0.0082
R^2_{OOS}	-0.0262	-0.0344	-0.0260	-0.0017	-0.0187
$R^2_{OOS,rec}$	-0.0720	-0.1107	-0.0839	-0.0182	-0.0056
$R^2_{OOS,exp}$	-0.0048	0.0013	0.0012	0.0061	-0.0249

Table 5: 1-step-ahead predictability R^2_{FMSE} values relative to expanding window historical averages. Out-of-sample rolling window width of $T = 240$ used by the factor-based predictions, and $T^*_{IS} = 1$, $T^*_{OOS} = 240$ in (25). k selected by *GD*, BN_2 and *PCA*, *GR*, respectively.

The latter's particularly poor in-sample performance, together with *PCA*'s slightly weaker in-sample yet marginally better out-of-sample performance during expansions (relative to *GD-NR*), highlights how difficult it is to predict the equity premium. As equity premium predictability is likely to be episodic rather than permanent, see Demetrescu *et al.* (2022) among others, identifying such predictability spells and its respective predictors is non-trivial. Both in- and out-of-sample, making valid statements, *i.e.* based on inference, with respect to the (extracted) predictors must account for the effects of the stationarity-inducing data transformations and the varying and unknown persistence of the untransformed variables, as both data characteristics affect inference. In addition, the serial testing issue in the out-of-sample case and the general two-stage nature of the factor-based prediction further complicate inference, which is why we deem statements regarding the statistical significance of our predictability findings beyond the scope of this paper. Nonetheless, our factor extraction algorithm extracts meaningful factors that help predict the equity premium according to FMSE-based measures, with future research being required to investigate the question of how to validate this conclusion through inferential arguments.

6 Conclusion

In this paper, we have examined how to extract factors from a large dataset containing various data types. Out of the different factor extraction variants, we recommend *GD-NR* as it combines the reduced computational costs of gradient descent and the improved numerical accuracy of a Newton-Raphson step. This variant has been shown to perform particularly well in our extensive simulation study considering continuous, count, and binary observable data. Extending our extraction framework beyond these data types is straightforward. Thanks to the alternating regression structure built around generalized linear models, the framework can easily accommodate additional data types and distributional assumptions, *e.g.* categorical or duration data, so long as a corresponding generalized linear model is available. The necessary effort to do this can be

particularly small since many software packages already contain the required fundamental GLM estimators, additionally reducing the costs of implementing the framework using other software packages. How the framework is implemented might then enable numerical improvements reducing computational costs, *e.g.* through suitable parallelization or different gradient descent learning rates. Finally, convergence guarantees of the algorithm and how to select the number of factors in the face of sparse loading matrices or misspecified distributional assumptions leave room for further improvements. Nevertheless, our simulation study and the empirical application demonstrate that our framework already enables empirical researchers to make use of the ever-increasing datasets, compressing vast amounts of data into a few interpretable indices that can then be used to predict other variables of interest.

References

Ahn, S. C., & Horenstein, A. R. (2013). Eigenvalue Ratio Test for the Number of Factors. *Econometrica*, *81*(3), 1203–1227.

Bai, J., & Ng, S. (2002). Determining the Number of Factors in Approximate Factor Models. *Econometrica*, *70*(1), 191–221.

Bhamidipaty, L. M., Kochenderfer, M. J., & Hastie, T. (2025). ExpFamilyPCA.jl: A Julia Package for Exponential Family Principal Component Analysis. *Journal of Open Source Software*, *10*(105), 7403.

Breitung, J., & Eickmeier, S. (2016). Analyzing International Business and Financial Cycles using Multi-Level Factor Models: A Comparison of Alternative Approaches. In E. Hillebrand & S. J. Koopman (Eds.), *Dynamic Factor Models*. Emerald Group Publishing Limited.

Campbell, J. Y. (2008). Viewpoint: Estimating the equity premium. *Canadian Journal of Economics/Revue canadienne d'économique*, *41*(1), 1–21.

Campbell, J. Y., & Shiller, R. J. (1988). Stock Prices, Earnings, and Expected Dividends. *The Journal of Finance*, *43*(3), 661–676.

Campbell, J. Y., & Thompson, S. B. (2008). Predicting Excess Stock Returns Out of Sample: Can Anything Beat the Historical Average? *The Review of Financial Studies*, *21*(4), 1509–1531.

Cohn, J. B., Liu, Z., & Wardlaw, M. I. (2022). Count (and count-like) data in finance. *Journal of Financial Economics*, *146*(2), 529–551.

Collins, M., Dasgupta, S., & Schapire, R. E. (2001). A Generalization of Principal Components Analysis to the Exponential Family. In T. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Advances in Neural Information Processing Systems* (Vol. 14). MIT Press.

Creal, D., Schwaab, B., Koopman, S. J., & Lucas, A. (2014). Observation-driven mixed-measurement dynamic factor models with an application to credit risk. *The Review of Economics and Statistics*, *96*(5), 898–915.

Dai, Z., Jiang, F., Kang, J., & Xue, B. (2025). Stock return predictability in the frequency domain. *International Journal of Forecasting*, *41*(3), 1126–1147.

Demetrescu, M., Georgiev, I., Rodrigues, P. M. M., & Taylor, A. M. R. (2022). Testing for episodic predictability in stock returns. *Journal of Econometrics*, *227*(1), 85–113.

Duchi, J., Hazan, E., & Singer, Y. (2011). Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, *12*(61), 2121–2159.

Fama, E. F. (1970). Efficient Capital Markets: A Review of Theory and Empirical Work. *The Journal of Finance*, *25*(2), 383–417.

Fama, E. F., & French, K. R. (1988). Dividend yields and expected stock returns. *Journal of Financial Economics*, *22*(1), 3–25.

Goyal, A., Welch, I., & Zafirov, A. (2024). A Comprehensive 2022 Look at the Empirical Performance of Equity Premium Prediction. *The Review of Financial Studies*, *37*(11), 3490–3557.

Harvey, D. I., Leybourne, S. J., Sollis, R., & Taylor, A. M. R. (2021). Real-time detection of regimes of predictability in the US equity premium. *Journal of Applied Econometrics*, *36*(1), 45–70.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (2nd edition) [Corrected 12th printing, 2017]. Springer.

He, Y., Zhao, R., & Zhou, W.-X. (2023). *An Efficient Iterative Least Squares Algorithm for Large-dimensional Matrix Factor Model via Random Projection* (arXiv preprint No. 2301.00360v2). arXiv.

Ibbotson, R. G., Sindelar, J. L., & Ritter, J. R. (1994). The Market's Problems with the Pricing of Initial Public Offerings. *Journal of Applied Corporate Finance*, *7*(1), 66–74.

Khan, M. E., Bouchard, G., Murphy, K. P., & Marlin, B. M. (2010). Variational Bounds for Mixed-Data Factor Analysis. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, & A. Culotta (Eds.), *Advances in Neural Information Processing Systems* (Vol. 23). Curran Associates, Inc.

Kreiss, J.-P. (1985). A note on M-estimation in stationary ARMA processes. *Statistics & Decisions*, *3*, 317–336.

Liu, L. T., Dobriban, E., & Singer, A. (2018). ePCA: High dimensional exponential family PCA. *The Annals of Applied Statistics*, *12*(4), 2121–2150.

Lu, M., He, K., Huang, J. Z., & Qian, X. (2018). Principal Component Analysis for Exponential Family Data. In G. R. Naik (Ed.), *Advances in Principal Component Analysis: Research and Development* (pp. 193–223). Springer Singapore.

Ludvigson, S. C., & Ng, S. (2011). A Factor Analysis of Bond Risk Premia. In A. Ullah & D. E. A. Giles (Eds.), *Handbook of Empirical Economics and Finance* (1st ed., pp. 313–371). Chapman and Hall/CRC.

McCracken, M. W., & Ng, S. (2016). FRED-MD: A Monthly Database for Macroeconomic Research. *Journal of Business & Economic Statistics*, *34*(4), 574–589.

Murphy, K. P. (2012). *Machine Learning: A Probabilistic Perspective*. MIT press.

Neely, C. J., Rapach, D. E., Tu, J., & Zhou, G. (2014). Forecasting the Equity Risk Premium: The Role of Technical Indicators. *Management Science*, *60*(7), 1772–1791.

Phillips, P. C. B., & Lee, J. H. (2013). Predictive regression under various degrees of persistence and robust long-horizon regression. *Journal of Econometrics*, *177*(2), 250–264.

Stock, J. H., & Watson, M. W. (2002a). Forecasting Using Principal Components From a Large Number of Predictors. *Journal of the American Statistical Association*, *97*(460), 1167–1179.

Stock, J. H., & Watson, M. W. (2002b). Macroeconomic Forecasting Using Diffusion Indexes. *Journal of Business & Economic Statistics*, *20*(2), 147–162.

Welch, I., & Goyal, A. (2008). A Comprehensive Look at The Empirical Performance of Equity Premium Prediction. *The Review of Financial Studies*, *21*(4), 1455–1508.

Welling, M., Chemudugunta, C., & Sutter, N. (2008). Deterministic Latent Variable Models and their Pitfalls. *Proceedings of the 2008 SIAM International Conference on Data Mining*, 196–207.

Welsh, A. H., & Ronchetti, E. (2002). A journey in single steps: robust one-step M-estimation in linear regression. *Journal of Statistical Planning and Inference*, *103*(1–2), 287–310.

Ye, K., & Lim, L.-H. (2016). Schubert Varieties and Distances between Subspaces of Different Dimensions. *SIAM Journal on Matrix Analysis and Applications*, *37*(3), 1176–1197.

Zhao, J., Shang, C., Li, S., Xin, L., & Yu, P. L. H. (2025). Choosing the number of factors in factor analysis with incomplete data via a novel hierarchical Bayesian information criterion. *Advances in Data Analysis and Classification*, *19*, 209–235.

Zhu, P., & Knyazev, A. V. (2013). Angles between subspaces and their tangents. *Journal of Numerical Mathematics*, *21*(4), 325–340.

Algorithm 1: Multi-data type Factor Extraction Algorithm

Input:

Dataset $\mathbf{X}_{N\times T}$ with mixed data types;

Con-/Divergence thresholds;

Iteration limits; both overall and for R^2_{adj} changes

Output:

Estimated factor scores $\hat{\mathbf{F}}_{(k+1)\times T}$

Estimated loadings $\hat{\mathbf{\Lambda}}_{N\times(k+1)}$

Procedure:

Obtain initial $\hat{\mathbf{F}}^{(0)}$, $\hat{\mathbf{\Lambda}}^{(0)}$ values based on standard PCA applied to all variables in $\mathbf{X}$;

Compute initial adjusted pseudo-R^2: ${R^2_{\text{adj}}}^{(0)}$;

Set iteration counter $s \leftarrow 0$;

for *s smaller equal iteration limit* **do**

- Update factors to $\hat{\mathbf{F}}^{(s+1)}$ using current $\hat{\mathbf{\Lambda}}^{(s)}$ and $\hat{\mathbf{F}}^{(s)}$;
- Orthogonalize $\hat{\mathbf{F}}^{(s+1)}$ via compact SVD;
- Update loadings to $\hat{\mathbf{\Lambda}}^{(s+1)}$ using current $\hat{\mathbf{\Lambda}}^{(s)}$ and orthogonalized $\hat{\mathbf{F}}^{(s+1)}$;
- Compute new adjusted pseudo-R^2: ${R^2_{\text{adj}}}^{(s+1)}$;
- Increment step counter: $s \leftarrow s + 1$;
- **if** R^2_{adj} *stabilizes within tolerance or deteriorates for some consecutive cycles* **then**
  - **Terminate** (*convergence or deterioration confirmed*);

if *Extraction variant is GD-NR* **then** `// Optional refinement step`

- Perform one Newton–Raphson update cycle after the *GD* loop terminated;

return Final estimates $\left(\hat{\mathbf{F}}^{(s)}, \hat{\mathbf{\Lambda}}^{(s)}\right)$;

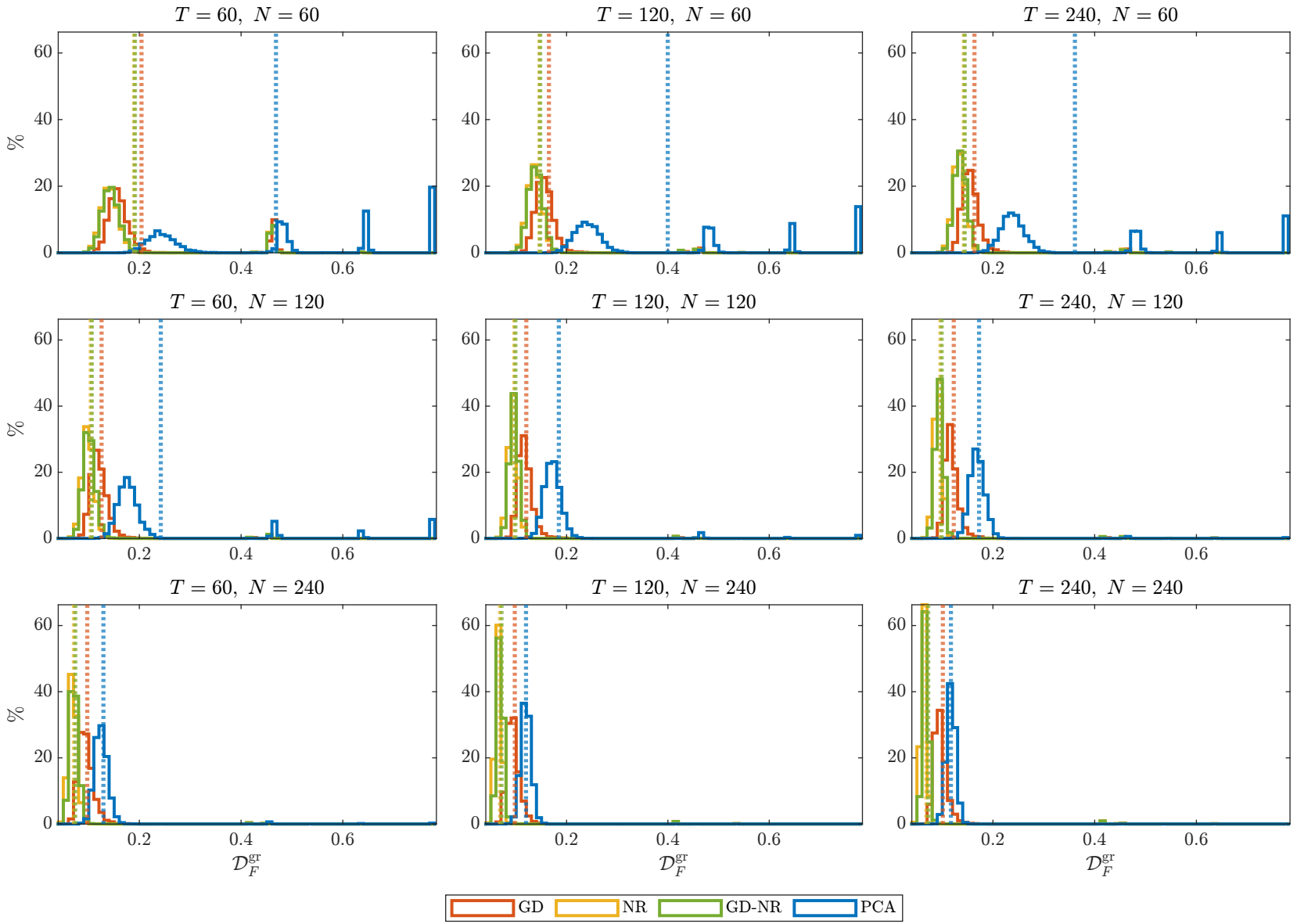

Figure 6: $\mathcal{D}_F^{\text{gr}}$ histogram (average values as dotted lines). Data simulated from DGP1.1.

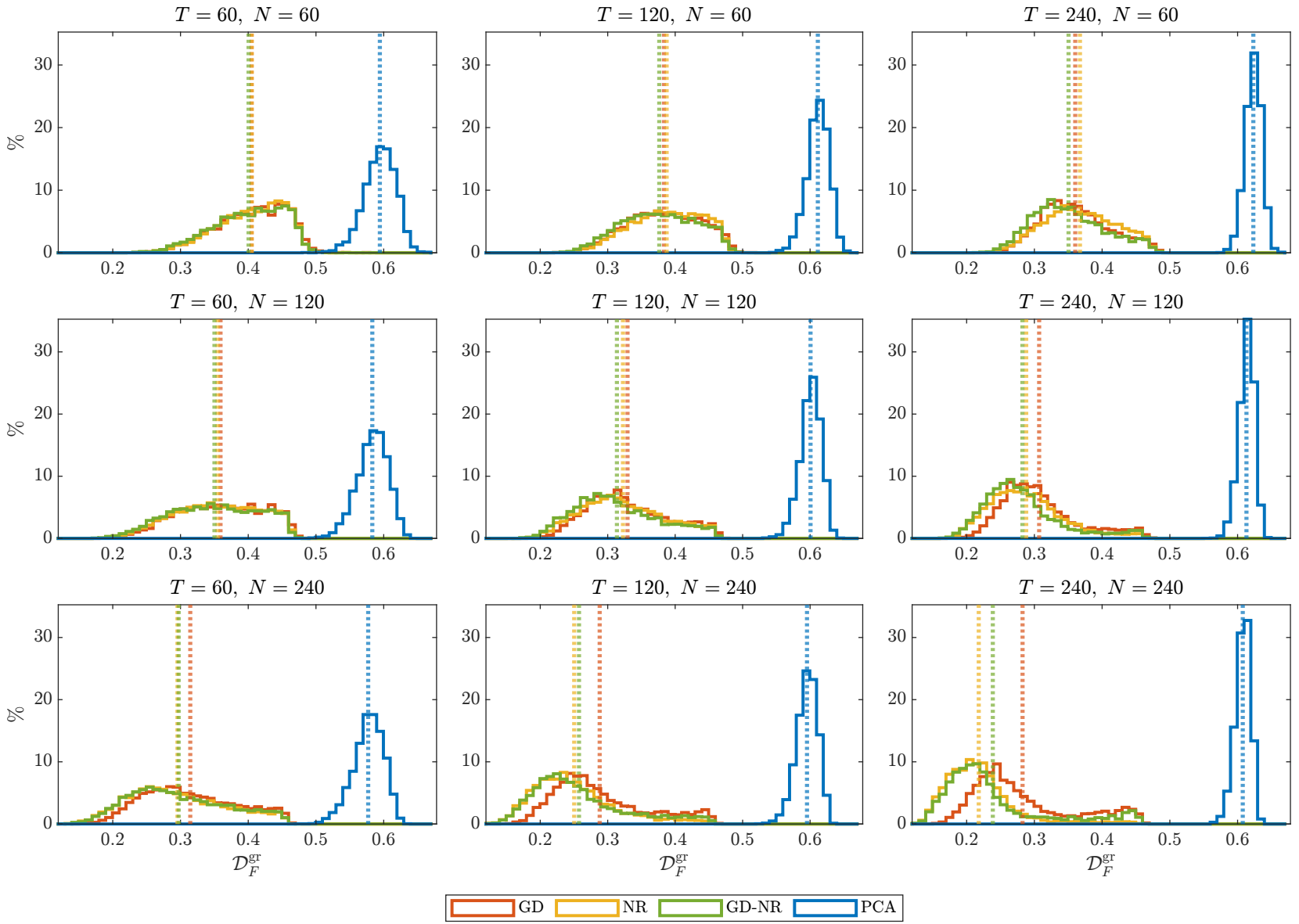

Figure 7: $\mathcal{D}_F^{\mathrm{gr}}$ histogram (average values as dotted lines). Data simulated from DGP4.3.

