Bank of Russia

Stochastic Gradient Variational Bayes and Normalizing Flows for Estimating Macroeconomic Models

WORKING PAPER SERIES

No. 61 / September 2020

Ramis Khabibullin
Sergei Seleznev

Ramis Khabibullin

Bank of Russia. Email: KhabibullinRA@cbr.ru

Sergei Seleznev

Bank of Russia. Email: SeleznevSM@cbr.ru

We are grateful to anonymous referees, Gary Koop, Robert Kohn, Minh Ngoc Tran, Alexey Ponomarenko, Artem Prokhorov, participants of the GRIPS Computation and Econometrics Workshop, participants of the 27th Annual Symposium of the Society for Nonlinear Dynamics and Econometrics, participants of the Bank of Russia Workshop 'Recent Trends in the Central Bank Macroeconomic Modeling', participants of the Econometric Society World Congress 2020 for their helpful comments and suggestions.

Bank of Russia Working Paper Series is anonymously refereed by members of the Bank of Russia Research Advisory Board and external reviewers.

Cover image: Shutterstock.com

Address: 12 Neglinnaya street, Moscow, 107016
Tel.: +7 495 771-91-00, +7 495 621-64-65 (fax)
Website: www.cbr.ru

Abstract

We illustrate the ability of the stochastic gradient variational Bayes algorithm, which is a very popular machine learning tool, to work with macrodata and macromodels. Choosing two approximations (mean-field and normalizing flows), we test properties of algorithms for a set of models and show that these models can be estimated fast despite the presence of estimated hyperparameters. Finally, we discuss the difficulties and possible directions of further research.

JEL-classification: C11, C32, C32, C45, E17.

Keywords: Stochastic gradient variational Bayes, normalizing flows, mean-field approximation, sparse Bayesian learning, BVAR, Bayesian neural network, DFM.

Contents

Introduction

Bayesian modelling is a popular approach for estimating macroeconomic models due to regularisation properties and the ability of taking into account epistemic and aleatoric uncertainties. It is one of the main tools for the inference in vector autoregressions (see Litterman (1980), Doan, Litterman and Sims (1984), Sims (1993), Villani (2009), Banbura, Giannone and Reichlin (2010), Koop and Korobilis (2010), Giannone, Lenza and Primiceri (2015)), dynamic factor models (see Otrok and Whiteman (1998), Kim and Nelson (1998), Aguilar and West (2000), Blake and Mumtaz (2012)), dynamic stochastic general equilibrium models (see Smets and Wouters (2003, 2007), Fernandez-Villaverde and Rubio-Ramirez (2007), Justiniano and Primiceri (2008), Herbst and Schorfheide (2015)), agent based models (Grazzini, Richardi and Tsionas (2017), Gatti and Grazzini (2018), Lux (2018) among others.

Posterior distribution plays a key role in Bayesian inference, but unfortunately in most cases it is not possible to sample directly from or integrating over it. Usually, this problem is solved by using approximations. There are two most popular ways to approximate posterior distributions: Monte Carlo approximation and direct approximation. The first group, for instance, includes Gibbs Sampling (see Casella and George (1992)), Importance Sampling (see Owen (2013)), Metropolis-Hastings (see Chib and Greenberg (1995)), Hamiltonian Monte Carlo (see Neal (2011)), No-U-Turn Sampling (see Hoffman and Gelman (2014)), Sequential Monte Carlo (see Doucet, De Freitas and Gordon (2001)) algorithms and the second group contains MAP estimation, Expectation Propagation algorithm (see Minka (2001)), Variational Bayes estimation (see Wainwright and Jordan (2008)), α-divergence (see Li and Turner (2016)) among others. Monte Carlo algorithms (asymptotically) sample from exact posterior that imply accuracy, but direct methods are faster in many tasks. Monte Carlo (MC) estimation methods are widely used in macroeconomics[1] in opposite to direct approximations (except for MAP estimation). To the best of our knowledge, despite the success in other fields there is only small fraction of papers that uses direct approximations (see Korobilis (2017), Koop and Korobilis (2018), Seleznev (2018)) for macromodelling.

To illustrate usefulness and partially fill this gap, we apply the Variational Bayes algorithm (VB) for inference in three classes of models which are of great interest in macroeconomic society in recent years: Bayesian vector autoregression with sparse priors and t-Student errors (t-Student sparse BVAR), Bayesian neural network (BNN) and dynamic factor model (DFM). We choose these models to show the flexibility of the VB approach. In all exercises we ask the method to estimate posterior simultaneously maximising marginal likelihood with respect to hyperparameters[2], which is a very challenging task for MC methods. BVAR exercise shows the applicability of the method to a very popular class of linear models, but without standard restrictions such as Gaussian noise. BNN exercise demonstrates the ability of method to work with highly non-linear models where efficient

[1] For example, Gibbs Sampling: BVAR (see Karlsson (2012)) and DFM (see Blake and Mumtaz (2012)); Metropolis-Hastings algorithm: DSGE (see Herbst and Schorfheide (2016)); Hamiltonian Monte Carlo: Cointegrated BVAR (see Marowka, Peters, Kantas and Bagnarosa (2017)); No-U-Turn Sampling: SVMA (see Plagborg-Moller (2016)); Sequential Monte Carlo: DSGE (see Herbst and Schorfheide (2015)) and ABM (see Lux (2018)).

[2] We treat variance of prior for each coefficient and degrees of freedom for t-Student distribution as hyperparameters.

MC algorithms, such as Gibbs Sampling, are infeasible. DFM exercise shows the ability of the VB method to estimate state-space models (models with temporal dependencies) that can also be useful for ABM and DSGE models. The direct measures of accuracy cannot be directly applied for models described above due to the absence of closed form marginal likelihood expression, so we additionally run an experiment with classical sparse Bayesian learning regression.

The traditional VB approach restricts families of posterior and approximation densities because of the need to have closed form or simply solved (optimised) steps[3] and is not convenient for direct application in some of described earlier tasks. To overcome these difficulties, we use two popular machine learning tricks: stochastic gradient estimation procedure and normalizing flow density estimation.

The stochastic gradient estimation procedure for VB algorithm or stochastic gradient variational Bayes (SGVB) was introduced in Kingma and Welling (2014) and is popular in a wide range of algorithms including variational autoencoders (see Kingma and Welling (2014)), variational dropout (see Kingma, Salimans and Welling (2015), Gal and Ghahramani (2016), Molchanov, Ashukha and Vetrov (2017)), importance weighted autoencoders (see Burda, Grosse and Salakhutdinov (2015)), Hamiltonian variational inference (see Salimans, Kingma and Welling (2015)), Bayesian compression (see Louizos, Ulrich and Welling (2017)), and variational sequential Monte Carlo (see Naesseth, Linderman, Ranganath and Blei (2018)). It helps to optimise evidence lower bound (ELBO) for a wide variety of posterior, but requires the ability of fast and accurate sampling from approximate density. This requirement usually restricts the family of approximate densities.

Normalizing flows (NF) is a transformation of random variables that forms a rich family of densities. In machine learning, it is often used as an alternative for generative adversarial networks (see Goodfellow, Pouget-Abadie, Mirza, Xu, Warge-Farley, Ozair, Courville and Bengio (2014)) or variational autoencoders (see Kingma and Welling (2014)) for generation of objects similar to data (see Dinh, Krueger and Bengio (2014), Dinh, Sohl-Dickstein and Bengio (2016), Kingma and Dhariwal (2018)). Rezende and Mohamed (2015) propose use of NF transformation as a family of approximations for variational inference and mitigation of the problem of restriction for VB approximations. It increases computational complexity, but in practice still works fast relative to MC methods.

We describe SGVB and NF in more detail in Section 2 and Section 3. Section 4 is devoted to details of the models. Results are presented in Section 5. In Section 6 we discuss results and further directions of work. Section 7 concludes.

Stochastic Gradient Variational Bayes

VB algorithm maximises lower bound of logarithm of marginal likelihood with respect to approximate density and hyperparameters:

$$\log p(y|x,\varphi) = \log \int p(y,\theta|x,\varphi)d\theta = \log \int \frac{p(y|\theta,x,\varphi)p(\theta|\varphi)}{q(\theta)} q(\theta)d\theta \quad (1)$$

$$\geq \int (\log p(y,\theta|x,\varphi) - \log q(\theta))q(\theta)d\theta = \quad (2)$$

[3] Probably it is the reason of small popularity of this method among macroeconomists.

$$\log p(y|x,\varphi) - \int (\log q(\theta) - \log p(\theta|y,x,\varphi)) q(\theta) d\theta = \quad (3)$$
$$\log p(y|x,\varphi) - KL(q(\theta)||\log p(\theta|y,x,\varphi)) \; = L(q,\varphi) \quad (4)$$

where y, x, φ, θ are sets of dependent variables, regressors, hyperparameters and parameters; $q(\theta)$ is approximate density; $p(y|x,\varphi)$ and $L(q,\varphi)$ are logarithms of marginal likelihood and its lower bound; $KL\big(q(\theta)||p(\theta|y,x,\varphi)\big)$ is a KL divergence.

Traditional VB uses an iterative procedure that assumes $q(\theta)$ to be a product of block densities (see Ormerod and Wand (2010), Blei, Kucukelbir and McAuliffe (2017)) and simple form for $p(\theta|y,x,\varphi)$ implying closed form for each step. SGVB allows for optimising $L(q,\varphi)$ directly with stochastic optimization. It utilises the fact that $\int (\log p(y,\theta|x,\varphi) - \log q(\theta)) q(\theta) d\theta$ and its derivatives can be estimated via samples. For a parametric family $q_\psi(\theta)$ we get:

$$\begin{gathered}
\nabla_\psi \Big(\int \big(\log p(y,\theta|x,\varphi) - \log q_\psi(\theta) \big) q_\psi(\theta) d\theta \Big) \\
= \int \nabla_\psi \log q_\psi(\theta) \big(\log p(y,\theta|x,\varphi) - \log q_\psi(\theta) \big) q_\psi(\theta) d\theta \\
+ \int \nabla_\psi \big(\log p(y,\theta|x,\varphi) - \log q_\psi(\theta) \big) q_\psi(\theta) d\theta \\
\approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi \log q_\psi(\theta_i) \big(\log p(y,\theta_i|x,\varphi) - \log q_\psi(\theta_i) \big) \\
+ \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi \big(\log p(y,\theta_i|x,\varphi) - \log q_\psi(\theta_i) \big) \\
\theta_i \sim q_\psi(\theta), i = 1, \dots, N
\end{gathered}$$