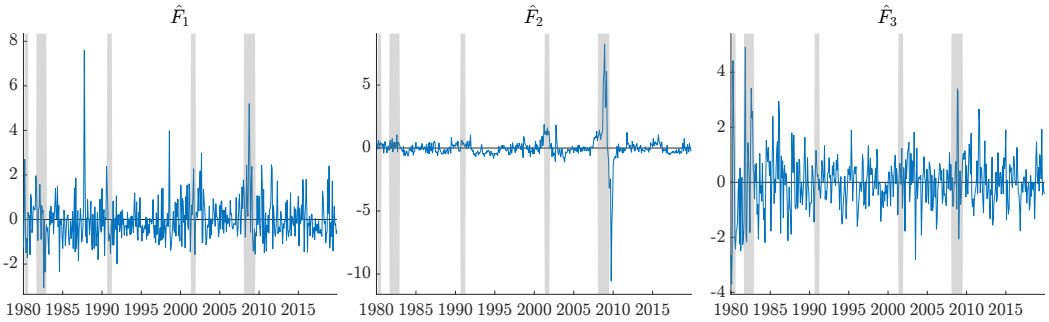

Figure 8: Extracted factor time series plot; Recessions periods (NBER recession indicator) highlighted in grey. Factors extracted using PCA, and k selected by GR.

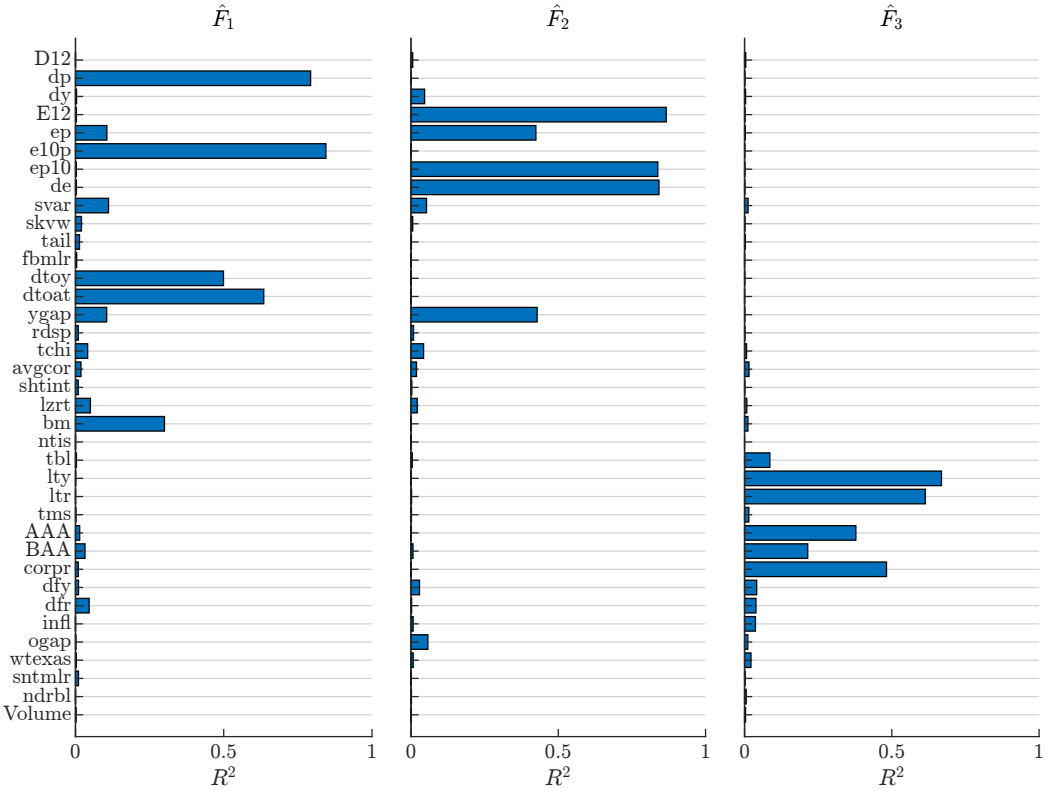

Figure 9: Univariate Pseudo-R^2 values of regressing X_i on $\hat{F}_i$. Factors extracted using PCA, and k selected by GR.

Supplementary Appendix to "Fast Factor Extraction for Mixed Type Financial Data"

Fabian Schmidt[a], Matei Demetrescu[a]

[a] Department of Statistics, TU Dortmund University

May 20, 2026

Abstract

This supplementary appendix contains two sections. Section S.1 collects the additional Monte Carlo results from section 4 of the main paper. Afterwards, section S.2 provides additional material related to the empirical application reported in section 5 of the main part.

*The authors gratefully acknowledge financial support from the German Research Foundation (DFG) within TRR 391: Spatio-temporal Statistics for the Transition of Energy and Transport (520388526). Address correspondence to: Matei Demetrescu, TU Dortmund University, Department of Statistics, Vogelpothsweg 78, 44227 Dortmund, Germany. Email: mdeme@statistik.tu-dortmund.de.

S.1 Additional Simulation Study Results

N	T	R^2_{adj}	BN_1	BN_2	BN_3	ER	GR
60	60	4.23 (0.50)	3.94 (0.28)	3.85 (0.38)	4.20 (0.47)	2.56 (1.22)	2.87 (1.18)
	120	4.21 (0.47)	4.00 (0.14)	3.99 (0.15)	4.05 (0.23)	2.88 (1.22)	3.20 (1.09)
	240	4.26 (0.52)	4.01 (0.13)	4.01 (0.11)	4.03 (0.17)	3.07 (1.19)	3.39 (1.02)
120	60	4.15 (0.39)	4.00 (0.10)	4.00 (0.12)	4.03 (0.18)	3.46 (1.04)	3.71 (0.77)
	120	4.18 (0.43)	4.01 (0.12)	4.01 (0.08)	4.10 (0.32)	3.86 (0.57)	3.95 (0.34)
	240	4.22 (0.48)	4.02 (0.13)	4.01 (0.10)	4.05 (0.22)	3.95 (0.34)	3.98 (0.17)
240	60	4.12 (0.35)	4.01 (0.10)	4.01 (0.09)	4.01 (0.12)	3.95 (0.37)	3.98 (0.19)
	120	4.17 (0.42)	4.01 (0.09)	4.01 (0.07)	4.03 (0.19)	4.00 (0.04)	4.00 (0.00)
	240	4.21 (0.47)	4.02 (0.14)	4.01 (0.12)	4.10 (0.31)	4.00 (0.00)	4.00 (0.00)

Table S.1: Average number of extracted factors using selection criteria R^2_{adj}, BN_1, BN_2, BN_3, ER, and GR (Standard deviations in parentheses). Data simulated from DGP1.2.

N	T	*GD*				*NR*				*GD-NR*		*PCA*
		R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	sec.	R^2_{adj}
60	60	0.446 (0.103)	15.66 (2.15)	15.43 (2.15)	0.0048 (0.0024)	0.450 (0.105)	7.84 (0.49)	5.71 (1.78)	0.0111 (0.0045)	0.449 (0.105)	0.0062 (0.0030)	0.189 (0.060)
	120	0.469 (0.090)	15.37 (1.84)	15.16 (1.89)	0.0077 (0.0012)	0.475 (0.091)	7.76 (0.50)	5.54 (1.72)	0.0164 (0.0023)	0.474 (0.091)	0.0098 (0.0014)	0.208 (0.059)
	240	0.480 (0.085)	15.20 (1.80)	15.03 (1.86)	0.0144 (0.0021)	0.485 (0.086)	7.71 (0.51)	5.27 (1.63)	0.0292 (0.0033)	0.485 (0.086)	0.0181 (0.0022)	0.219 (0.057)
120	60	0.456 (0.094)	15.06 (1.87)	14.83 (1.88)	0.0076 (0.0013)	0.461 (0.095)	7.47 (0.60)	5.70 (1.60)	0.0146 (0.0026)	0.461 (0.095)	0.0095 (0.0017)	0.242 (0.051)
	120	0.479 (0.085)	14.75 (1.67)	14.54 (1.71)	0.0137 (0.0018)	0.483 (0.086)	7.40 (0.59)	5.85 (1.59)	0.0234 (0.0030)	0.483 (0.086)	0.0167 (0.0021)	0.263 (0.041)
	240	0.492 (0.078)	14.54 (1.51)	14.36 (1.55)	0.0258 (0.0030)	0.496 (0.078)	7.32 (0.53)	5.70 (1.52)	0.0412 (0.0042)	0.496 (0.078)	0.0313 (0.0033)	0.272 (0.039)
240	60	0.460 (0.093)	14.73 (1.66)	14.53 (1.68)	0.0138 (0.0019)	0.465 (0.094)	7.39 (0.63)	6.13 (1.48)	0.0239 (0.0030)	0.464 (0.094)	0.0169 (0.0020)	0.269 (0.040)
	120	0.482 (0.079)	14.44 (1.50)	14.27 (1.52)	0.0262 (0.0029)	0.487 (0.080)	7.31 (0.59)	6.31 (1.38)	0.0389 (0.0041)	0.486 (0.080)	0.0312 (0.0032)	0.282 (0.036)
	240	0.497 (0.074)	14.34 (1.38)	14.20 (1.41)	0.0512 (0.0052)	0.501 (0.074)	7.23 (0.51)	6.22 (1.32)	0.0701 (0.0064)	0.501 (0.074)	0.0603 (0.0054)	0.292 (0.035)

Table S.2: Average R^2_{adj}, total/R^2_{adj}-optimal iteration counts $\mathcal{I}/\mathcal{I}^*$, and computation times in seconds (Standard deviations in parentheses). Data simulated from DGP1.2.

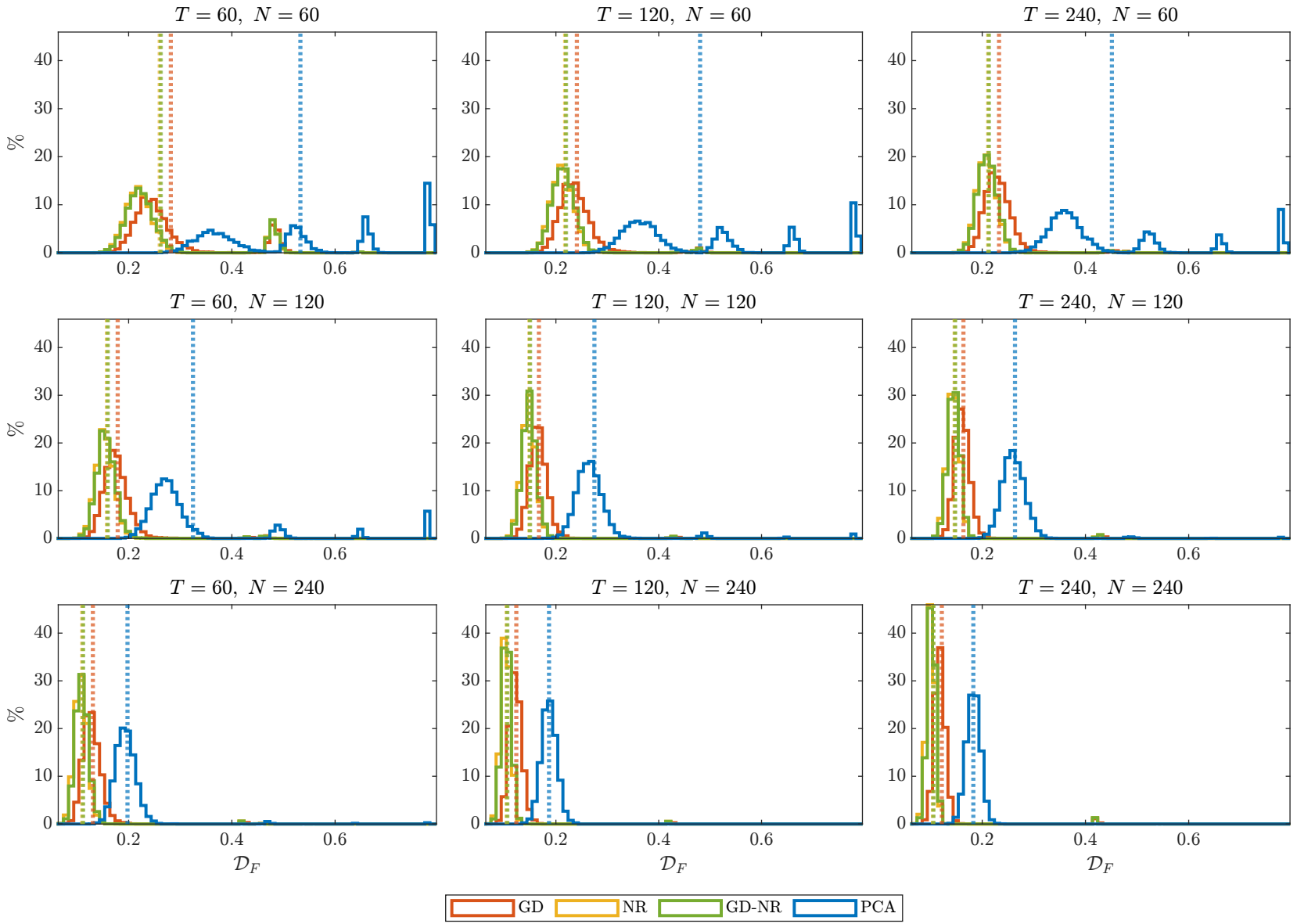

Figure S.1: $\mathcal{D}_F$ histogram (average values as dotted lines). Data simulated from DGP1.2.

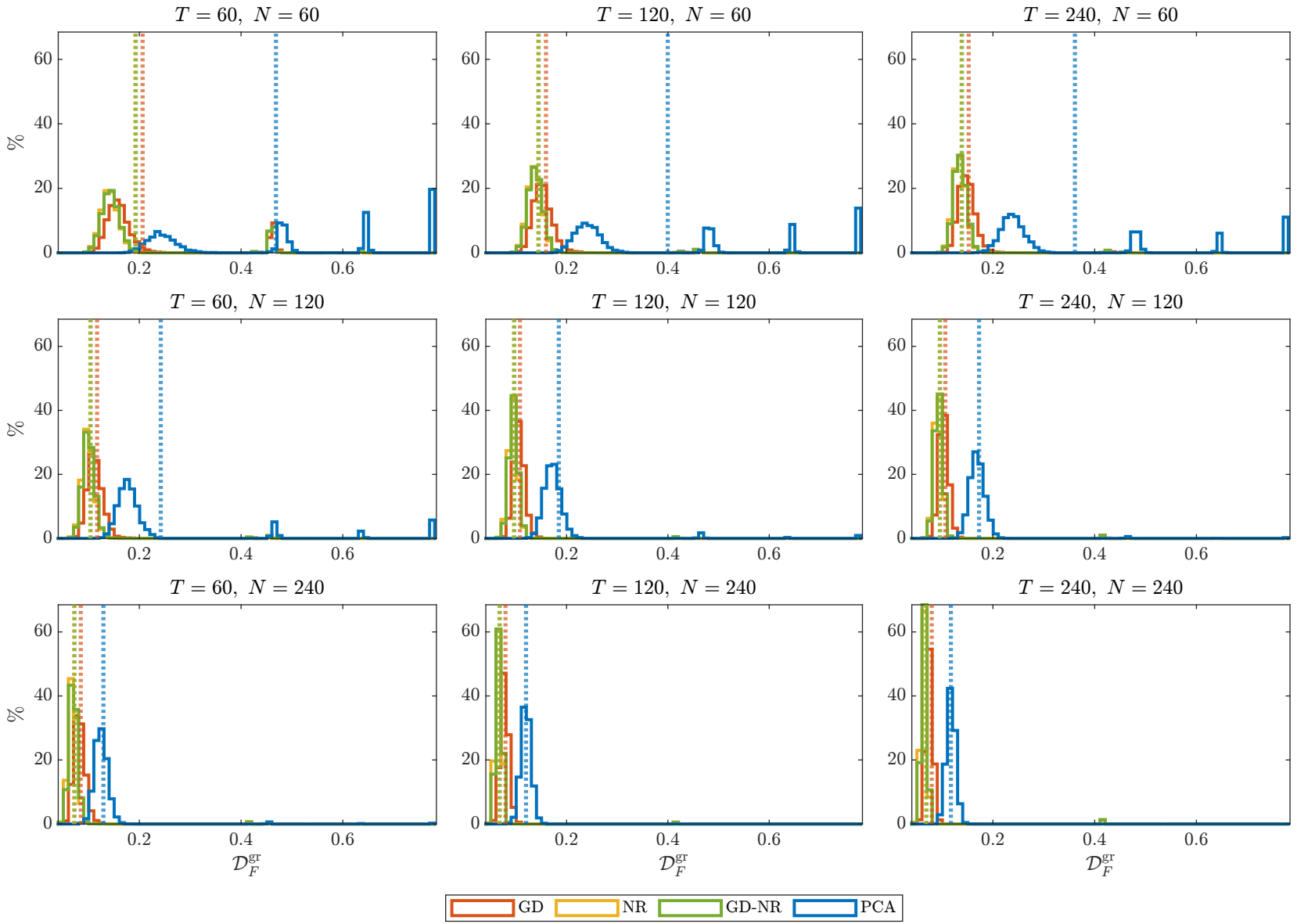

Figure S.2: $\mathcal{D}_F^{\mathrm{gr}}$ histogram (average values as dotted lines). Data simulated from DGP1.2.

N	T	R^2_{adj}	BN_1	BN_2	BN_3	ER	GR
60	60	4.42 (0.67)	3.24 (0.64)	2.70 (0.69)	4.30 (0.60)	1.89 (1.03)	2.06 (1.09)
	120	4.20 (0.46)	3.56 (0.55)	3.29 (0.62)	3.95 (0.24)	2.00 (1.09)	2.20 (1.14)
	240	4.09 (0.33)	3.72 (0.47)	3.62 (0.53)	3.92 (0.30)	2.15 (1.18)	2.38 (1.21)
120	60	4.25 (0.50)	3.67 (0.50)	3.44 (0.59)	3.97 (0.19)	2.31 (1.25)	2.60 (1.25)
	120	4.23 (0.49)	3.96 (0.20)	3.83 (0.39)	4.04 (0.22)	2.89 (1.31)	3.21 (1.18)
	240	4.16 (0.44)	3.99 (0.12)	3.98 (0.15)	4.00 (0.10)	3.39 (1.12)	3.62 (0.91)
240	60	4.10 (0.33)	3.88 (0.32)	3.82 (0.39)	3.97 (0.17)	3.17 (1.24)	3.44 (1.07)
	120	4.18 (0.45)	4.00 (0.05)	4.00 (0.07)	4.00 (0.07)	3.81 (0.69)	3.92 (0.46)
	240	4.17 (0.46)	4.00 (0.07)	4.00 (0.07)	4.02 (0.17)	3.98 (0.22)	4.00 (0.11)

Table S.3: Average number of extracted factors using selection criteria R^2_{adj}, BN_1, BN_2, BN_3, ER, and GR (Standard deviations in parentheses). Data simulated from DGP2.

N	T	*GD*				*NR*				*GD-NR*		*PCA*
		R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	sec.	R^2_{adj}
60	60	0.233 (0.072)	14.58 (2.39)	14.42 (2.45)	0.0047 (0.0019)	0.219 (0.078)	7.00 (0.17)	6.08 (1.41)	0.0102 (0.0044)	0.219 (0.078)	0.0061 (0.0032)	0.087 (0.035)
	120	0.227 (0.059)	15.59 (2.04)	15.50 (2.07)	0.0082 (0.0013)	0.222 (0.062)	7.06 (0.25)	6.03 (1.46)	0.0153 (0.0019)	0.222 (0.062)	0.0103 (0.0015)	0.085 (0.032)
	240	0.221 (0.051)	16.11 (1.91)	16.05 (1.92)	0.0156 (0.0022)	0.222 (0.054)	7.15 (0.38)	6.14 (1.47)	0.0276 (0.0030)	0.222 (0.054)	0.0193 (0.0024)	0.086 (0.032)
120	60	0.241 (0.066)	15.58 (1.88)	15.49 (1.91)	0.0081 (0.0014)	0.238 (0.070)	7.00 (0.15)	5.90 (1.39)	0.0139 (0.0019)	0.238 (0.070)	0.0100 (0.0014)	0.111 (0.038)
	120	0.235 (0.054)	15.99 (1.61)	15.94 (1.62)	0.0161 (0.0019)	0.237 (0.056)	7.06 (0.25)	6.06 (1.37)	0.0238 (0.0025)	0.237 (0.056)	0.0191 (0.0021)	0.120 (0.035)
	240	0.232 (0.046)	16.14 (1.46)	16.10 (1.47)	0.0306 (0.0030)	0.237 (0.048)	7.13 (0.36)	6.54 (1.16)	0.0429 (0.0037)	0.237 (0.047)	0.0362 (0.0033)	0.128 (0.030)
240	60	0.246 (0.066)	15.85 (1.62)	15.79 (1.63)	0.0150 (0.0019)	0.247 (0.068)	6.99 (0.16)	5.86 (1.33)	0.0230 (0.0022)	0.247 (0.068)	0.0181 (0.0021)	0.139 (0.037)
	120	0.240 (0.051)	15.90 (1.36)	15.88 (1.37)	0.0302 (0.0030)	0.244 (0.051)	7.05 (0.24)	6.28 (1.18)	0.0395 (0.0032)	0.244 (0.051)	0.0352 (0.0030)	0.148 (0.027)
	240	0.237 (0.044)	15.90 (1.32)	15.88 (1.33)	0.0637 (0.0051)	0.242 (0.045)	7.11 (0.34)	6.74 (0.89)	0.0773 (0.0057)	0.242 (0.050)	0.0729 (0.0054)	0.152 (0.024)

Table S.4: Average R^2_{adj}, total/R^2_{adj}-optimal iteration counts $\mathcal{I}/\mathcal{I}^*$, and computation times in seconds (Standard deviations in parentheses). Data simulated from DGP2.

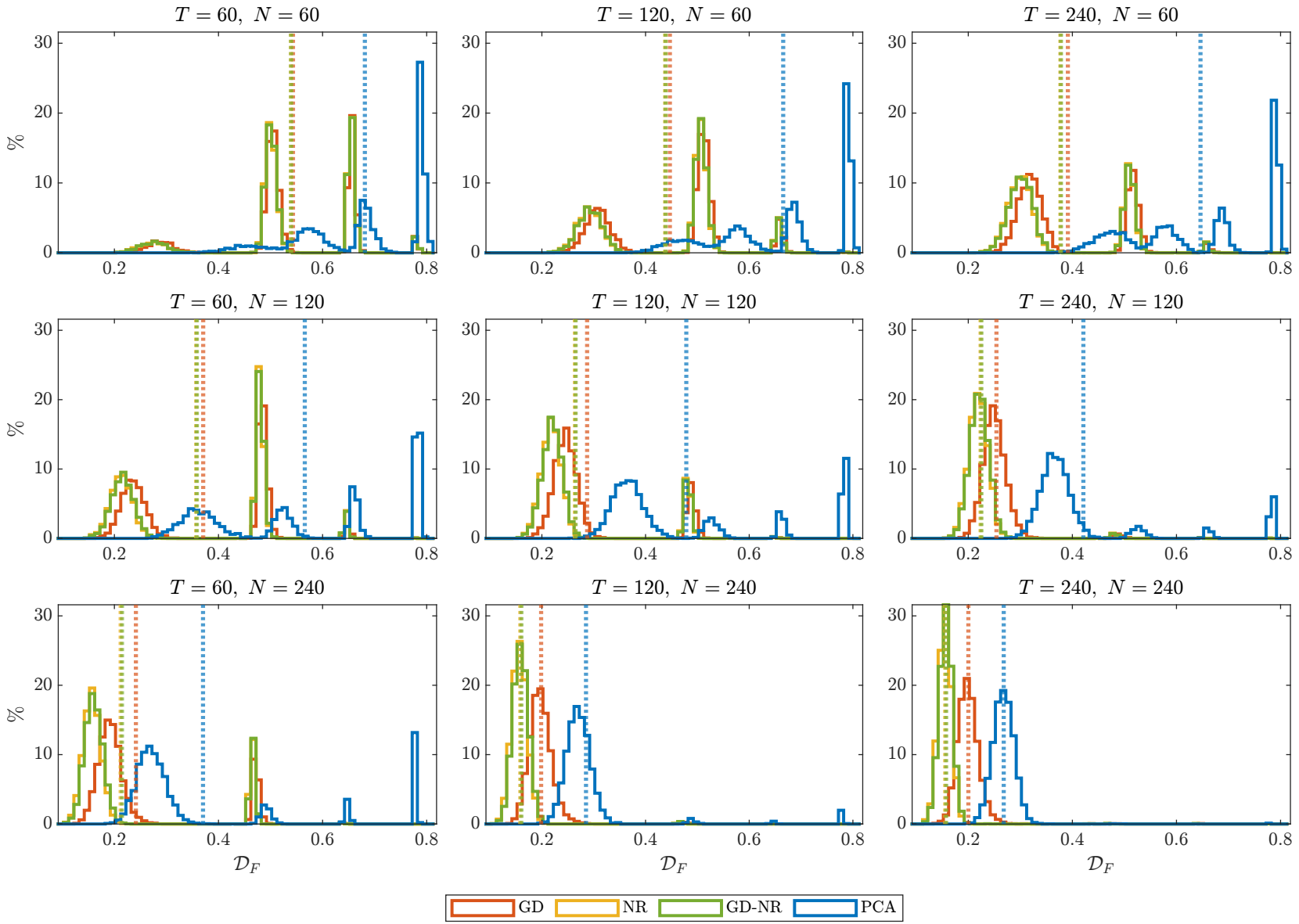

Figure S.3: $\mathcal{D}_F$ histogram (average values as dotted lines). Data simulated from DGP2.

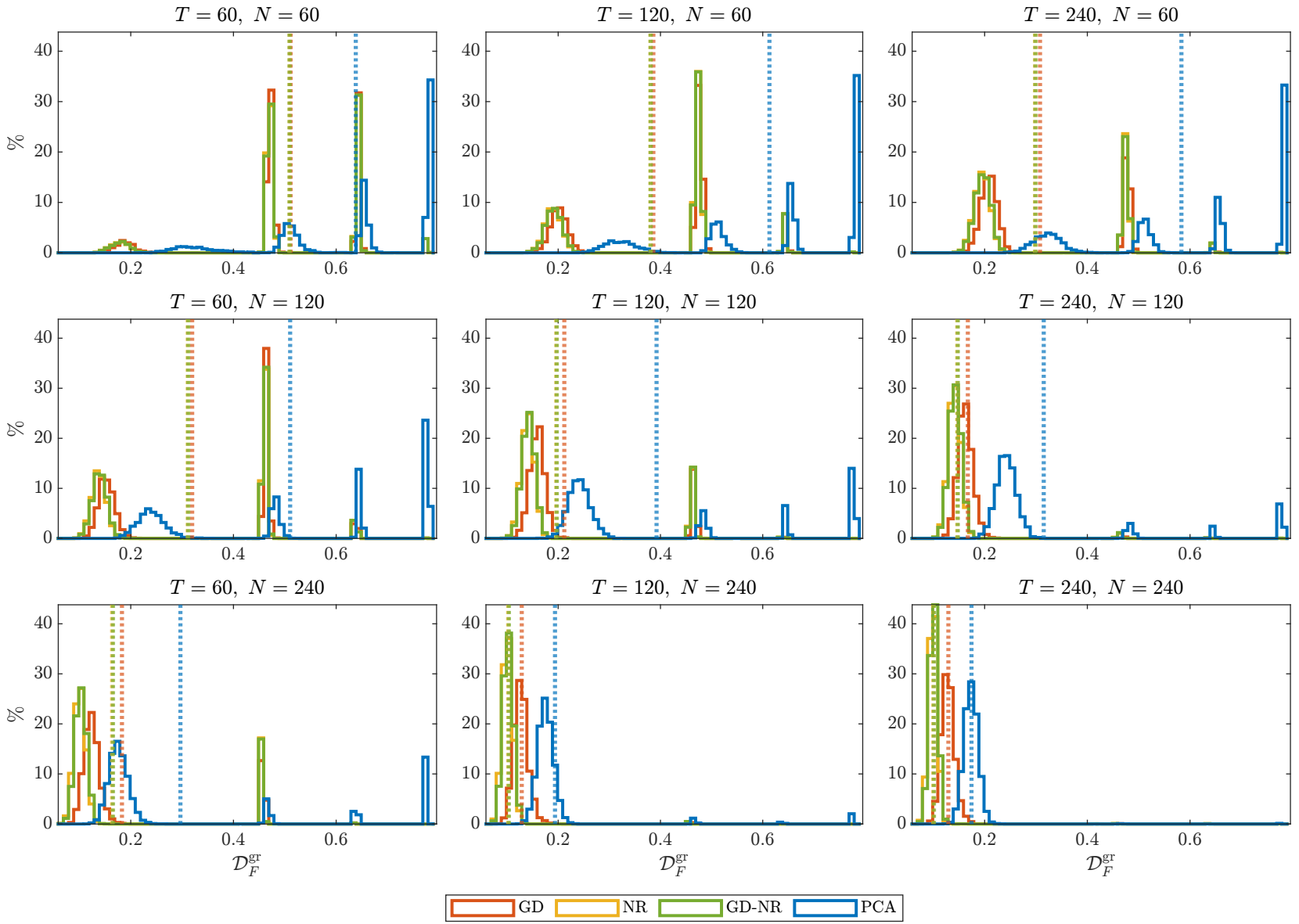

Figure S.4: $\mathcal{D}_F^{\text{gr}}$ histogram (average values as dotted lines). Data simulated from DGP2.

N	T	R^2_{adj}	BN_1	BN_2	BN_3	ER	GR
60	60	4.30 (0.55)	3.93 (0.26)	3.77 (0.44)	4.24 (0.50)	2.53 (1.21)	2.83 (1.19)
	120	4.21 (0.46)	3.99 (0.12)	3.98 (0.16)	4.02 (0.15)	2.82 (1.23)	3.12 (1.14)
	240	4.12 (0.40)	3.99 (0.16)	3.99 (0.17)	4.00 (0.16)	3.03 (1.21)	3.33 (1.06)
120	60	4.17 (0.42)	4.00 (0.07)	3.99 (0.11)	4.01 (0.11)	3.49 (1.02)	3.70 (0.79)
	120	4.18 (0.47)	4.00 (0.07)	4.00 (0.05)	4.07 (0.28)	3.86 (0.57)	3.94 (0.36)
	240	4.11 (0.38)	4.00 (0.08)	4.00 (0.08)	4.00 (0.11)	3.97 (0.27)	3.99 (0.17)
240	60	4.08 (0.31)	4.00 (0.03)	4.00 (0.03)	4.00 (0.04)	3.96 (0.30)	3.99 (0.16)
	120	4.12 (0.40)	4.00 (0.05)	4.00 (0.04)	4.01 (0.11)	4.00 (0.01)	4.00 (0.00)
	240	4.10 (0.34)	4.00 (0.07)	4.00 (0.06)	4.03 (0.18)	4.00 (0.00)	4.00 (0.00)

Table S.5: Average number of extracted factors using selection criteria R^2_{adj}, BN_1, BN_2, BN_3, ER, and GR (Standard deviations in parentheses). Data simulated from DGP3.

N	T	*GD*				*NR*				*GD-NR*		*PCA*
		R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	sec.	R^2_{adj}
60	60	0.360 (0.066)	15.20 (1.83)	15.09 (1.86)	0.0046 (0.0007)	0.361 (0.068)	7.01 (0.10)	5.14 (1.52)	0.0092 (0.0013)	0.361 (0.068)	0.0058 (0.0009)	0.151 (0.047)
	120	0.370 (0.053)	15.21 (1.63)	15.15 (1.65)	0.0082 (0.0012)	0.375 (0.054)	7.07 (0.26)	5.17 (1.40)	0.0155 (0.0019)	0.375 (0.054)	0.0102 (0.0013)	0.163 (0.045)
	240	0.373 (0.045)	15.16 (1.56)	15.11 (1.60)	0.0149 (0.0019)	0.380 (0.047)	7.22 (0.45)	5.42 (1.34)	0.0280 (0.0031)	0.378 (0.073)	0.0187 (0.0021)	0.171 (0.042)
120	60	0.369 (0.056)	15.19 (1.49)	15.11 (1.54)	0.0080 (0.0011)	0.373 (0.057)	7.00 (0.09)	5.19 (1.40)	0.0139 (0.0016)	0.373 (0.057)	0.0098 (0.0012)	0.199 (0.038)
	120	0.380 (0.046)	15.03 (1.37)	14.98 (1.38)	0.0154 (0.0018)	0.386 (0.047)	7.07 (0.25)	5.46 (1.27)	0.0238 (0.0024)	0.385 (0.047)	0.0184 (0.0019)	0.214 (0.027)
	240	0.384 (0.036)	14.95 (1.24)	14.93 (1.26)	0.0289 (0.0028)	0.391 (0.038)	7.16 (0.39)	5.69 (1.22)	0.0430 (0.0038)	0.391 (0.038)	0.0344 (0.0030)	0.220 (0.022)
240	60	0.373 (0.054)	14.98 (1.33)	14.93 (1.34)	0.0144 (0.0017)	0.378 (0.055)	7.00 (0.12)	5.51 (1.29)	0.0230 (0.0023)	0.378 (0.055)	0.0174 (0.0018)	0.224 (0.024)
	120	0.384 (0.040)	14.90 (1.20)	14.88 (1.20)	0.0287 (0.0027)	0.390 (0.041)	7.04 (0.21)	5.79 (1.15)	0.0394 (0.0031)	0.389 (0.041)	0.0337 (0.0029)	0.233 (0.019)
	240	0.388 (0.031)	14.88 (1.09)	14.87 (1.10)	0.0608 (0.0045)	0.395 (0.032)	7.14 (0.39)	6.01 (1.12)	0.0778 (0.0061)	0.394 (0.036)	0.0700 (0.0049)	0.238 (0.016)

Table S.6: Average R^2_{adj}, total/R^2_{adj}-optimal iteration counts $\mathcal{I}/\mathcal{I}^*$, and computation times in seconds (Standard deviations in parentheses). Data simulated from DGP3.