This estimator has a large variance in practice, so Kingma and Welling (2014) proposed a reparameterization trick[4]. Distribution $q_\psi(\theta)$ is replaced with $q\big(g_\psi(e)\big)$, where $e \sim p(e)$. $p(e)$ does not depend on parameters and the new estimator can be written in the following form:

$$\begin{gathered}
\nabla_\psi \Big(\int \big(\log p(y,\theta|x,\varphi) - \log q_\psi(\theta) \big) q_\psi(\theta) d\theta \Big) \\
= \nabla_\psi \Big(\int \Big(\log p\big(y, g_\psi(e)|x,\varphi\big) - \log q\Big(g_\psi(e)\Big) \Big) p(e) de \Big) \\
= \Big(\int \nabla_\psi \big(\log p\big(y, g_\psi(e)|x,\varphi\big) - \log q\big(g_\psi(e)\big) \big) p(e) de \Big) \quad (5) \\
\approx \frac{1}{N} \sum_{i=1}^{N} \nabla_\psi \big(\log p\big(y, g_\psi(e_i)|x,\varphi\big) - \log q\big(g_\psi(e_i)\big) \big) \quad (6)
\end{gathered}$$

$$e_i \sim p(e), i = 1, \dots, N$$

This estimator is unbiased, so under the appropriate schedule, learning rates can be used in stochastic gradient algorithm or its extensions (see Kushner and Yin (2003)).

Mean-field and Normalizing Flows Approximation

In this paper, we approximate the posterior with Gaussian mean-field and NF approximation. The former has the form:

[4] It cannot be applied in all cases.

$$q_\psi(\theta) = q_{\psi_1}(\theta^1) q_{\psi_2}(\theta^2) \dots q_{\psi_D}(\theta^D)$$
$$q_{\psi_d}(\theta^d) \sim \mu_d + \sigma_d N(0,1),\ d = 1, \dots, D$$

where D is the dimensionality of the space of parameters, and ψ_d are parameters of dth component of approximate distribution.

The latter uses a chain of invertible transformations:

$$\theta = g_\psi(e) = f_{\psi_K}^K \left(f_{\psi_{K-1}}^{K-1} \left(\dots \left(f_{\psi_1}^1 (e) \right) \right) \right)$$

and the identities:

$$q(\theta) = \left| \det \frac{d\theta}{de} \right|^{-1} q(e)$$
$$= \left| \det \frac{dg_\psi(e)}{de} \right|^{-1} q(e)$$
$$= q(e) \left| \det \frac{df_{\psi_1}^1}{de} \right|^{-1} \prod_{k=2}^K \left| \det \frac{df_{\psi_k}^k}{df_{\psi_{k-1}}^k} \right|^{-1}$$

where K is the number of transformations applied to initial random variables e, and ψ_k are parameters of kth transformation.

The second term in (6) in this case is equal to:

$$\frac{1}{N} \sum_{i=1}^N \nabla_\psi \left(-\log q \left(g_\psi(e_i) \right) \right) = \frac{1}{N} \sum_{i=1}^N \nabla_\psi \left(-\log q(e_i) + \log \left| \det \frac{df_{\psi_1}^1}{de} \right| + \sum_{k=2}^K \log \left| \det \frac{df_{\psi_k}^k}{df_{\psi_{k-1}}^k} \right| \right)$$

The main difficulty of this approach is choosing the functional form of transformation to be flexible enough and computationally efficient. There is a number of approaches in the literature to construct such transformation: non-linear independent component estimation structure (see Dinh, Krueger and Bengio (2014)), planar and radial flows (see Rezende and Mohamed (2015)), real-value non volume preserving transformation (see Dinh, Sohl-Dickstein and Bengio (2016)), inverse autoregressive flows (see Kingma, Salimans, Jozefowicz, Chen, Sutskever and Welling (2016)), masked autoregressive flows (see Papamakarios, Pavlakou and Murray (2017)), Sylvester NF (see van den Berg, Hasenclever, Tomczac and Welling (2018)) and neural autoregressive flows (see Huang, Krueger, Lacoste and Courville (2018)).

Here we use Sylvester NF (SNF) which has the following form:

$$f_{\psi_k}^k(z) = z + Ah(Bz + b) \quad (7)$$

where A, B and b are $D \times M$, $M \times D$ and $M \times 1$ matrices, $h(\cdot)$ is an activation function and $M \leq D$. Berg, Hasenclever, Tomczac and Welling (2018) showed that:

$$\det \frac{df_{\psi_k}^k(z)}{dz} = \det(I_M + diag\left(h'(Bz + b)\right)BA) \quad (8)$$

To ensure inevitability, the authors apply the reparametrisation of (7):

$$f_{\psi_k}^k(z) = z + QRh\left(\tilde{R}Q^T z + b\right) \quad (9)$$

and set:

$$R_{mm}\tilde{R}_{mm} > -1/\|h'\|_\infty,\ m = 1, \dots, M$$

where R and $\tilde{R}$ are upper triangular $M \times M$ matrices, Q is $D \times M$ matrix with columns forming orthonormal set of vectors. In this case (8) has the form:

$$\det \frac{df_{\psi_k}^k(z)}{dz} = \det(I_M + diag\left(h'\left(\tilde{R}Q^T z + b\right)\right)\tilde{R}R) \quad (10)$$

We chose Q to be a permutation matrix.

Models

Sparse Bayesian learning regression

Sparse Bayesian learning (SBL) regression problem can be written as (see Tipping (2001)):

$$y_i = A + Bx_i + e_i \quad (11)$$
$$e_i \sim N(0, \sigma), \quad i = 1, \dots, N \quad (12)$$
$$A \sim N(0, \sigma_A), \; B_d \sim N(0, a_d \sigma_B), \; d = 1, \dots, D \quad (13)$$

where y_i is a dependent variable, x_i is $D \times 1$ vector of covariates, and e_t is an error, A is an intercept, B is $1 \times D$ matrix of coefficients, σ is estimated error covariance, a is $D \times 1$ vector estimated hyperparameters, σ_A and σ_B are non-estimated hyperparameters.