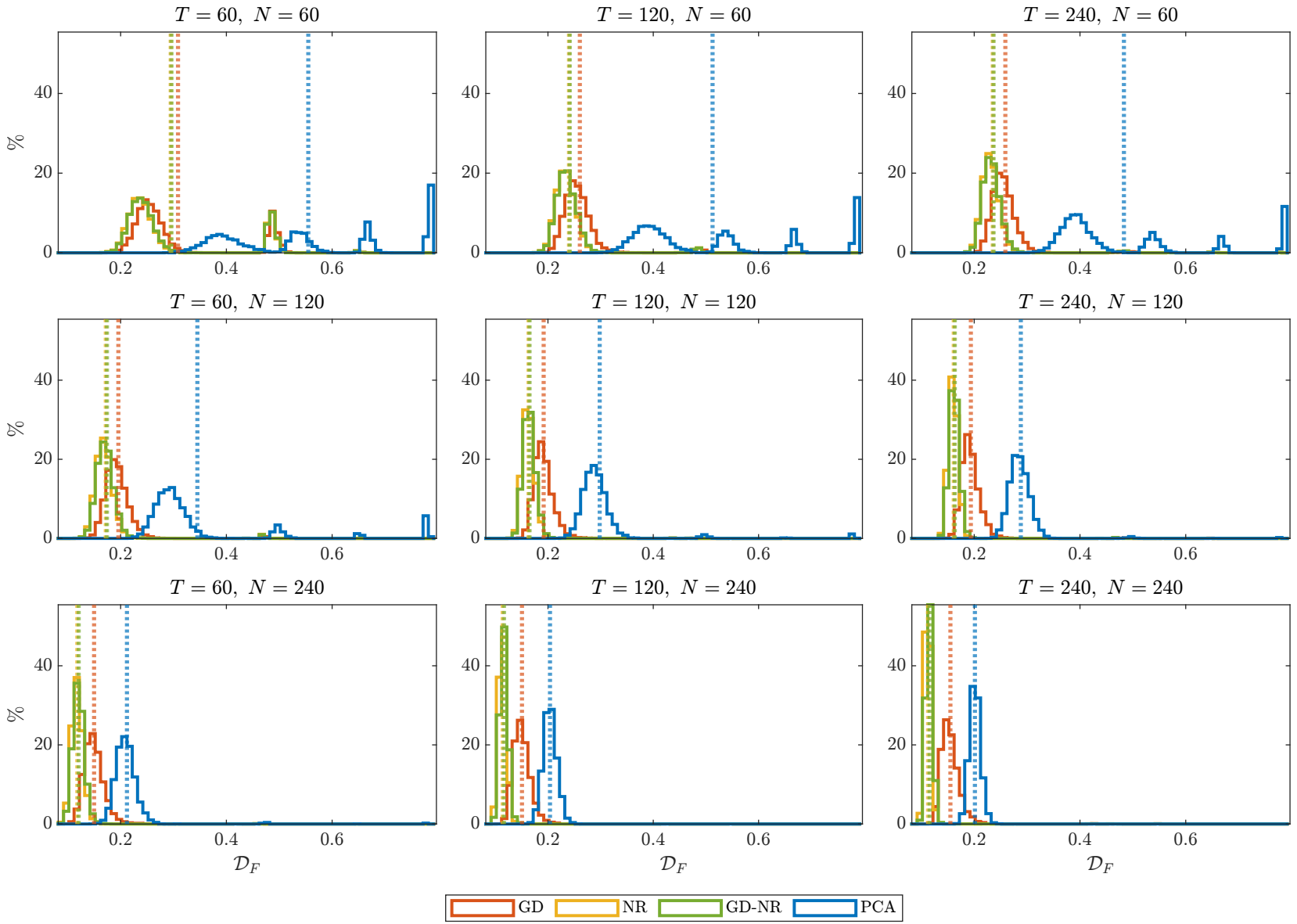

Figure S.5: $\mathcal{D}_F$ histogram (average values as dotted lines). Data simulated from DGP3.

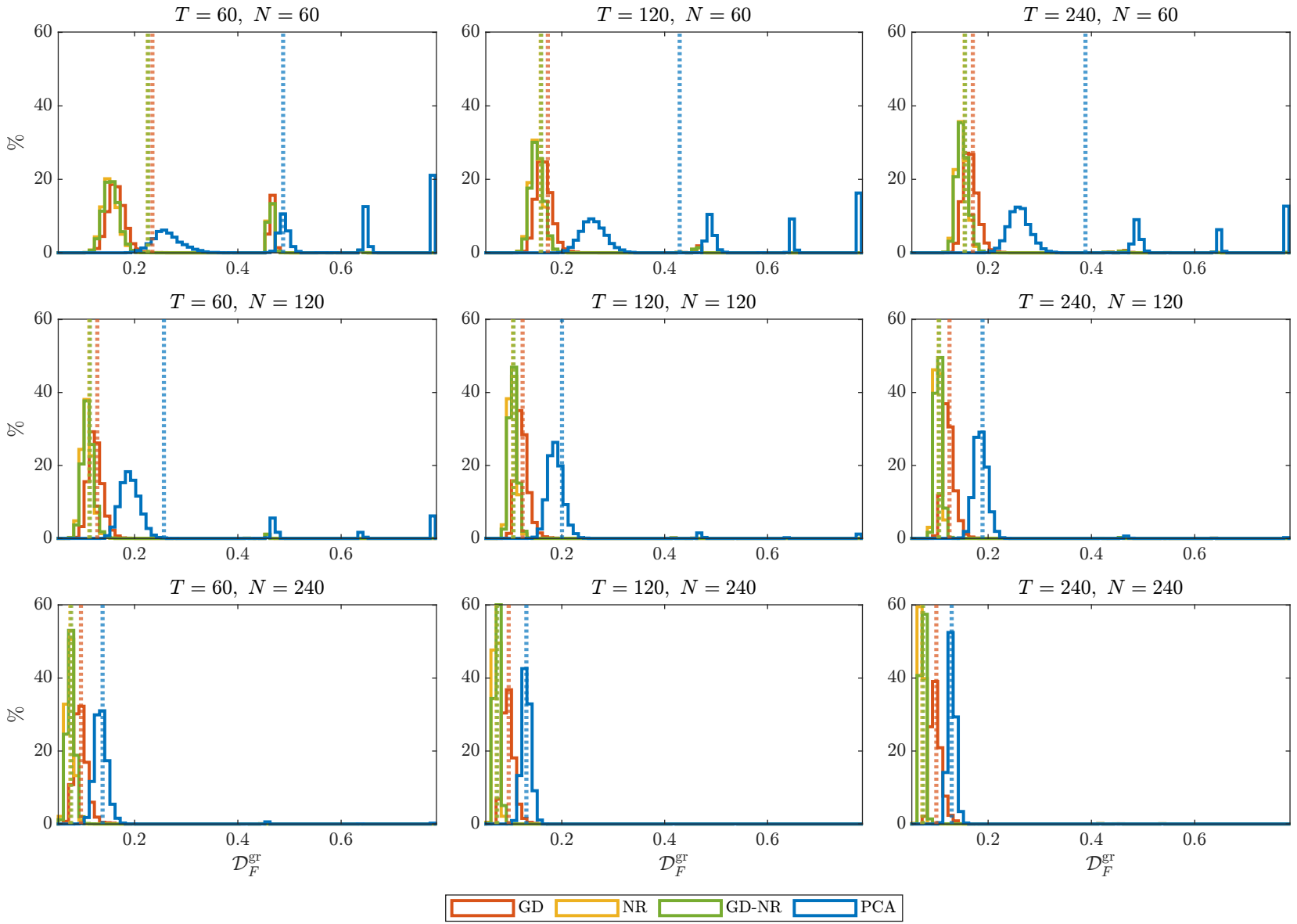

Figure S.6: $\mathcal{D}_F^{\text{gr}}$ histogram (average values as dotted lines). Data simulated from DGP3.

N	T	R^2_{adj}	BN_1	BN_2	BN_3	ER	GR
60	60	3.48 (0.77)	2.54 (0.52)	2.23 (0.53)	3.33 (0.64)	1.69 (0.47)	1.75 (0.43)
	120	3.33 (0.62)	2.75 (0.45)	2.61 (0.50)	2.98 (0.26)	1.79 (0.40)	1.84 (0.36)
	240	3.26 (0.57)	2.85 (0.39)	2.79 (0.43)	2.95 (0.31)	1.86 (0.35)	1.90 (0.30)
120	60	3.33 (0.59)	2.77 (0.42)	2.61 (0.49)	2.99 (0.19)	1.89 (0.31)	1.93 (0.25)
	120	3.49 (0.70)	2.96 (0.22)	2.86 (0.36)	3.14 (0.39)	1.97 (0.18)	1.99 (0.12)
	240	3.57 (0.75)	3.01 (0.22)	2.99 (0.23)	3.06 (0.31)	1.99 (0.10)	1.99 (0.07)
240	60	3.23 (0.48)	2.89 (0.32)	2.83 (0.37)	2.98 (0.16)	1.99 (0.11)	1.99 (0.08)
	120	3.55 (0.69)	3.00 (0.11)	2.99 (0.12)	3.03 (0.20)	2.00 (0.02)	2.00 (0.01)
	240	3.72 (0.78)	3.02 (0.22)	3.01 (0.20)	3.26 (0.52)	2.00 (0.00)	2.00 (0.00)

Table S.7: Average number of extracted factors using selection criteria R^2_{adj}, BN_1, BN_2, BN_3, ER, and GR (Standard deviations in parentheses). Data simulated from DGP4.1.

N	T	*GD*				*NR*				*GD-NR*		*PCA*
		R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	sec.	R^2_{adj}
60	60	0.246 (0.084)	13.41 (2.22)	13.25 (2.27)	0.0040 (0.0007)	0.239 (0.086)	7.07 (0.27)	6.07 (1.47)	0.0089 (0.0013)	0.239 (0.086)	0.0052 (0.0009)	0.129 (0.044)
	120	0.262 (0.081)	13.99 (2.08)	13.91 (2.13)	0.0074 (0.0012)	0.262 (0.083)	7.24 (0.44)	6.22 (1.41)	0.0152 (0.0020)	0.262 (0.083)	0.0093 (0.0014)	0.137 (0.043)
	240	0.269 (0.076)	14.50 (2.18)	14.46 (2.20)	0.0139 (0.0022)	0.272 (0.078)	7.50 (0.61)	6.58 (1.35)	0.0281 (0.0034)	0.272 (0.078)	0.0175 (0.0024)	0.142 (0.042)
120	60	0.256 (0.081)	13.88 (2.03)	13.76 (2.07)	0.0072 (0.0011)	0.255 (0.083)	7.05 (0.23)	5.94 (1.42)	0.0136 (0.0016)	0.255 (0.083)	0.0090 (0.0013)	0.150 (0.040)
	120	0.271 (0.075)	14.06 (1.76)	13.99 (1.79)	0.0141 (0.0019)	0.273 (0.076)	7.21 (0.42)	6.26 (1.28)	0.0234 (0.0025)	0.273 (0.076)	0.0171 (0.0021)	0.160 (0.037)
	240	0.279 (0.070)	14.54 (2.13)	14.51 (2.14)	0.0272 (0.0038)	0.283 (0.072)	7.48 (0.62)	6.66 (1.25)	0.0431 (0.0046)	0.269 (0.961)	0.0326 (0.0040)	0.164 (0.036)
240	60	0.260 (0.077)	14.06 (1.81)	13.95 (1.84)	0.0132 (0.0018)	0.261 (0.078)	7.06 (0.23)	6.03 (1.29)	0.0226 (0.0023)	0.261 (0.078)	0.0162 (0.0021)	0.161 (0.036)
	120	0.275 (0.071)	14.14 (1.65)	14.08 (1.68)	0.0266 (0.0031)	0.279 (0.072)	7.21 (0.42)	6.37 (1.18)	0.0388 (0.0036)	0.279 (0.072)	0.0315 (0.0032)	0.170 (0.034)
	240	0.283 (0.066)	14.46 (2.04)	14.44 (2.05)	0.0574 (0.0070)	0.288 (0.068)	7.48 (0.59)	6.76 (1.14)	0.0776 (0.0072)	0.287 (0.068)	0.0662 (0.0072)	0.173 (0.033)

Table S.8: Average R^2_{adj}, total/R^2_{adj}-optimal iteration counts $\mathcal{I}/\mathcal{I}^*$, and computation times in seconds (Standard deviations in parentheses). Data simulated from DGP4.1.

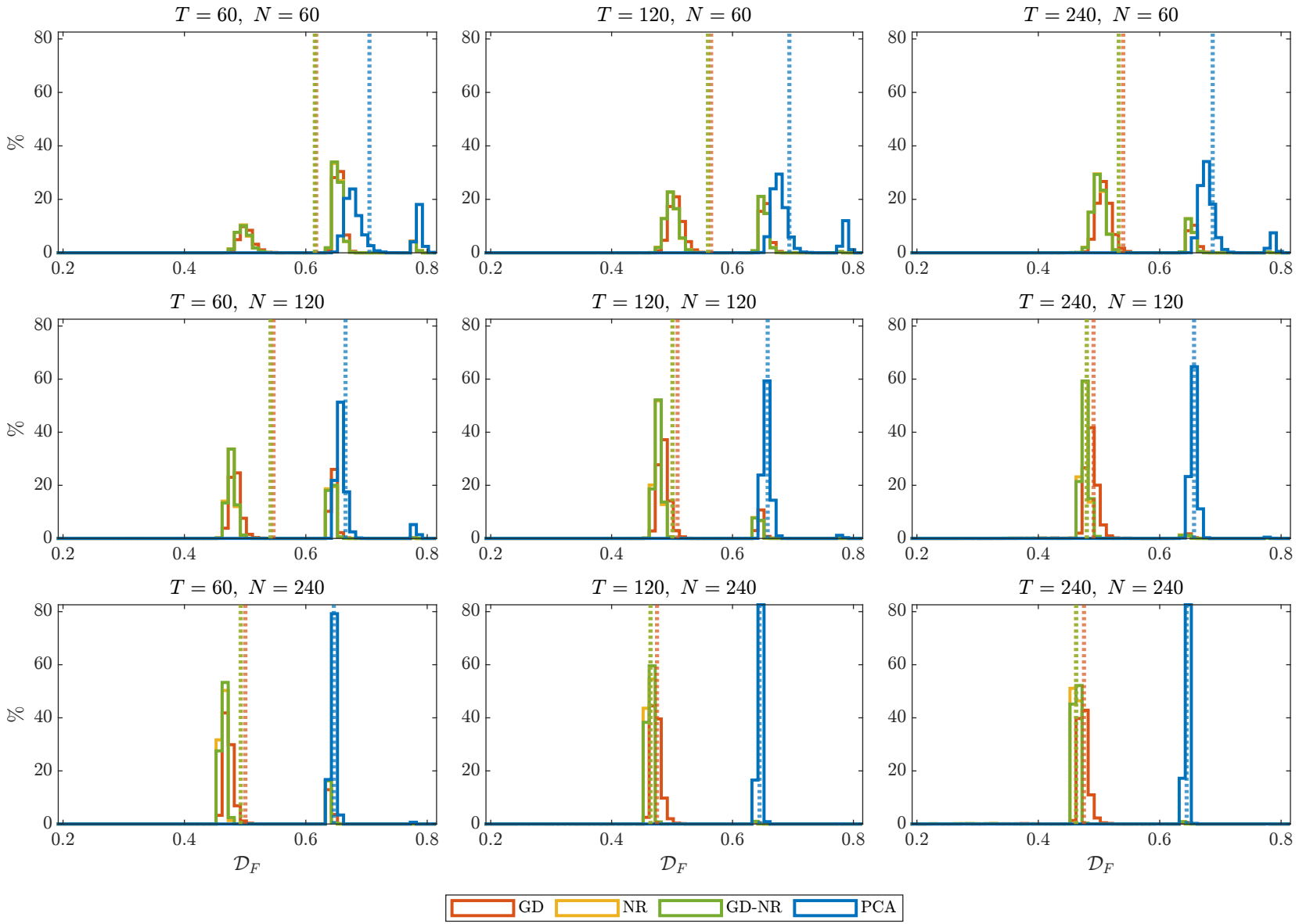

Figure S.7: $\mathcal{D}_F$ histogram (average values as dotted lines). Data simulated from DGP4.1.

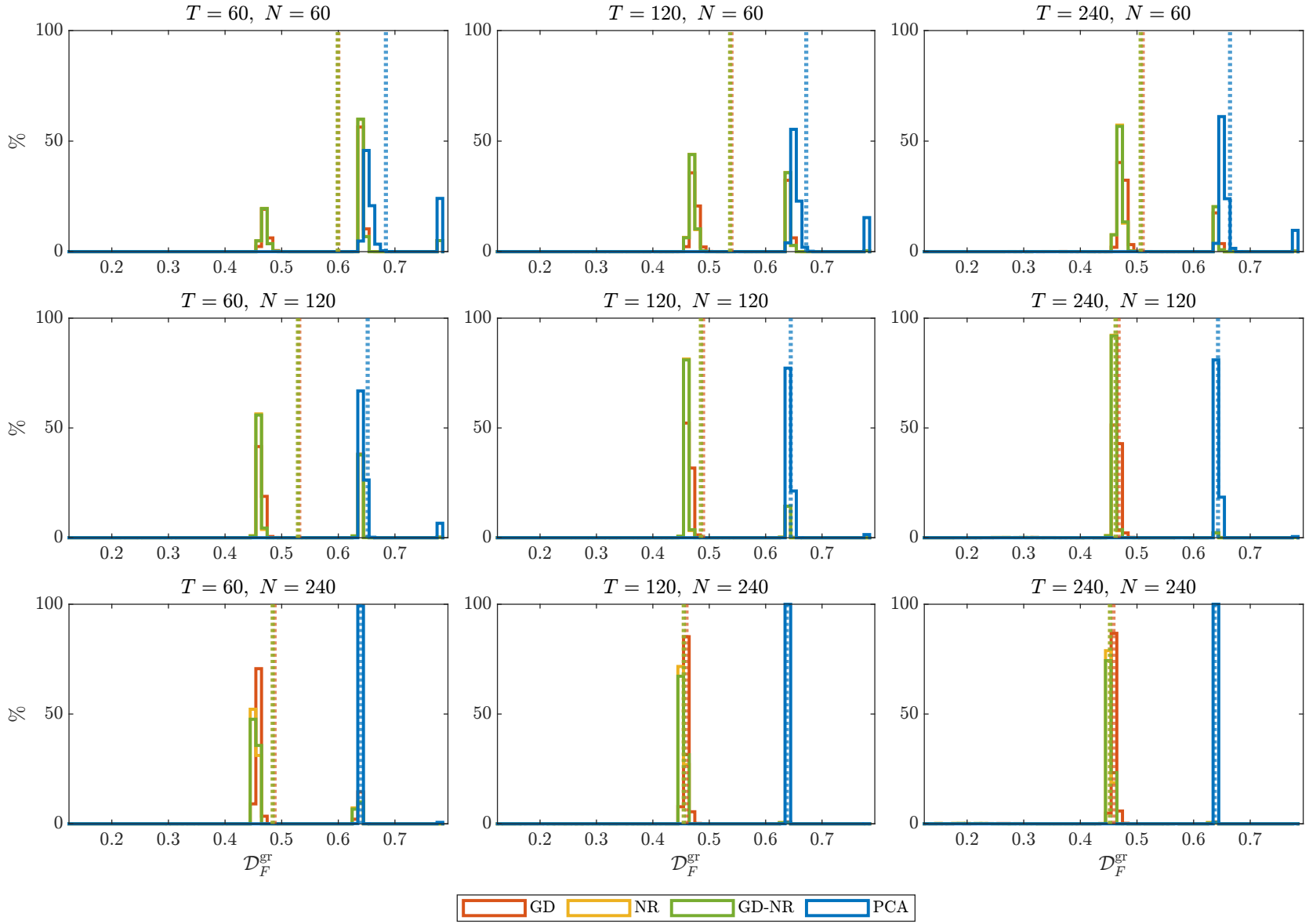

Figure S.8: $\mathcal{D}_F^{\text{gr}}$ histogram (average values as dotted lines). Data simulated from DGP4.1.

N	T	R^2_{adj}	BN_1	BN_2	BN_3	ER	GR
60	60	3.81 (1.08)	2.42 (0.68)	2.12 (0.67)	3.71 (1.05)	1.69 (0.47)	1.75 (0.43)
	120	3.99 (1.08)	2.61 (0.69)	2.47 (0.68)	3.12 (0.83)	1.79 (0.40)	1.84 (0.36)
	240	4.08 (1.12)	2.75 (0.74)	2.68 (0.73)	2.99 (0.80)	1.86 (0.35)	1.90 (0.30)
120	60	3.86 (1.02)	2.62 (0.64)	2.49 (0.63)	3.07 (0.77)	1.89 (0.31)	1.93 (0.25)
	120	4.09 (1.03)	2.92 (0.69)	2.74 (0.65)	3.71 (0.94)	1.97 (0.18)	1.99 (0.12)
	240	4.14 (1.03)	3.12 (0.72)	3.02 (0.69)	3.50 (0.82)	1.99 (0.10)	1.99 (0.07)
240	60	3.81 (0.95)	2.76 (0.61)	2.70 (0.61)	2.96 (0.65)	1.99 (0.11)	1.99 (0.08)
	120	4.13 (1.02)	3.09 (0.68)	3.00 (0.66)	3.45 (0.79)	2.00 (0.02)	2.00 (0.01)
	240	4.22 (1.02)	3.34 (0.70)	3.23 (0.67)	3.85 (0.92)	2.00 (0.00)	2.00 (0.00)

Table S.9: Average number of extracted factors using selection criteria R^2_{adj}, BN_1, BN_2, BN_3, ER, and GR (Standard deviations in parentheses). Data simulated from DGP4.2.

N	T	*GD*				*NR*				*GD-NR*		*PCA*
		R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	sec.	R^2_{adj}
60	60	0.237 (0.083)	19.54 (5.68)	19.46 (5.74)	0.0055 (0.0027)	0.233 (0.087)	9.32 (1.28)	8.89 (1.73)	0.0123 (0.0051)	0.231 (0.087)	0.0067 (0.0027)	0.129 (0.044)
	120	0.252 (0.081)	21.30 (5.97)	21.23 (6.03)	0.0100 (0.0028)	0.256 (0.084)	9.35 (1.20)	8.85 (1.65)	0.0185 (0.0031)	0.253 (0.084)	0.0119 (0.0030)	0.137 (0.043)
	240	0.259 (0.077)	22.69 (6.18)	22.62 (6.25)	0.0199 (0.0055)	0.265 (0.079)	9.39 (1.17)	8.78 (1.76)	0.0337 (0.0050)	0.263 (0.080)	0.0235 (0.0056)	0.142 (0.042)
120	60	0.247 (0.081)	20.27 (5.36)	20.21 (5.40)	0.0096 (0.0025)	0.249 (0.082)	9.09 (1.13)	8.62 (1.62)	0.0170 (0.0033)	0.248 (0.083)	0.0113 (0.0027)	0.150 (0.040)
	120	0.262 (0.074)	21.86 (5.44)	21.82 (5.47)	0.0189 (0.0047)	0.267 (0.076)	9.04 (1.02)	8.48 (1.57)	0.0271 (0.0039)	0.266 (0.077)	0.0218 (0.0047)	0.160 (0.037)
	240	0.270 (0.071)	23.58 (5.36)	23.53 (5.40)	0.0387 (0.0086)	0.279 (0.072)	8.98 (0.92)	8.37 (1.49)	0.0480 (0.0056)	0.278 (0.073)	0.0439 (0.0087)	0.164 (0.036)
240	60	0.252 (0.076)	20.66 (4.98)	20.61 (5.02)	0.0181 (0.0043)	0.256 (0.077)	8.86 (1.03)	8.31 (1.59)	0.0283 (0.0065)	0.255 (0.078)	0.0212 (0.0044)	0.161 (0.036)
	120	0.267 (0.071)	22.54 (4.95)	22.51 (4.96)	0.0377 (0.0080)	0.274 (0.071)	8.64 (0.85)	8.04 (1.46)	0.0444 (0.0067)	0.274 (0.072)	0.0425 (0.0080)	0.170 (0.034)
	240	0.275 (0.067)	24.08 (4.79)	24.04 (4.81)	0.0785 (0.0149)	0.285 (0.068)	8.65 (0.78)	8.08 (1.32)	0.0802 (0.0087)	0.285 (0.068)	0.0872 (0.0152)	0.173 (0.033)

Table S.10: Average R^2_{adj}, total/R^2_{adj}-optimal iteration counts $\mathcal{I}/\mathcal{I}^*$, and computation times in seconds (Standard deviations in parentheses). Data simulated from DGP4.2.

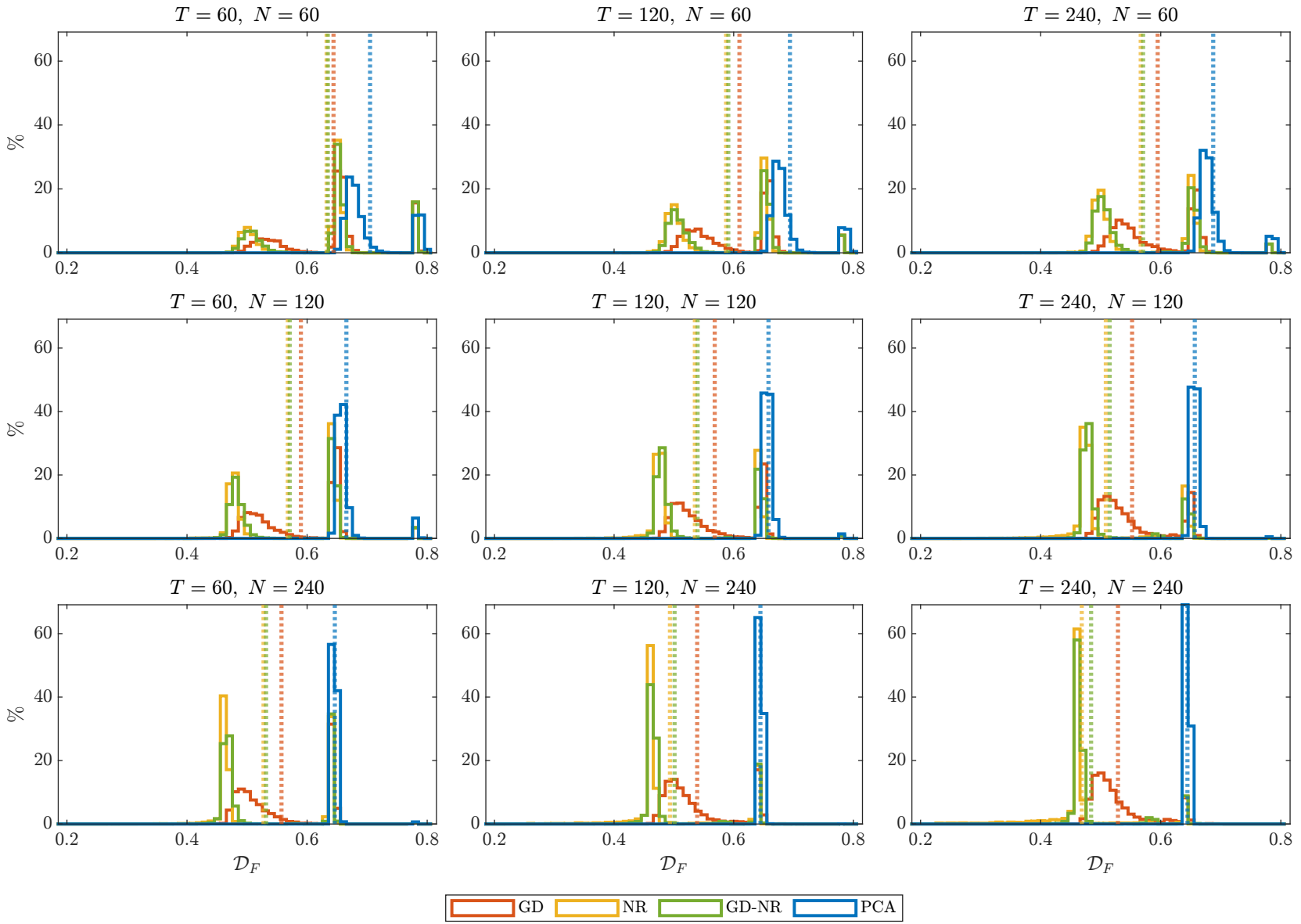

Figure S.9: $\mathcal{D}_F$ histogram (average values as dotted lines). Data simulated from DGP4.2.

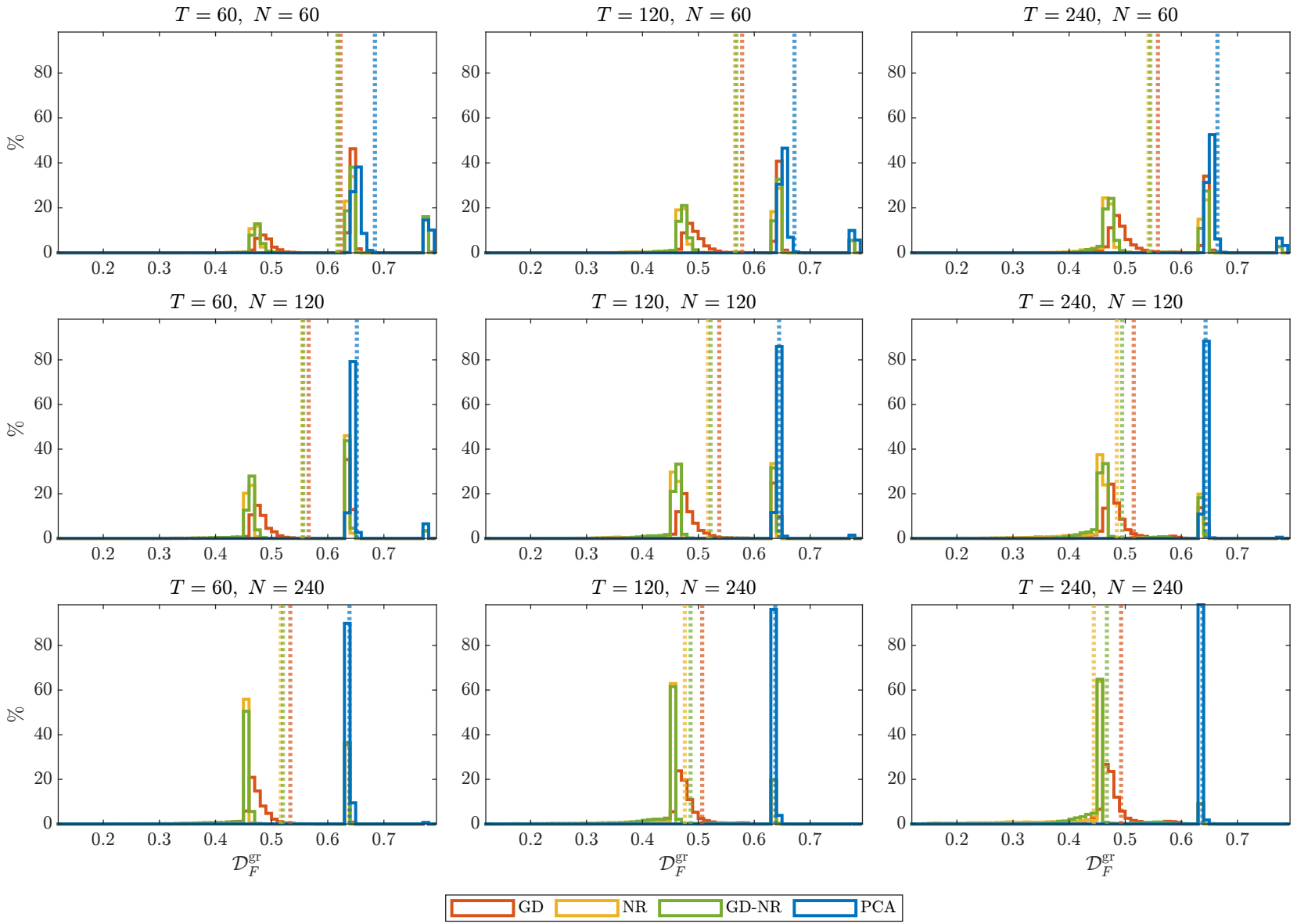

Figure S.10: $\mathcal{D}_F^{\text{gr}}$ histogram (average values as dotted lines). Data simulated from DGP4.2.

N	T	*GD*				*NR*				*GD-NR*		*PCA*
		R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	sec.	R^2_{adj}
60	60	0.244 (0.083)	16.24 (2.21)	16.13 (2.28)	0.0049 (0.0008)	0.249 (0.084)	7.09 (0.29)	6.98 (0.60)	0.0093 (0.0014)	0.248 (0.084)	0.0062 (0.0010)	0.132 (0.044)
	120	0.260 (0.080)	15.93 (2.35)	15.85 (2.39)	0.0087 (0.0016)	0.266 (0.082)	7.27 (0.47)	6.84 (1.09)	0.0162 (0.0027)	0.265 (0.082)	0.0108 (0.0018)	0.134 (0.044)
	240	0.266 (0.075)	16.28 (2.90)	16.24 (2.93)	0.0159 (0.0028)	0.274 (0.078)	7.55 (0.66)	6.63 (1.53)	0.0295 (0.0038)	0.264 (0.194)	0.0198 (0.0030)	0.136 (0.046)
120	60	0.255 (0.081)	15.90 (2.09)	15.79 (2.13)	0.0084 (0.0013)	0.259 (0.082)	7.07 (0.25)	6.96 (0.55)	0.0141 (0.0017)	0.259 (0.082)	0.0102 (0.0014)	0.150 (0.038)
	120	0.270 (0.074)	15.73 (2.26)	15.66 (2.29)	0.0160 (0.0023)	0.275 (0.075)	7.24 (0.44)	6.84 (1.01)	0.0246 (0.0027)	0.274 (0.075)	0.0191 (0.0025)	0.156 (0.036)
	240	0.275 (0.068)	16.29 (3.13)	16.25 (3.15)	0.0309 (0.0053)	0.284 (0.072)	7.54 (0.67)	6.76 (1.42)	0.0455 (0.0052)	0.272 (0.208)	0.0365 (0.0056)	0.157 (0.036)
240	60	0.258 (0.076)	15.79 (2.04)	15.66 (2.05)	0.0150 (0.0021)	0.263 (0.077)	7.07 (0.25)	7.00 (0.43)	0.0234 (0.0025)	0.262 (0.077)	0.0182 (0.0023)	0.160 (0.036)
	120	0.274 (0.070)	15.78 (2.40)	15.70 (2.43)	0.0299 (0.0042)	0.280 (0.072)	7.26 (0.47)	7.03 (0.81)	0.0409 (0.0040)	0.279 (0.071)	0.0351 (0.0044)	0.168 (0.033)
	240	0.279 (0.064)	16.42 (3.38)	16.39 (3.40)	0.0653 (0.0112)	0.289 (0.068)	7.58 (0.69)	7.30 (1.02)	0.0821 (0.0084)	0.270 (0.263)	0.0748 (0.0116)	0.170 (0.033)

Table S.11: Average R^2_{adj}, total/R^2_{adj}-optimal iteration counts $\mathcal{I}/\mathcal{I}^*$, and computation times in seconds (Standard deviations in parentheses). Data simulated from DGP4.3. R^2_{adj} outliers omitted in *GD-NR*: $N = 60, T = 240$; $N = 120, T = 240$; $N = 240, T = 240$.

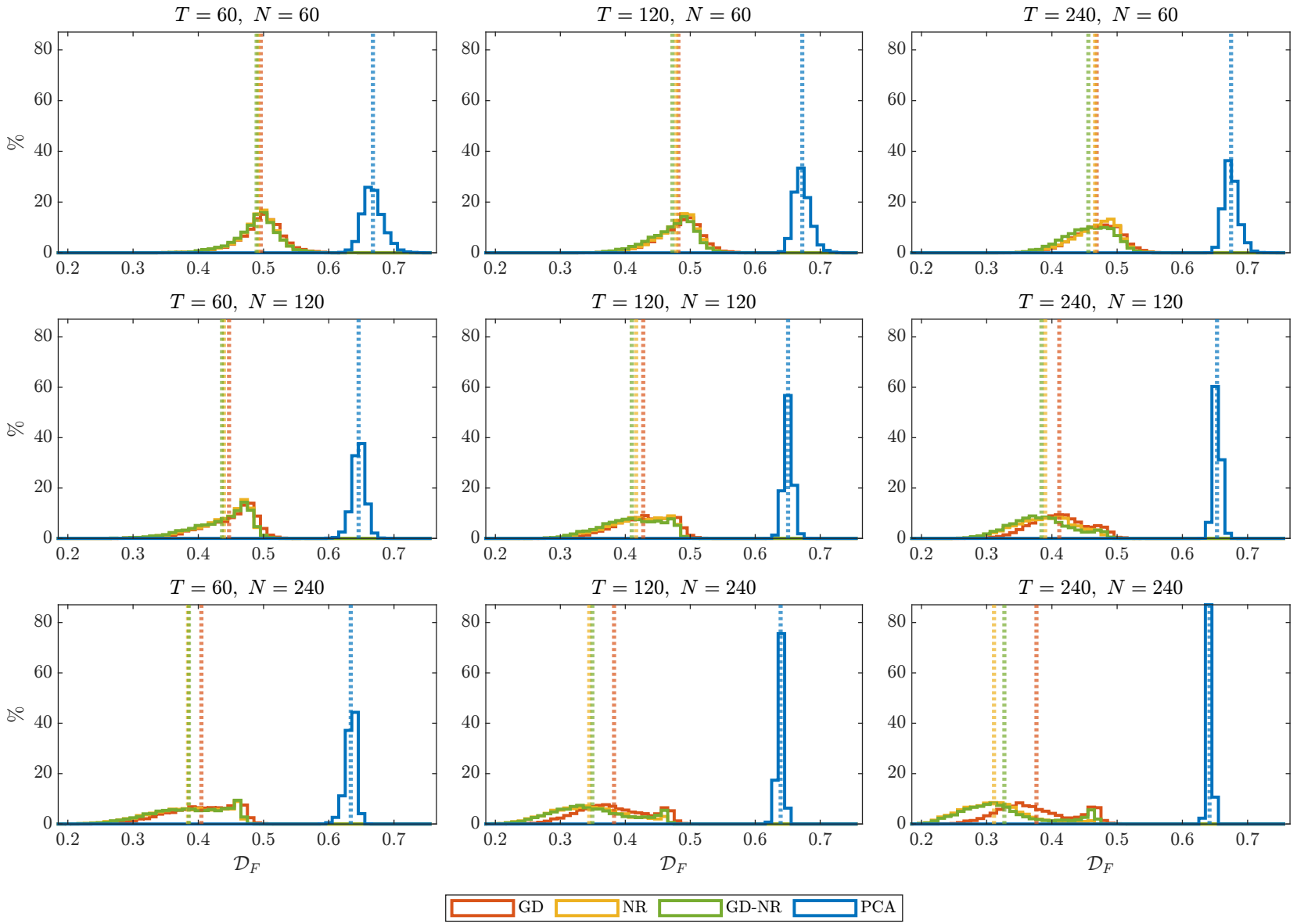

Figure S.11: $\mathcal{D}_F$ histogram (average values as dotted lines). Data simulated from DGP4.3.