Bayesian vector autoregression with sparse priors and t-Student errors

We estimate Bayesian VAR with t-Student errors and prior in the spirit of Sparse Bayesian Learning (see Tipping (2001)). It was shown that sparse Bayesian learning (SBL) prior prunes predictors in linear regression under Gaussian errors (see Faul and Tipping (2002), Wipf and Nagarajan (2007)), but we found empirically that it works well with t-Student errors and in a non-linear case[5].

t-Student sparse BVAR has the following form[6]:

$$y_t = A + B_1 y_{t-1} + \cdots + B_p y_{t-p} + Ce_t \quad (14)$$
$$e_{it} \sim St(d_i, 0, \sigma_i), \quad i = 1, \dots, N, t = 1, \dots, T \quad (15)$$
$$A_i \sim N(0, \sigma_A), \; B_{l,ij} \sim N\left(0, a_{l,ij}\sigma_B\right), \; \log \sigma_i \sim N(\mu_\sigma, \sigma_\sigma) \quad i, j = 1, \dots, N, \; l = 1, \dots, p \quad (16)$$

where y_t is $N \times 1$ vector of endogenous variables, e_t is $N \times 1$ vector of shocks, A is $N \times 1$ vector of intercepts, $B_1, \dots, B_p$ are $N \times N$ matrices of coefficients, σ is $N \times 1$ vector of scale parameters for t-distribution, a, d and C are $N \times 1$, $N \times 1$, $N \times pN$ matrices of estimated hyperparameters, and σ_A, σ_B, μ_σ and σ_σ are non-estimated hyperparameters. Depending on the assumptions on matrix C, there are two types of models: diagonal (C is set to be identity matrix) and non-diagonal (C is lower triangular matrix with ones on its main diagonal and estimated hyperparameters in positions below the main diagonal). Results with estimated matrices are shown in Appendix B.

[5] See BNN.
[6] In this section we overload our notations.

Bayesian neural network

In recent years, neural networks were with a great success applied for a wide variety of tasks (see Goodfellow, Bengio and Courville (2016)), but usually they require large datasets. BNN [7] is an alternative that can alleviate this problem but requires large computations using MC methods for estimation.

Here we use neural network with 2 hidden layers:

$$h_t^1 = h(W_1 x_t + b_1) \tag{17}$$
$$h_t^2 = h(W_2 h_t^1 + b_2) \tag{18}$$
$$y_t = W_3 h_t^2 + b_3 + C e_t \tag{19}$$

where y_t is $N \times 1$ vector of endogenous variables, x_t is $pN \times 1$ vector of concatenated lags, e_t is $N \times 1$ vector of shocks (see eq.(12)), W_1, W_2 and W_3 are $N_1 \times pN$, $N_2 \times N_1$ and $N \times N_2$ matrices of coefficients with SBL prior, b_1, b_2 and b_3 are $N_1 \times 1$, $N_2 \times 1$ and $N \times 1$ vectors of biases with SBL prior, $h(\cdot)$ is an activation function.

Dynamic Factor Model (DFM)

DFM model is widely applied for different exercises (see Stock and Watson (2016)) due to its ability to take into account information of many time series. DFM in this paper has the form:

$$F_t = A + B_1 F_{t-1} + \cdots + B_p F_{t-p} + e_t \tag{20}$$
$$y_t = C + D F_t + e_t^{obs} \tag{21}$$

$$e_{kt} \sim N(0, 1), \quad k = 1, \dots, K_{max}, t = 1, \dots, T \tag{22}$$
$$e_{it}^{obs} \sim N\left(0, \sigma_i^{obs}\right), \quad i = 1, \dots, N, t = 1, \dots, T \tag{23}$$

$$A_k \sim N(0, \sigma_A), \qquad B_{l,jk} \sim N(0, a_k \sigma_B), \qquad C_i \sim N(0, \sigma_C), \quad D_{ik} \sim N(0, a_k \sigma_D),$$
$$\log \sigma_i \sim N\left(\mu_{\sigma,obs}, \sigma_{\sigma,obs}\right), j, k = 1, \dots, K_{max}, t = 1, \dots, T, \quad i = 1, \dots, N \tag{24}$$

where y_t is $N \times 1$ vector of endogenous variables, F_t is $K_{max} \times 1$ vector of factors, e_t is $K_{max} \times 1$ vector of shocks, e_t^{obs} is $N \times 1$ vector of observable errors, A, B_1,..., B_p, C and D are matrices of $K_{max} \times 1$, $K_{max} \times K_{max}$, ..., $K_{max} \times K_{max}$, $N \times 1$ and $N \times K_{max}$ coefficients, σ^{obs} is $N \times 1$ vector of scale parameters, a is $K_{max} \times 1$ vector of estimated hyperparameters, σ_A, σ_B, σ_C, σ_D,, $\mu_{\sigma,obs}$ and $\sigma_{\sigma,obs}$ are non-estimated hyperparameters. Note, that K_{max} is assumed to be upper bound for the number of factors and vector of hyperparameters a chooses relevant factors.

[7] See Krueger, Huang, Islam, Turner, Lacoste and Courville (2018) for ML applications.

Experiments

All experiments were run in Tensorflow[8] (see Abadi et al. (2016)) on a Desktop PC with the following specifications: Intel(R) Core(TM) i5-4210U CPU @ 1.70 GHz 2.40 GHz and RAM 4 GB. For training models we use Adam optimiser (see Kingma and Ba (2014)) with the learning rate 0.001[9] with slight modifications which will be described separately for each model. In experiments with NF approximation we set $M = 50$, $K = 20$ (except for SBL regression with 10 covariates), tanh nonlinearity and use Xavier style initialisation (with slight modification for DFM). Firstly, for each model we run an experiment[10] with artificial data and then with real data (except for SBL regression). We also standardise data before real data experiments.

Sparse Bayesian learning regression

For SBL regression we can directly compare the performance of mean-field and NF approximations with respect to exact marginal likelihood optimisation. For this comparison, we randomly generated covariates from random normal distribution and multiply then by random matrix. Coefficients were generated from normal distribution and then multiplied by vector of discrete 0/1 random variables with a different degree of sparsity for experiments. All models were estimated with 50,000 iterations of Adam.

Six experiments were run for 10/50 covariates, 0.2/0.5/0.8 sparsity[11] and 100 observations. In all experiments except for one mean-field and NF approximations choose similar structure to the direct marginal likelihood optimisation (see Figures 1–2). Also note that ELBO and marginal likelihood[12] are close to maximum likelihood (ML) values (see Table 1). Even for the mean-field approximation with 10 covariates and 0.8 sparsity where the structure is different from other models, ELBO and marginal likelihood are close to ML.

Bayesian vector autoregression with sparse priors and t-Student errors

To demonstrate the ability of VB algorithms for optimisation of lower bound of marginal likelihood with respect to hyperparameters and choosing right sparse structure for Bayesian vector autoregression with sparse priors and t-Student errors, we generate 3 time series (see Figure 3) with a diagonal covariance matrix, 15, 20 and 30 degrees of freedom, 5 lags and sparse structure. Models with 30, 100 and 1000 points are estimated using 50,000 iterations of the Adam

[8] Note that for SGVB one may use flexible frameworks for Bayesian estimation such as Stan (see Stan Development Team (2016)), Edward (see Tran, Kucukelbir, Dieng, Rudolph, Liang and Blei (2017), Tran, Hoffman, Saurous, Brevdo, Murphy and Blei (2017)) and PyMC3 (Salvatier, Wiecki and Fonnesbeck (2016)) to avoid tedious code writing.

[9] Despite the fact that conditions for convergence don't hold for this learning rate it often used in ML and usually works well in practice. We discuss the choosing of optimiser in Section 6.

[10] Each experiment was run at least 3 times. In tables and graphs we show the best result.

[11] In this subsection degree of sparsity denotes expected number of nonzero coefficients.

[12] Marginal likelihood is estimated via 100,000 importance sampling draws.

	ELBO		Marginal likelihood		
	MF	NF	MF	NF	ML
Artificial data, 10 covariates, 0.2 sparsity	-158.4	-159.9	-158.2	-158.8	-158.1
Artificial data, 10 covariates, 0.5 sparsity	-164.3	-164.9	-163.3	-163.3	-162.9
Artificial data, 10 covariates, 0.8 sparsity	-177.1	-175.7	-175.6	-174.7	-174.6
Artificial data, 50 covariates, 0.2 sparsity	-187.9	-190.9	-184.4	-185.9	-183.9
Artificial data, 50 covariates, 0.5 sparsity	-246.4	-244.7	-241	-240.2	-239
Artificial data, 50 covariates, 0.8 sparsity	-259.8	-254.1	-252.5	-248.7	-247.8

Table 1. ELBO and marginal likelihood for SBL regression

algorithm. Results for mean-field, NF and OLS estimates[13,14] of coefficients are shown in Figures 4–6. VB approximations produce sparse solutions for all dataset sizes. For 30 points, the OLS estimate is not sparse and the NF approximation has lower sparsity than the mean-field approximation, but of course this relation between NF and mean field algorithms is data dependent. However, as expected this sparsity does not imply better estimates. It can be seen from Table 2 that ELBO and marginal likelihood are larger for NF approximation. It is also fulfilled for 100 and 1000 points. Note that for 100 and 1000 points, VB algorithms choose predictors with 1 and 0 errors respectively, while OLS implies near-zero coefficients only for 1000 points.

For the experiment on real data, we choose a dataset with 7 variables from Giannone, Lenza and Primiceri (2015). Unlike in Giannone, Lenza and Primiceri (2015) log of real GDP, GDP deflator, real consumption, real investment, hours worked and real compensation per hours were differentiated; federal fund rate was used without any changes. Similarly to artificial data we estimated model with 5 lags, so finally dataset consists of 194 points from 1960Q3 to 2008Q4. As an alternative to VB algorithms, we applied the Gibbs Sampling algorithm estimated via NF hyperparameters. We also show OLS coefficients to illustrate absence of sparsity. Estimates are visualised in Figure 7. In general, results are consistent with findings for artificial data. As in the case of artificial data, the NF algorithm has a larger ELBO and marginal likelihood (see Table 2), but the difference between marginal likelihood and ELBO is approximately equal. It means that NF distribution underfits true posterior. Visually, mean estimates for Gibbs Sampling and NF algorithms are similar (see Figure 7). Correlations for a number of individual pairs of coefficients have less similarity, but remain close in average (see Figure 8). The maximum absolute (mean) difference between means is equal to 0.03 (0.002), while for correlations is 0.45 (0.02). We discuss potential sources and consequences of the underfitting in the next Section.

Bayesian neural network

Using the same 3 time series of artificial data and US Data, we ask VB algorithms to estimate BNN with 30 (10) neurons for first (second) layer and LeakyReLu nonlinearity to show the ability of algorithms to work with nonlinear models. The previously used Adam algorithm for some experiments converges to poor local optimums with near zero

[13] For Bayesian estimates we show mean results.
[14] All coefficients are estimated using 100,000 draws.

coefficients, so we modified it. 10,000 iterations were run as previously. After that d and σ were fixed and replaced for the next 5000 iterations with large (50) and small (0.01) values respectively. The subsequent iterations were run via Adam algorithm. The second part of the algorithm helps us to "overfit" data, so optimiser is guided to have non-zero coefficients. The total number of iterations for artificial and real data is 50,000 and 100,000 respectively[15].