N	T	R^2_{adj}	BN_1	BN_2	BN_3	ER	GR
60	60	5.91 (0.30)	5.25 (0.75)	4.69 (0.78)	5.90 (0.31)	2.54 (1.22)	2.87 (1.18)
	120	5.87 (0.37)	5.19 (0.81)	4.92 (0.83)	5.70 (0.56)	2.87 (1.23)	3.21 (1.10)
	240	5.61 (0.74)	4.86 (0.89)	4.74 (0.87)	5.18 (0.88)	3.07 (1.20)	3.40 (1.01)
120	60	5.94 (0.25)	5.35 (0.74)	5.04 (0.81)	5.83 (0.40)	3.46 (1.03)	3.71 (0.76)
	120	5.85 (0.41)	5.19 (0.83)	4.79 (0.83)	5.80 (0.48)	3.84 (0.61)	3.94 (0.37)
	240	5.35 (0.85)	4.70 (0.86)	4.54 (0.80)	5.11 (0.90)	3.96 (0.30)	3.98 (0.18)
240	60	5.94 (0.24)	5.20 (0.80)	5.03 (0.83)	5.61 (0.62)	3.94 (0.39)	3.99 (0.14)
	120	5.72 (0.57)	4.88 (0.85)	4.66 (0.80)	5.42 (0.75)	4.00 (0.04)	4.00 (0.00)
	240	5.00 (0.85)	4.38 (0.66)	4.24 (0.54)	4.87 (0.84)	4.00 (0.00)	4.00 (0.00)

Table S.12: Average number of extracted factors using selection criteria R^2_{adj}, BN_1, BN_2, BN_3, ER, and GR (Standard deviations in parentheses). Data simulated from DGP5.

N	T	*GD*				*NR*				*GD-NR*		*PCA*
		R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	sec.	R^2_{adj}
60	60	0.456 (0.092)	16.50 (2.38)	16.45 (2.39)	0.0052 (0.0012)	0.458 (0.098)	7.13 (0.34)	7.00 (0.65)	0.0100 (0.0023)	0.455 (0.097)	0.0065 (0.0013)	0.189 (0.061)
	120	0.463 (0.079)	16.90 (2.68)	16.86 (2.68)	0.0093 (0.0017)	0.473 (0.085)	7.28 (0.47)	7.18 (0.69)	0.0166 (0.0023)	0.466 (0.225)	0.0114 (0.0019)	0.208 (0.059)
	240	0.460 (0.073)	16.86 (2.99)	16.83 (2.99)	0.0169 (0.0032)	0.472 (0.078)	7.39 (0.56)	7.28 (0.77)	0.0300 (0.0041)	0.463 (0.180)	0.0208 (0.0035)	0.220 (0.057)
120	60	0.445 (0.084)	16.78 (2.47)	16.71 (2.46)	0.0090 (0.0016)	0.450 (0.090)	7.07 (0.26)	6.85 (0.75)	0.0148 (0.0021)	0.447 (0.089)	0.0109 (0.0017)	0.242 (0.050)
	120	0.446 (0.075)	16.68 (3.01)	16.63 (2.98)	0.0170 (0.0030)	0.453 (0.080)	7.21 (0.43)	6.86 (0.96)	0.0252 (0.0030)	0.450 (0.079)	0.0202 (0.0032)	0.263 (0.042)
	240	0.444 (0.067)	16.39 (3.23)	16.33 (3.19)	0.0318 (0.0060)	0.454 (0.071)	7.26 (0.49)	6.84 (1.05)	0.0449 (0.0049)	0.451 (0.072)	0.0376 (0.0063)	0.272 (0.039)
240	60	0.435 (0.083)	16.79 (2.69)	16.69 (2.65)	0.0162 (0.0028)	0.441 (0.088)	7.05 (0.25)	6.74 (0.86)	0.0243 (0.0026)	0.438 (0.087)	0.0194 (0.0029)	0.269 (0.040)
	120	0.433 (0.071)	16.40 (3.13)	16.33 (3.07)	0.0316 (0.0057)	0.440 (0.075)	7.18 (0.43)	6.67 (1.10)	0.0416 (0.0044)	0.434 (0.164)	0.0369 (0.0059)	0.282 (0.036)
	240	0.433 (0.066)	15.51 (2.73)	15.46 (2.66)	0.0636 (0.0097)	0.441 (0.069)	7.11 (0.37)	6.36 (1.13)	0.0787 (0.0066)	0.439 (0.069)	0.0729 (0.0100)	0.292 (0.036)

Table S.13: Average R^2_{adj}, total/R^2_{adj}-optimal iteration counts $\mathcal{I}/\mathcal{I}^*$, and computation times in seconds (Standard deviations in parentheses). Data simulated from DGP5. R^2_{adj} outliers omitted in *GD-NR*: $N = 60, T = 240$; $N = 240, T = 240$.

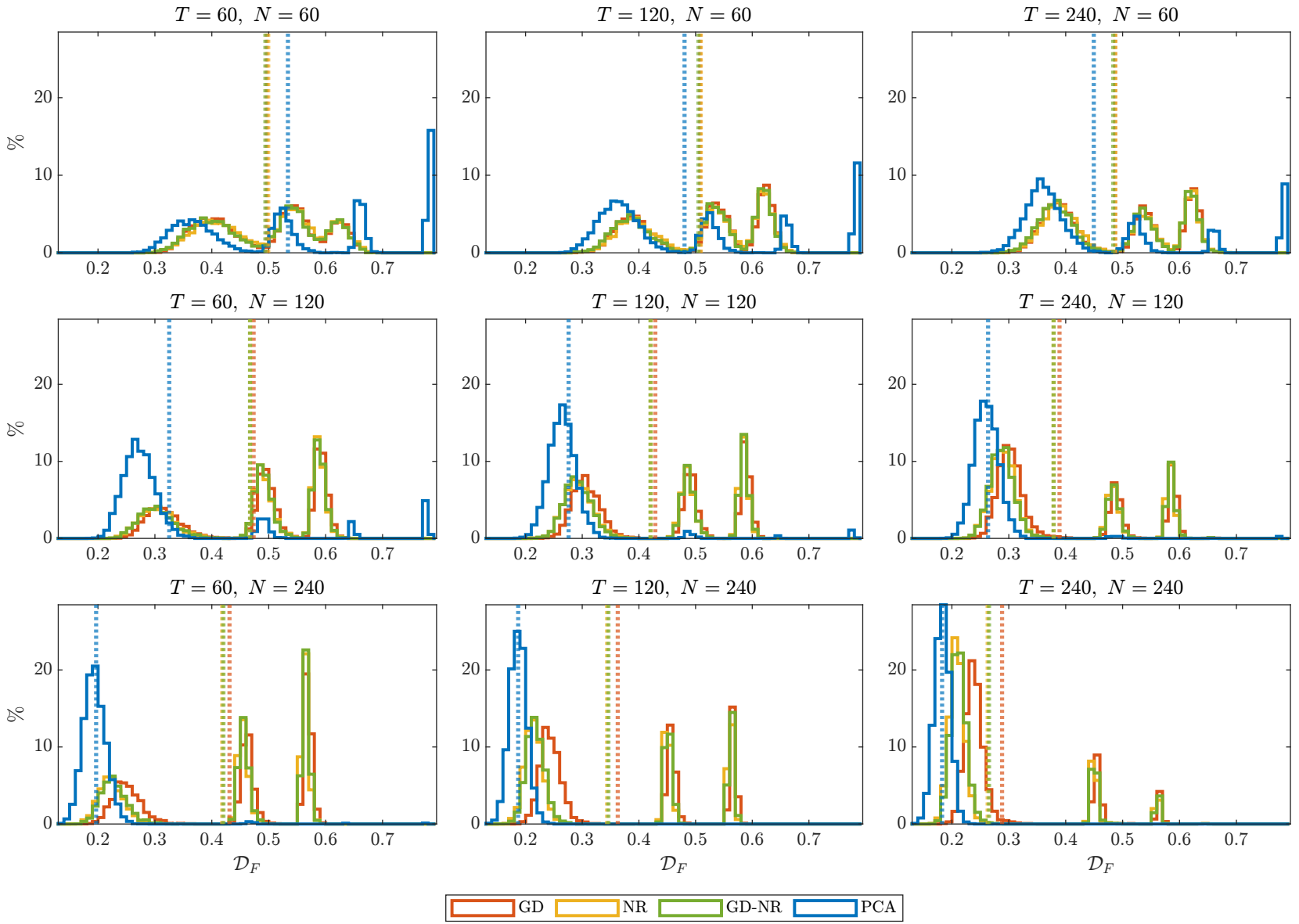

Figure S.12: $\mathcal{D}_F$ histogram (average values as dotted lines). Data simulated from DGP5.

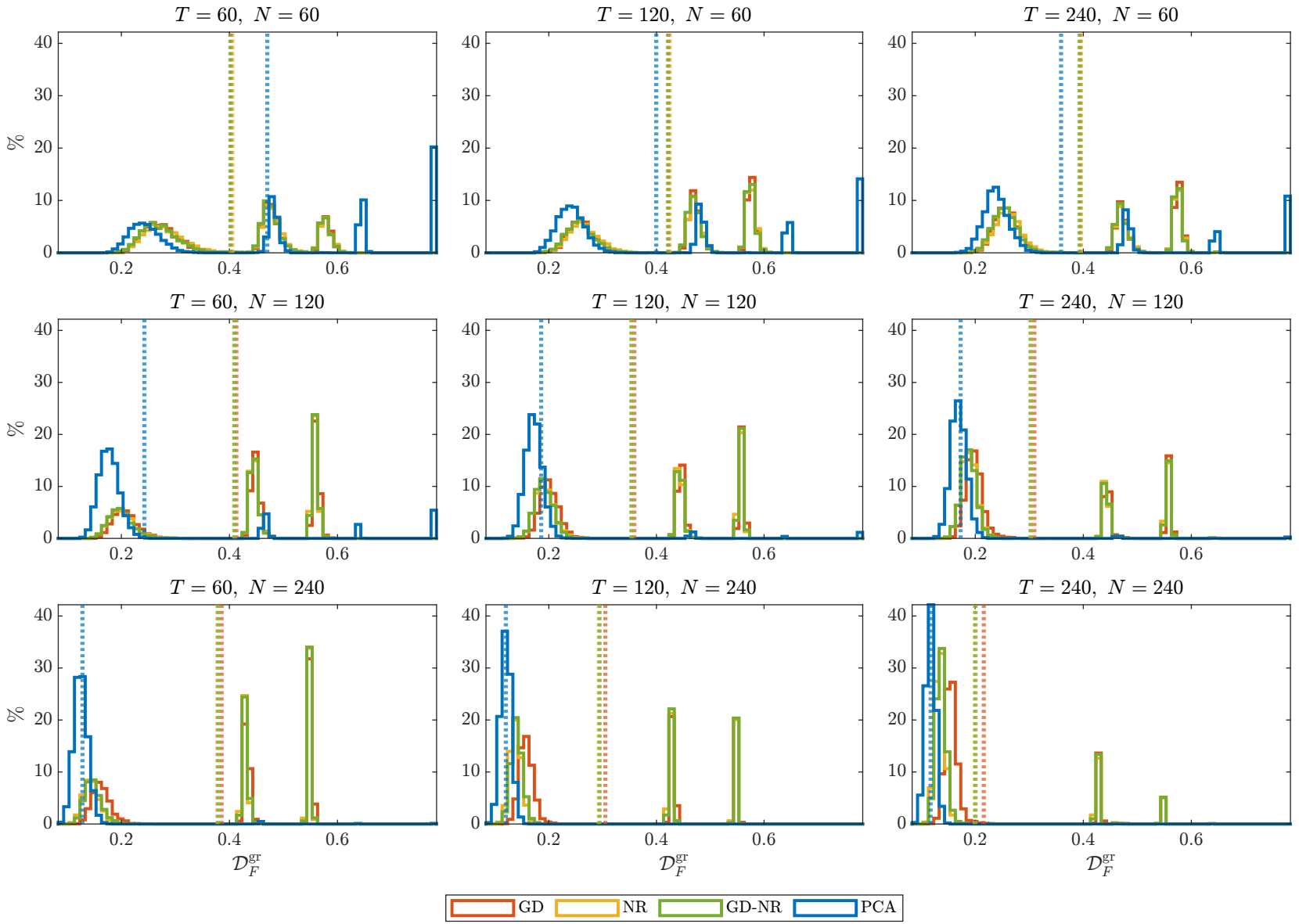

Figure S.13: $\mathcal{D}_F^{\text{gr}}$ histogram (average values as dotted lines). Data simulated from DGP5.

N	T	R^2_{adj}	BN_1	BN_2	BN_3	ER	GR
140	60	4.29 (0.56)	3.98 (0.21)	3.95 (0.25)	4.03 (0.22)	2.52 (1.21)	2.84 (1.18)
	120	4.31 (0.60)	4.02 (0.17)	4.01 (0.16)	4.11 (0.35)	2.90 (1.22)	3.22 (1.09)
	240	4.19 (0.54)	4.00 (0.24)	3.99 (0.24)	4.03 (0.30)	3.10 (1.18)	3.43 (0.98)
140	60	4.16 (0.43)	4.02 (0.15)	4.02 (0.13)	4.05 (0.24)	3.44 (1.05)	3.72 (0.75)
	120	4.23 (0.51)	4.05 (0.23)	4.03 (0.20)	4.14 (0.39)	3.85 (0.59)	3.95 (0.33)
	240	4.14 (0.42)	4.03 (0.24)	4.02 (0.23)	4.07 (0.32)	3.96 (0.33)	3.99 (0.18)
140	60	4.19 (0.45)	4.00 (0.06)	3.99 (0.08)	4.00 (0.07)	3.94 (0.36)	3.98 (0.18)
	120	4.19 (0.46)	4.00 (0.04)	4.00 (0.03)	4.03 (0.18)	4.00 (0.02)	4.00 (0.02)
	240	4.21 (0.49)	4.00 (0.05)	4.00 (0.04)	4.01 (0.12)	4.00 (0.00)	4.00 (0.00)

Table S.14: Average number of extracted factors using selection criteria R^2_{adj}, BN_1, BN_2, BN_3, ER, and GR (Standard deviations in parentheses). Data simulated from DGP6.

N	T	*GD*				*NR*				*GD-NR*		*PCA*
		R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	$\mathcal{I}$	$\mathcal{I}^*$	sec.	R^2_{adj}	sec.	R^2_{adj}
$\binom{20}{40}{80}$	60	0.440 (0.098)	14.95 (1.61)	14.87 (1.64)	0.0109 (0.0015)	0.445 (0.100)	7.01 (0.09)	5.82 (1.22)	0.0181 (0.0020)	0.444 (0.099)	0.0133 (0.0017)	0.188 (0.059)
	120	0.459 (0.085)	14.80 (1.67)	14.74 (1.69)	0.0213 (0.0026)	0.466 (0.087)	7.10 (0.31)	6.12 (1.15)	0.0307 (0.0029)	0.466 (0.087)	0.0252 (0.0027)	0.209 (0.058)
	240	0.469 (0.077)	14.70 (1.69)	14.67 (1.70)	0.0418 (0.0046)	0.479 (0.080)	7.27 (0.52)	6.45 (1.15)	0.0571 (0.0053)	0.477 (0.085)	0.0490 (0.0048)	0.220 (0.055)
$\binom{40}{80}{20}$	60	0.510 (0.092)	14.75 (1.60)	14.71 (1.60)	0.0092 (0.0013)	0.517 (0.094)	7.03 (0.16)	6.02 (1.13)	0.0162 (0.0020)	0.517 (0.094)	0.0113 (0.0015)	0.242 (0.051)
	120	0.531 (0.080)	14.62 (1.73)	14.59 (1.72)	0.0175 (0.0022)	0.541 (0.083)	7.21 (0.44)	6.50 (1.05)	0.0278 (0.0030)	0.540 (0.082)	0.0211 (0.0024)	0.264 (0.041)
	240	0.536 (0.071)	14.68 (1.85)	14.67 (1.85)	0.0344 (0.0043)	0.550 (0.074)	7.54 (0.63)	6.95 (1.08)	0.0525 (0.0054)	0.542 (0.140)	0.0407 (0.0045)	0.272 (0.039)
$\binom{80}{20}{40}$	60	0.371 (0.083)	14.43 (1.28)	14.38 (1.28)	0.0069 (0.0010)	0.377 (0.084)	6.49 (0.50)	4.48 (1.43)	0.0119 (0.0018)	0.377 (0.084)	0.0085 (0.0011)	0.269 (0.040)
	120	0.393 (0.076)	14.29 (1.09)	14.26 (1.11)	0.0132 (0.0014)	0.399 (0.078)	6.52 (0.50)	4.92 (1.32)	0.0208 (0.0026)	0.399 (0.077)	0.0160 (0.0016)	0.284 (0.037)
	240	0.403 (0.070)	14.39 (1.06)	14.37 (1.06)	0.0256 (0.0023)	0.410 (0.071)	6.62 (0.49)	5.19 (1.19)	0.0390 (0.0040)	0.410 (0.071)	0.0308 (0.0026)	0.292 (0.036)

Table S.15: Average R^2_{adj}, total/R^2_{adj}-optimal iteration counts $\mathcal{I}/\mathcal{I}^*$, and computation times in seconds (Standard deviations in parentheses). Data simulated from DGP6. R^2_{adj} outliers omitted in *GD-NR*: $\mathbf{n} = (20, 40, 80), T = 240$; $\mathbf{n} = (40, 80, 20), T = 240$.

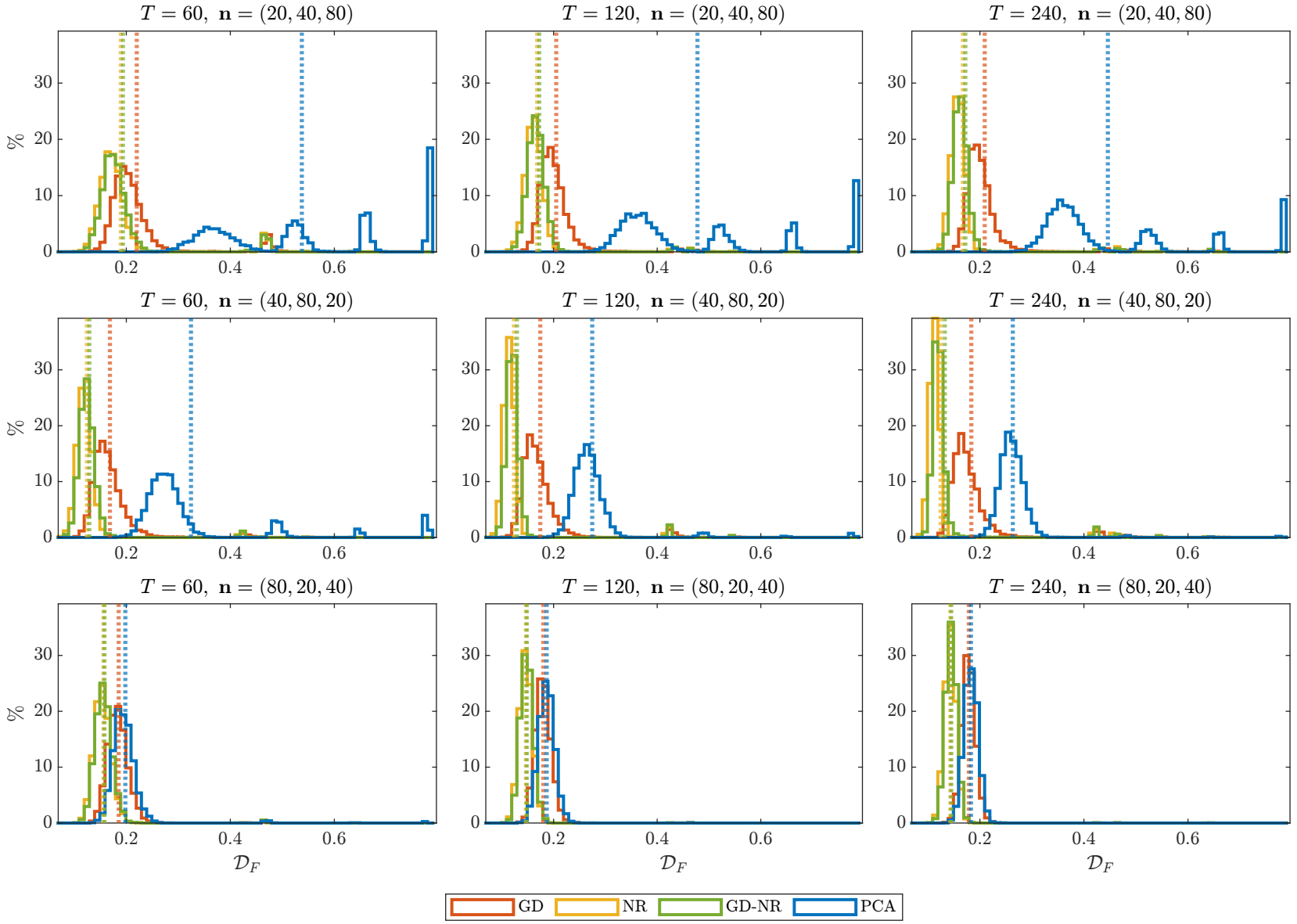

Figure S.14: $\mathcal{D}_F$ histogram (average values as dotted lines). Data simulated from DGP6.

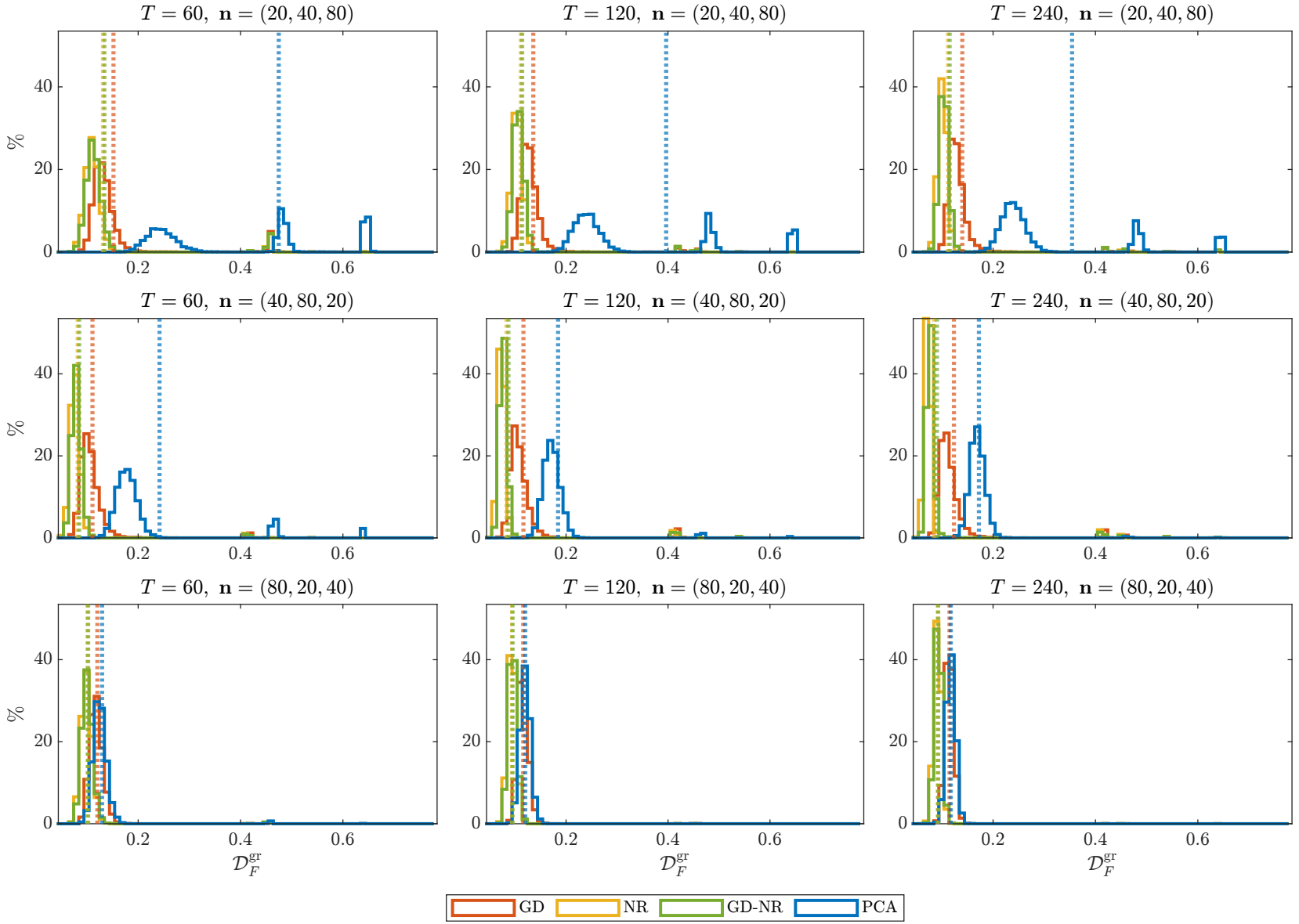

Figure S.15: $\mathcal{D}_F^{\mathrm{gr}}$ histogram (average values as dotted lines). Data simulated from DGP6.

S.2 Empirical Application - Supporting Material

Table S.16: List of variables providing a short description of the variables and the data source (GW: Welch and Goyal (2008), https://sites.google.com/view/agoyal145; GW2: Goyal *et al.* (2024), https://sites.google.com/view/agoyal145; JR: Ibbotson *et al.* (1994), https://site.warrington.ufl.edu/ritter/ipo-data; YF: Yahoo Finance, https://finance.yahoo.com/quote/^GSPC). ρ is an AR(1) coefficient estimate, tcode the transformation code in the notation of McCracken and Ng (2016) (1: no transformation; 2: Δx_t; 5: $\Delta \log(x_t)$) with ρ_{tcode} representing the post-transformation AR(1) coefficient estimate. $\bar{X}$ and S_X denote the post-transformation sample average and standard deviations.

	Description; Data source	ρ	tcode	ρ_{tcode}	$\bar{X}$	S_X
D12	12-month moving sum of dividend; GW	1.0074	5	0.9360	0.0049	0.0059
dp	Dividend-price ratio; GW2	0.9908	2	0.0556	-0.0022	0.0434
dy	Dividend yield; GW2	0.9907	2	0.0496	-0.0022	0.0434
E12	12-month moving sum of earnings; GW	1.0052	5	0.7456	0.0047	0.0600
ep	Earnings-price ratio; GW2	0.9827	2	0.4049	-0.0024	0.0695
e10p	Earnings' ten-year moving average divided by price; GW	0.9896	2	0.0369	-0.0020	0.0430
ep10	Earnings divided by prices' ten-year moving average; GW	0.9908	2	0.7468	-0.0016	0.0602
de	Dividend-earnings ratio; GW2	0.9848	2	0.7496	0.0002	0.0610
svar	Stock market variance; GW	0.4545	1	0.4545	0.0025	0.0051
skvw	Average stock skewness; GW2	0.0789	1	0.0789	0.0298	0.0439
tail	Tail risk from cross section; GW2	0.8072	1	0.8072	0.4521	0.0292
fbmlr	Single factor from book-to-market cross section (log returns); GW2	0.9673	2	-0.0060	-0.0005	0.0282
dtoy	Nearness to Dow 52-week high; GW2	0.8844	2	-0.0438	0.0001	0.0360
dtoat	Nearness to Dow all-time high; GW2	0.9350	2	-0.0539	0.0004	0.0339
ygap	Stock-bond yield gap; GW2	0.9816	2	0.4024	-0.0022	0.0691
rdsp	Stock return dispersion; GW2	0.6979	1	0.6979	0.0299	0.0136

Continued on the next page

Table S.16 Variable list (continued)

	Description; Data source	ρ	tcode	ρ_{tcode}	$\bar{X}$	S_X
tchi	Neely et al. (2014) technical indicators; GW2	-	-	-	0.2594	1.3405
avgcor	Average correlation of daily stock returns; GW2	0.8842	1	0.8842	0.2911	0.1172
shtint	Short stock interest; GW2	0.9834	2	-0.1033	-0.0057	0.2755
lzrt	9 illiquidity measures; GW2	0.8323	1	0.8323	-1.7378	0.2184
bm	Book-to-market ratio; GW	0.9875	2	0.0765	-0.0018	0.0219
ntis	Net equity expansion; GW	0.9803	2	0.1418	-0.0000	0.0040
tbl	Treasury bill rate; GW	0.9875	2	0.3558	-0.0002	0.0046
lty	Long-term government bond yield; GW	0.9951	2	0.0824	-0.0002	0.0033
ltr	Long-term government bond return; GW	0.0441	1	0.0441	0.0076	0.0323
tms	Term spread (lty - tbl); GW2	0.9400	2	0.1038	0.0000	0.0047
AAA	Corporate bond yield; AAA-rated firms; GW	0.9970	2	0.3115	-0.0002	0.0025
BAA	Corporate bond yield; BAA-rated firms; GW	0.9977	2	0.3750	-0.0002	0.0024
corpr	Long-term corporate bond returns; GW	0.1132	1	0.1132	0.0078	0.0283
dfy	Default yield spread (BAA - AAA); GW	0.9636	2	0.3157	-0.0000	0.0012
dfr	Default return spread (corpr - ltr); GW	-0.0297	1	-0.0297	0.0002	0.0154
infl	Inflation; GW	0.5618	1	0.5618	0.0025	0.0035
ogap	Industrial production output gap; GW2	0.9956	2	0.2710	0.0000	0.0066
wtexas	Oil price changes; GW2	0.1514	1	0.1514	0.0051	0.0907
sntmlr	Optimized investor sentiment index (log returns); GW2	0.9241	2	-0.1858	0.0002	0.1349
ndrbl	New orders/shipments to durable goods; GW2	0.6111	1	0.6111	-0.0099	0.0393
Volume	S&P 500 Volume; YF	0.9755	5	-0.4891	0.0096	0.1532
50-dayUpCr	# of 50-day line upward crossings (mon. aggregate); YF	-	-	-	0.7354	0.9122
50-dayUpVi	# of days above 50-day line (mon. aggregate); YF	-	-	-	13.7833	7.6705

Continued on the next page

Table S.16 Variable list (continued)

	Description; Data source	ρ	tcode	ρ_{tcode}	$\bar{X}$	S_X
50-dayDoCr	# of 50-day line downward crossings (mon. aggregate); YF	-	-	-	0.7354	0.9568
50-dayDoVi	# of days below 50-day line (mon. aggregate); YF	-	-	-	7.2313	7.6935
200-dayUpCr	# of 200-day line upward crossings (mon. aggregate); YF	-	-	-	0.2708	0.6720
200-dayUpVi	# of days above 200-day line (mon. aggregate); YF	-	-	-	15.6458	8.4330
200-dayDoCr	# of 200-day line downward crossings (mon. aggregate); YF	-	-	-	0.2708	0.6751
200-dayDoVi	# of days below 200-day line (mon. aggregate); YF	-	-	-	5.3688	8.3703
BBUpCr2	# of upper BB (k=2) crossings (mon. aggregate); YF	-	-	-	0.5521	0.7370
BBUpVi2	# of days above upper BB (k=2) (mon. aggregate); YF	-	-	-	1.0104	1.5350
BBDoCr2	# of lower BB (k=2) crossings (mon. aggregate); YF	-	-	-	0.5292	0.7659
BBDoVi2	# of days below lower BB (k=2) (mon. aggregate); YF	-	-	-	0.9583	1.4812
RSIUpCr	# of upward RSI $>$ 50 crossings (mon. aggregate); YF	-	-	-	1.1625	1.0948
RSIUpVi	# of days RSI $>$ 50 (mon. aggregate); YF	-	-	-	13.5292	6.7143
RSI70Cr	# of RSI crossings to RSI $\geq$ 70 (mon. aggregate); YF	-	-	-	0.4938	0.8897
RSI70Vi	# of overbought days (RSI $\geq$ 70) (mon. aggregate); YF	-	-	-	1.7083	3.4747
RSIDoCr	# of downward RSI $<$ 50 crossings (mon. aggregate); YF	-	-	-	1.1625	1.0891
RSIDoVi	# of days RSI $<$ 50 (mon. aggregate); YF	-	-	-	7.4854	6.7076
RSI30Cr	# of RSI crossings to RSI $\leq$ 30 (mon. aggregate); YF	-	-	-	0.1875	0.5499
RSI30Vi	# of oversold days (RSI $\leq$ 30) (mon. aggregate); YF	-	-	-	0.4229	1.4239
IPO	# of IPOs excluding SPACs, direct listings, penny stocks, units, closed-end funds, etc.; JR	-	-	-	17.9604	16.1464
MAI(1,9)	MAI(1,9); GW	-	-	-	0.7375	0.4400
MAI(1,12)	MAI(1,12); GW	-	-	-	0.7688	0.4216

Continued on the next page

Table S.16 Variable list (continued)

	Description; Data source	ρ	tcode	ρ_{tcode}	$\bar{X}$	S_X
MAI(2,9)	MAI(2,9); GW	-	-	-	0.7500	0.4330
MAI(2,12)	MAI(2,12); GW	-	-	-	0.7625	0.4256
MAI(3,9)	MAI(3,9); GW	-	-	-	0.7583	0.4281
MAI(3,12)	MAI(3,12); GW	-	-	-	0.7646	0.4243
MOI(9)	MOI(9); GW	-	-	-	0.7688	0.4216
MOI(12)	MOI(12); GW	-	-	-	0.7854	0.4105
OBV(1,9)	OBV(1,9); GW	-	-	-	0.7375	0.4400
OBV(1,12)	OBV(1,12); GW	-	-	-	0.7625	0.4256
OBV(2,9)	OBV(2,9); GW	-	-	-	0.7333	0.4422
OBV(2,12)	OBV(2,12); GW	-	-	-	0.7583	0.4281
OBV(3,9)	OBV(3,9); GW	-	-	-	0.7521	0.4318
OBV(3,12)	OBV(3,12); GW	-	-	-	0.7500	0.4330

S.2.1 *GD* Results; k selected by BN_2

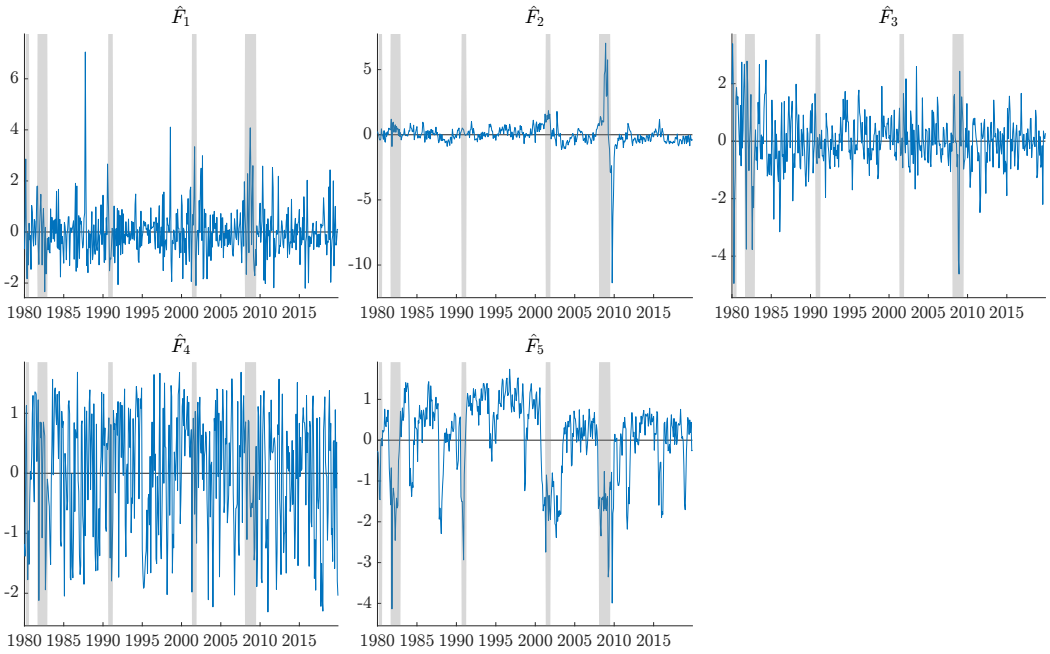

Figure S.16: Extracted factor time series plot; Recessions periods (NBER recession indicator) highlighted in grey. Factors extracted using *GD*, and k selected by *GD* and BN_2.