	ELBO		Marginal likelihood	
	MF	NF	MF	NF
Artificial data, 30 points	-172.4	-159.7	-169.5	-158.1
Artificial data, 100 points	-522.9	-515.5	-516.9	-514.6
Artificial data, 1000 points	-4692.7	-4687.4	-4682.1	-4681.6
US Data	-1374.5	-1362.9	-1364.1	-1352.7

Table 2. ELBO and marginal likelihood, Bayesian vector autoregression with sparse priors and t-Student errors

	ELBO		Marginal likelihood	
	MF	NF	MF	NF
Artificial data, 30 points	-165.9	-168.2	-157.9	-152.4
Artificial data, 100 points	-525.7	-527.15	-512.4	-506.5
Artificial data, 1000 points	-4697.5	-4702.6	-4681.9	-4679.2
US Data	-1240.2	-1236.9	-1203.1	-1192.1

Table 3. ELBO and marginal likelihood, Bayesian neural network

Figures 9–17 show estimates for W_1, W_2 and W_3 for models with 30, 100 and 1000 points. Table 3 shows ELBO and marginal likelihoods. For all models, BNN achieves close ELBO results to BVAR with sparse priors and t-Student errors, which has well estimates and contains true data generating process. Moreover, NF approximation achieves a better marginal likelihood than BVAR for all dataset sizes. We also found that ELBO for mean-field approximation for 30 and 100 points is larger than for NF approximation. It is a consequence of the optimisation procedure, but it is not the case for a lower learning rates NF (see next Section). An interesting fact is that all models have small fraction of non-zero elements and the structure of neurons are similar to BVAR structure. For example, the first variable in NF approximation for 100 points depends on neuron8. This neuron depends only on neuron3, which is a transformation of first lag of the first variable.

For the US Data, BNN significantly improves in-sample fit of BVAR with sparse priors and t-Student errors (see Tables 2–3). Additionally, note that both approximations choose in the first layer the larger number of neurons than the number of variables (see Figures 18–20). These results may be signals for importance of non-linearity for forecasting, but we did not test this and leave investigation of forecasting/overfitting properties of sparse model for macrodata for further research.

[15] We also tried to apply different types of annealing, but found that this algorithm works better. The combination of algorithms shows comparative results.

	Artificial data	US Data
MF	3	20
NF	3	20
IC1	6	9
IC2	3	7
IC3	10	20
PC1	9	18
PC2	7	17
PC3	10	20
AIC1	10	20
AIC2	10	20
AIC3	10	20
BIC1	10	20
BIC2	10	20
BIC3	3	7

Table 4. Number of estimated factors

		MF	NF	PCA
True	Factor 1	0.997	0.996	0.995
	Factor 2	0.992	0.997	0.992
	Factor 3	0.997	0.993	0.984
PCA	Factor 1	0.998	0.998	1
	Factor 2	0.992	0.997	1
	Factor 3	0.997	0.993	1

Table 5. R-squared for regressions of factors estimates on true and 3 PCA factors

Dynamic factor model

Similarly to previous two subsections we firstly generated artificial data. Artificial dataset consists of 50 time series with 100 points. These time series are driven by 3 factors (see Figure 21). Shocks for factors were generated from the standard normal distribution; observation errors have standard deviation 0.3. We set $K_{max} = 10$, so the total number of latent variables is more than 1500 which is compatible with BNN, but DFM has temporal dependence which might be potential source of difficulty. Adam algorithm with 50,000 iterations was used for both approximations.

It was found that mean-field and NF approximations choose correct number of factors in all experiments even when we estimate model with more than 1 lag (we run 5 experiments for 1–3 lags). Note that not all criteria from Bai and Ng (2002) choose correct number of factors (see Table 4) on these data. Because of the absence of factor normalization, estimated factors cannot be compared directly, and we regress mean of factors on true factors and 3 PCA components. Table 5 demonstrates that estimates are similar to the true factors. To illustrate the ability to recover data the product of factors and loadings was sampled. These data approximations plus noise from (23) are shown in Figures 22–24. Both approximations lie near true data.

	ELBO		Marginal likelihood	
	MF	NF	MF	NF
Artificial data	-3013.1	-3023.2	-2973.9	-2979.1
US Data, 1 lag	-39373	-40406	-39141	-40169
US Data, 2 lags	-40343	-40622	-40050	-40420
US Data, 3 lags	-40889	-41082	-40550	-40668

Table 6. ELBO and marginal likelihood, DFM

	MF	NF
Factor 1	0.22	0.89
Factor 2	0.38	0.5
Factor 3	0.94	0.98
Factor 4	0.65	0.94
Factor 5	0.7	0.14
Factor 6	0.35	0.66
Factor 7	0.91	0.97
Factor 8	0.99	0.68
Factor 9	0.98	0.97
Factor 10	0.68	0.87
Factor 11	0.07	0.72
Factor 12	0.94	0.86
Factor 13	0.35	0.4
Factor 14	0.98	0.96
Factor 15	0.28	0.4
Factor 16	0.49	0.79
Factor 17	0.76	0.64
Factor 18	0.56	0.26
Factor 19	0.98	0.59
Factor 20	0.15	0.14