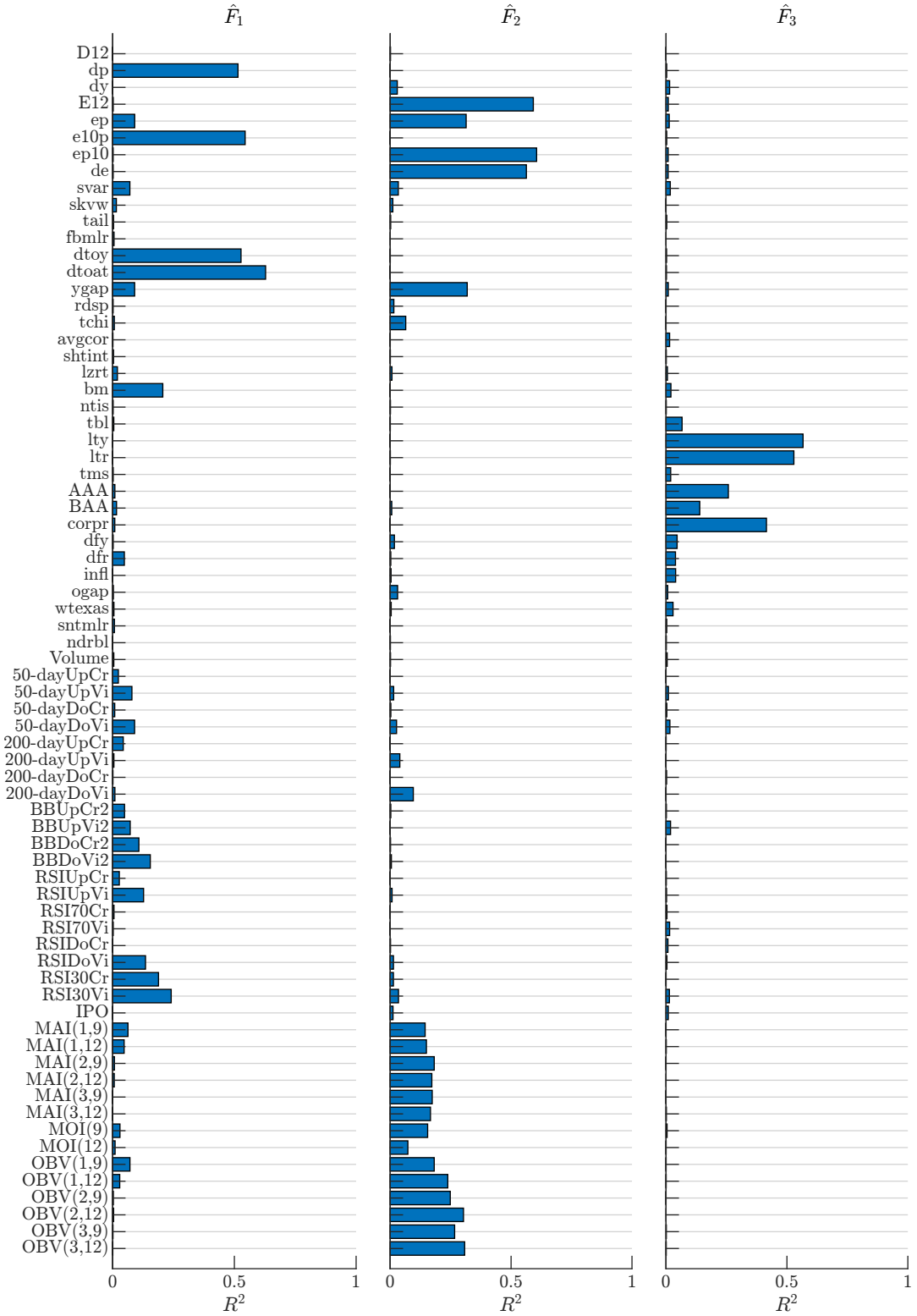

Figure S.17: Univariate Pseudo-R^2 values of regressing X_i on $\hat{F}_i$. Factors extracted using GD, and k selected by GD and BN_2.

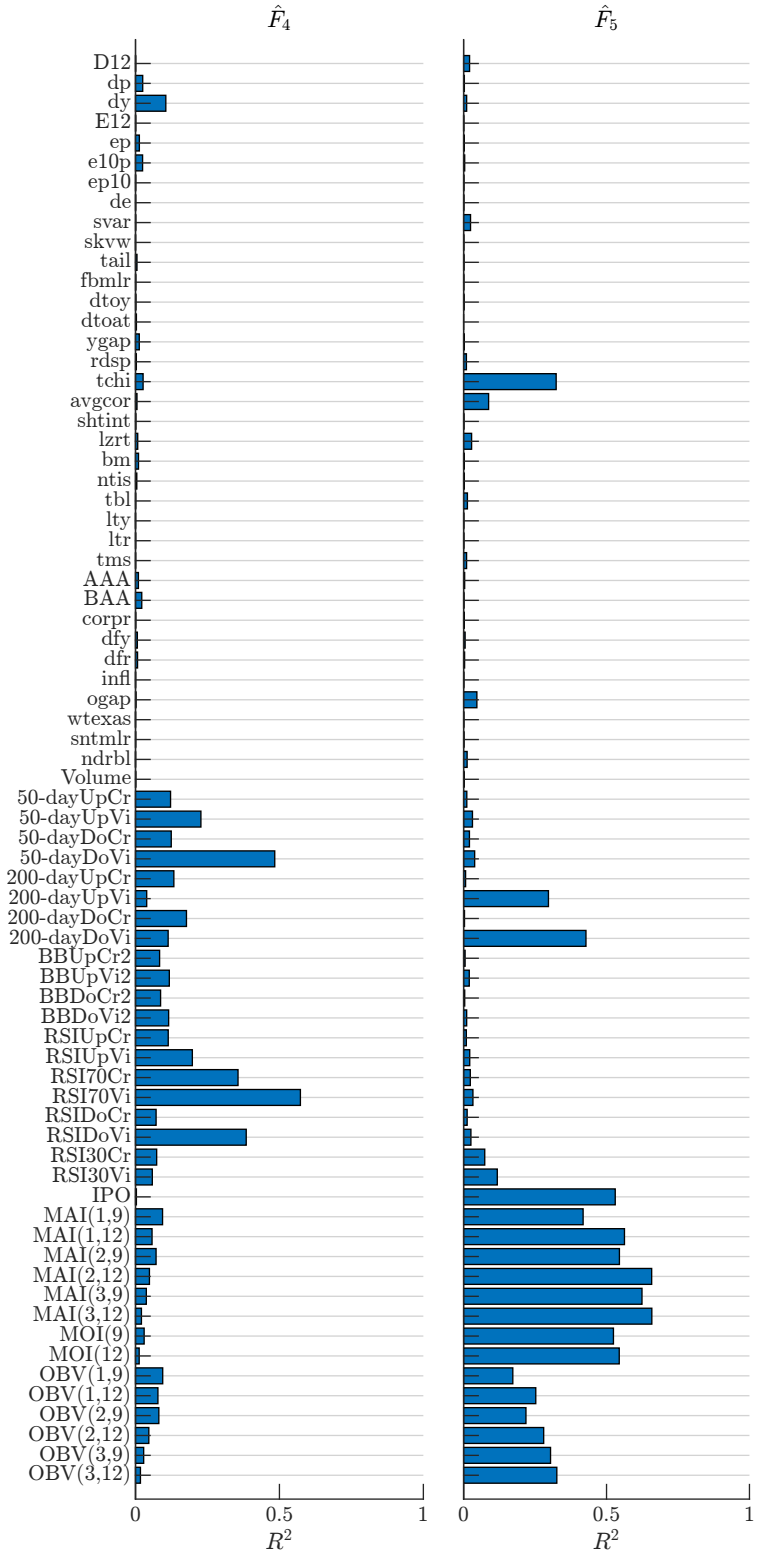

Figure S.18: Univariate Pseudo-R^2 values of regressing X_i on $\hat{F}_i$. Factors extracted using GD, and k selected by GD and BN_2.

S.2.2 *NR* Results; k selected by BN_2

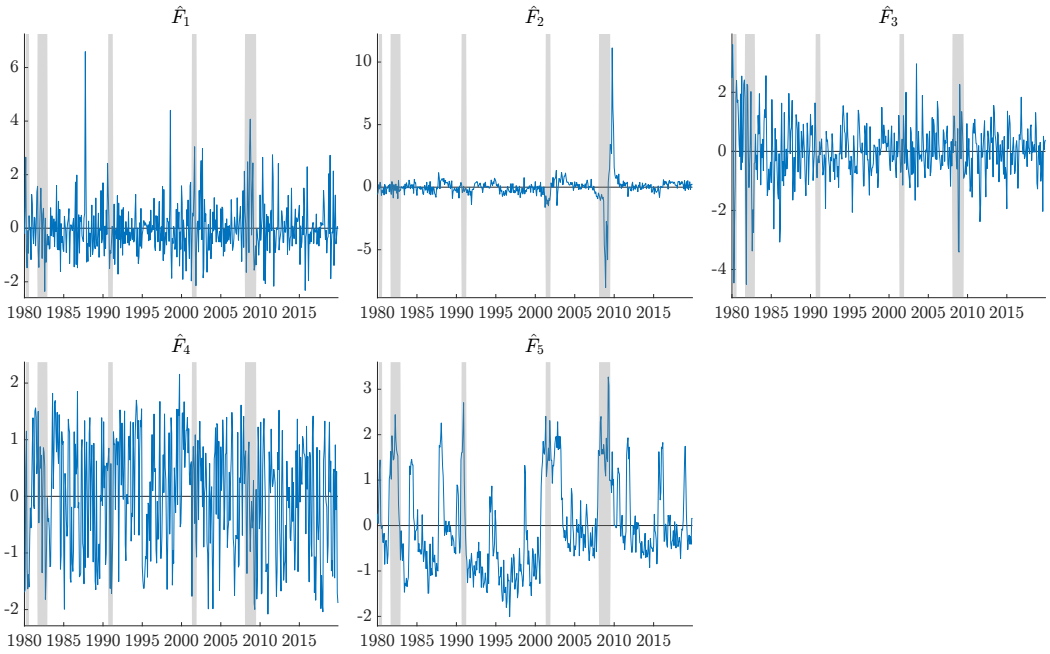

Figure S.19: Extracted factor time series plot; Recessions periods (NBER recession indicator) highlighted in grey. Factors extracted using NR, and k selected by GD and BN_2.

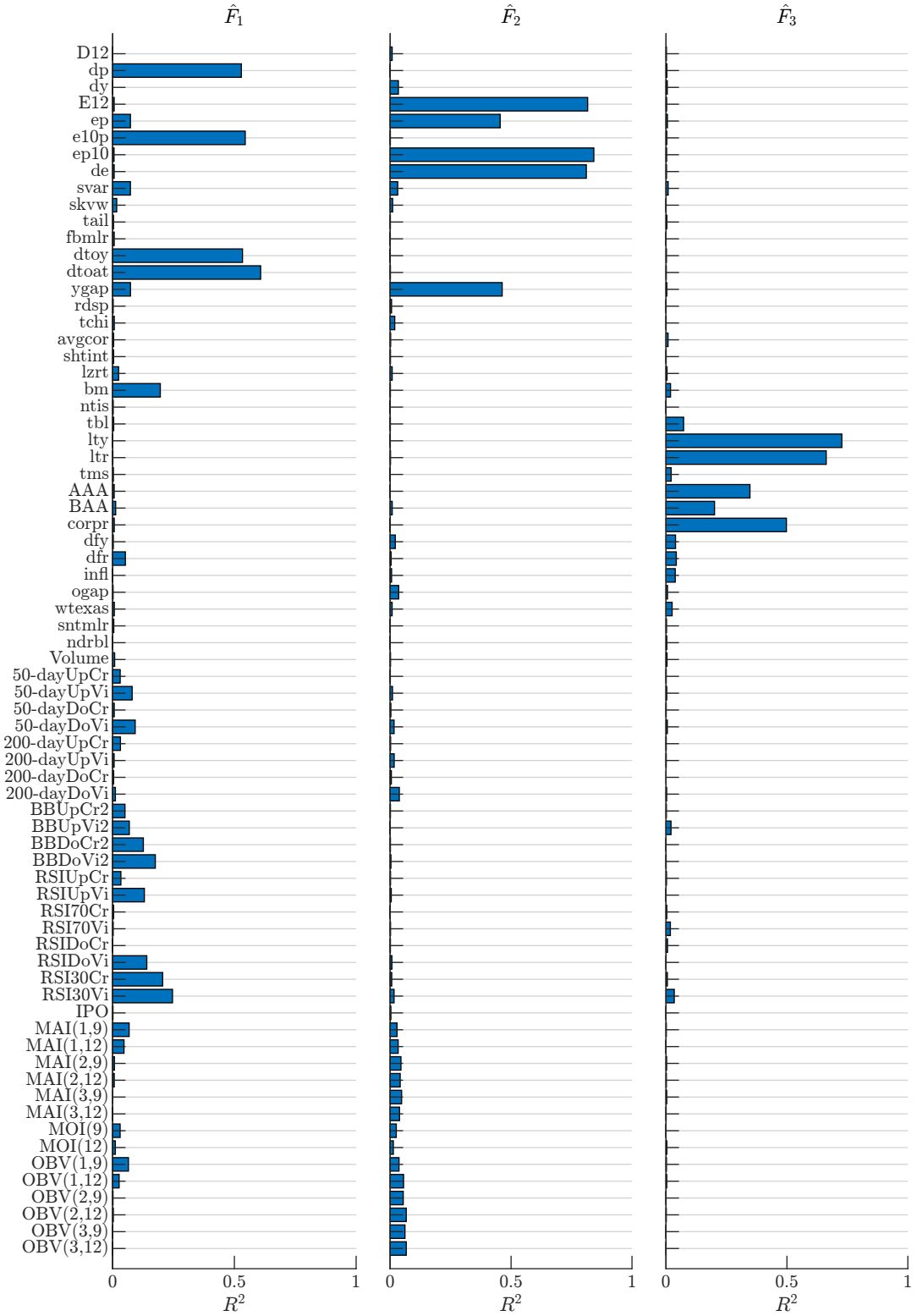

Figure S.20: Univariate Pseudo-R^2 values of regressing X_i on $\hat{F}_i$. Factors extracted using NR, and k selected by GD and BN_2.

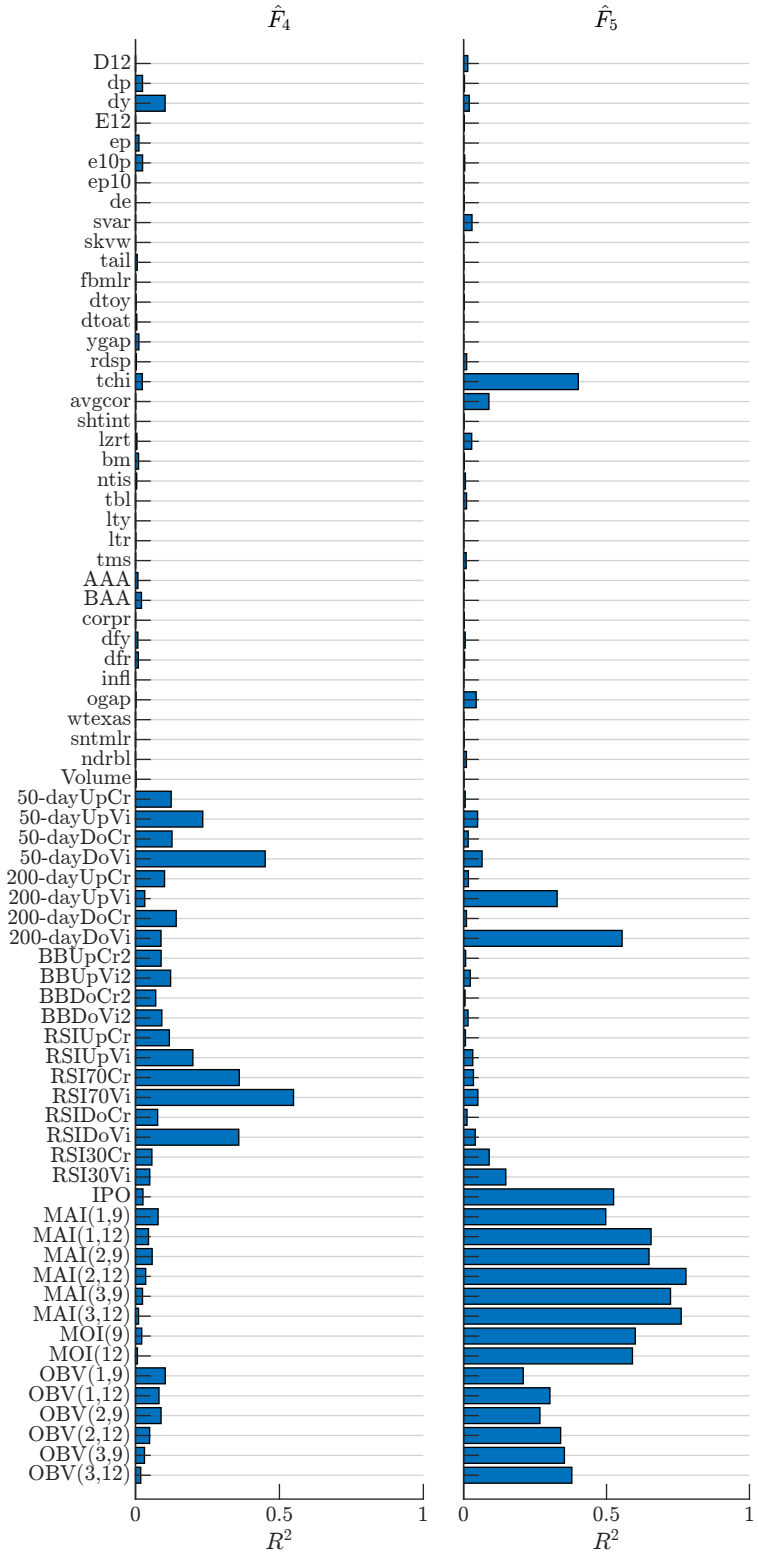

Figure S.21: Univariate Pseudo-R^2 values of regressing X_i on $\hat{F}_i$. Factors extracted using NR, and k selected by GD and BN_2.