Table 7. R-squared for regressions of factors estimates on 20 PCA factors, 1 lag

These models with $K_{max} = 20$ were applied for the September release[16] of monthly FRED database (see McCraken and Ng (2016)). We choose the maximal balanced panel from this dataset, so the final dataset consists of 128 series and 314 time periods. Models with 1–3 lags were estimated. In all cases mean-field and NF approximations choose 20 factors, in opposite to other criteria (see Table 4). For the US data as for the artificial data, mean-field outperforms NF approximation in terms of ELBO and marginal likelihood (see Table 6), but it is effect of optimisation procedure and partially discussed in next Section[17]. In opposite to artificial data, estimated factors are less related to PCA factors (see Table 7). Note, however, that for factors with large means of loadings R-squared is near 0.9 (see Figures 25–27). Loadings cannot be directly interpreted as an importance of factors (factors are not scaled, multimodality of distribution may appear[18] or, probably, we do not use enough lags), but it is a signal for that and have to be investigated later. Finally, we compared ability to recover true data of NF approximation and Gibbs Sampling given NF hyperparameters. In fact, it is a

[16] Data set was downloaded at 1 October, 2018.
[17] For instance, we run additional 50,000 iterations for artificial data with 0.0001 learning rate and achieve ELBO: -3002.8 and marginal likelihood: -2971.3.
[18] Visually, we did not found multimodality in our estimates.

comparison of recovering data given the same hyperparameters, so as in the case of BVAR it compares Bayesian parts of model. Figure 28 shows 6 randomly chosen series from dataset. Both algorithms demonstrate approximately the same estimates and capture main tendencies in data dynamics. For the most series, estimates are close to PCA with 20 factors, which is the best Frobenius norm estimate.

Discussion and Further Directions

Experiments show that mean-field and NF approximation might be useful for optimisation of marginal likelihood. As expected, NF approximation outperforms mean-field approximation for all models except for DFM model and number of SBL regressions, but we found some intuitively unusual results. Firstly, for a number of models the ELBO of mean-field approximation is larger than ELBO of NF approximation. The main reason of such behavior of models is optimisation procedure. For a bad initialisation, models may fall into poor local optimum. The non-decreasing learning rate is an alternative source of the problem. We found that both factors play significant role, but the second one is more important in investigated models. The number of additional experiments showed that using a decreasing schedule for the learning rate NF approximation helps to achieve better results; however, it requires much more iterations. Secondly, for a number of experiments the marginal likelihood is closer to ELBO for a mean-field approximation. This problem is similar to the first one and can be mitigated by decreasing schedule for the learning rate. Alternatively, the larger number of samples can be used for decreasing the variance of ELBO gradient. Achieving better results for NF approximations of DFM and SBL regression and decreasing the gap between ELBO and marginal likelihood can be done in the same ways.

We also noted that initialisation plays crucial role for state space models, especially, for coefficients of equation for factors. If eigenvalues of a generated matrix are more than 1, factors will be extremely large generating NaNs in the computation procedure. There are many solutions, but we tried two: clipping factors and initialising model with near zero matrices. Finally, we decided to merge these procedures, because the former ensures the absence of NaNs, but gradients may be large and the latter rarely produces NaNs in some experiments.

The computational time is a cornerstone of Bayesian inference. No experiment with mean-field approximation took us more than 1 hour[19]. NF approximations took us no more than 3 hours[20]. The most time consuming model is DFM. Probably, our realisation is not optimal and can be improved, but we found this time acceptable. One may easily use GPU and TPU (provided, for example, for free by Google Colab) or other programming languages to speed up computations. Interestingly, NF falls into the neighbourhood of final point after few thousands of iterations in opposite to mean-field approximations and drifts slowly after that. This fact can be used for high-dimensional model with few hyperparameters to stop model running earlier and apply IS algorithm.

We run only small fraction of possible experiments and just illustrated the potential of described techniques. We leave for further research forecasting properties of estimated models which is one of the main goals of building macromodels. There are a lot of other

[19] For US data, BVAR - 5 minutes, NN - 20 minutes, DFM – 50 minutes.
[20] For US data, BVAR - 20 minutes, NN - 40 minutes, DFM – 2 hours 40 minutes.

directions for further research including optimisation procedure and the form of NF approximation. The optimisation procedure may be modified by changing the learning rate schedule or increasing/decreasing number of iterations. One may use other stochastic optimisation procedures such as momentum (see Polyak (1964)), Nesterov momentum (see Nesterov (1983)), AdaGrad (see Duchi, Hazan and Singer (2011)), RMSProp (see Hinton (2012)), ADVI optimiser (see Kucukelbir, Tran, Ranganath, Gelman and Blei (2017)), restart optimisers (see Loshchilov and Hutter (2017)) and AddSign/PowerSign (see Bello, Zoph, Vasudevan and Le (2017)). NF approximation also requires choosing a number of hyperparameters such as depth and width. Moreover, as was mentioned in introduction other types of NF approximation exist and might be estimated. Even for the presented model, the properties under different parameters of generated data (different noise to signal ratios, misspecified models and so on) have to be investigated. The formal comparison of accuracy and speed with MCMC methods is also important, but our experience shows that VB methods are usually faster to achieve the adequate accuracy, especially in large scale applications (where the closed or simple Gibbs Sampling form are not available).

Only sparse models were investigated in the paper, but that was not the goal. Of course, many models with intractable marginal likelihood and/or posterior (with and without hyperparameters) can be estimated via presented algorithm. Moreover, estimated approximations can be used not directly, but as proposal densities for other algorithms such as importance sampling.

We should mention that we tried to estimate the ABM model which lies in the class of state space models, but its efficient realisation in Tensorflow requires considerable effort and lies outside of the scope of this paper.

Conclusion

We demonstrated the applicability of SGVB algorithm for three different classes of models. We applied traditional mean field approximation and more flexible NF approximation. The results showed that the SGVB algorithm is fast and relatively accurate, but we have a long way to go for full understanding the properties of approximations for macrodata. We hope that our paper will be a starting point for investigating properties of described algorithms for macromodels.

References

ABADI, M., BARHAM, P., CHEN, J., CHEN, Z., DAVIS, A., DEAN, J., DEVIN, M., GHEMAWAT, S., IRVING, G., ISARD, M., KUDLUR, M., LEVENBERG, J., MONGA, R., MOORE, S., MURRAY, D.G., STEINER, B., TUCKER, P., VASUDEVAN, V., WARDEN, P., WICKE, M., YU, Y. AND X. ZHENG (2016): "Tensorflow: a System for Large-scale Machine Learning", OSDI, 16, 265–283

AGUILAR, O. AND M. WEST (2000): "Bayesian Dynamic Factor Models and Portfolio Allocation", Journal of Business and Economic Statistics, 18(3), 338–357

BAI, J. AND S. NG (2002): "Determining the Number of Factors in Approximate Factor Models", Econometrica, 70(1), 191–221

BANBURA, M., GIANNONE, D. AND L. REICHLIN (2010): "Large Bayesian Vector Autoregressions", Journal of Applied Econometrics, 25(1), 71–92

BELLO, I., ZOPH, B. VASUDEVAN, V. AND Q.V. LE (2017), "Neural Optimizer Search with Reinforcement Learning", International Conference on Machine Learning

BLAKE, A. P. AND H. MUMTAZ, (2012): "Applied Bayesian Econometrics for Central Bankers", Technical Books

BLEI, D.M., KUCUKELBIR, A. AND J.D. MCAULIFFE, (2017): "Variational Inference: A Review for Statisticians", Journal of the American Statistical Association, 112(518), 859–877

CASELLA, G. AND E. I. GEORGE (1992): "Explaining the Gibbs sampler", The American Statistician, 46(3), 167–174

CHIB, S. AND E. GREENBERG (1995): "Understanding the Metropolis-Hastings Algorithm", The American Statistician, 49(4), 327–335

DINH, L., KRUEGER, D. AND Y. BENGIO (2014): "NICE: Non-linear Independent Components Estimation", arXiv preprint arXiv:1410.8516

DINH, L., SOHL-DICKSTEIN, J. AND S. BENGIO (2016): "Density Estimation Using Real NVP", arXiv preprint arXiv:1605.08803

DOAN, T., LITTERMAN, R. AND C. SIMS (1984): "Forecasting and Conditional Projection Using Realistic Prior Distributions", Econometric Reviews, 3(1), 1–100

DOUCET, A., DE FREITAS, N. AND N. GORDON (2001): "An Introduction to Sequential Monte Carlo Methods", In Sequential Monte Carlo Methods in Practice, Springer, New York, NY, 3–14

DUCHI, J., HAZAN, E. AND Y. SINGER (2011): "Adaptive Subgradients Methods for Online Learning and Stochastic Optimization", Journal of Machine Learning Research

FAUL, A.C. AND M.E. TIPPING (2002): "Analysis of sparse Bayesian Learning", Neural Information Processing Systems, 383–389

FERNANDEZ-VILLAVERDE, J. AND J. F. RUBIO-RAMÍREZ (2007): "Estimating Macroeconomic Models: A Likelihood Approach", The Review of Economic Studies, 74(4), 1059–1087

GAL, Y. AND Z. GHAHRAMANI (2016): "Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference", International Conference on Machine Learning

GIANNONE, D., LENZA M. AND G.E. PRIMICERI (2015): "Prior Selection for Vector Autoregressions", The Review of Economics and Statistics, 97, 436–451

GOODFELLOW, I., BENGIO, Y. AND A. COURVILLE (2016): "Deep learning", MIT press, Cambridge

GOODFELLOW, I., POUGET-ABADIE, J., MIRZA, M., XU, B., WARDE-FARLEY, D., OZAIR, S., COURVILLE, A. AND Y. BENGIO (2014): "Generative Adversarial Nets", In Advances in Neural Information Processing Systems, 2672–2680

GRAZZINI, J. AND M. RICHIARDI (2018): "Bayesian Estimation of Macroeconomic Agent-based Model", Manuscript

GRAZZINI, J., RICHIARDI, M. AND M. TSIONAS (2017): "Bayesian Estimation of Agent-based Models", Journal of Economic Dynamics and Control, 77, 26–47

HERBST, E.P. AND F. SCHORFHEIDE (2015): "Bayesian estimation of DSGE models", Princeton University Press

HINTON, G. (2012): "Neural Networks for Machine Learning", Coursera, video lectures.

HOFFMAN, M.D. AND A. GELMAN (2014): "The No-U-turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo", Journal of Machine Learning Research, 15(1), 1593–1623

HUANG, C. W., KRUEGER, D., LACOSTE, A. AND A. COURVILLE (2018): "Neural Autoregressive Flows", arXiv preprint arXiv:1804.00779

JUSTINIANO, A. AND G.E. PRIMICERI (2008): "The Time-varying Volatility of Macroeconomic Fluctuations", American Economic Review, 98(3), 604–641

KARLSSON, S. (2012): "Forecasting with Bayesian Vector Autoregressions", Orebro University

KIM, C. J. AND C. R. NELSON (1998): "Business Cycle Turning Points, a New Coincident Index, and Tests of Duration Dependence Based on a Dynamic Factor Model with Regime Switching", Review of Economics and Statistics, 80(2), 188–201

KINGMA, D. P., AND BA, J. (2014) "Adam: a Method for Stochastic Optimization", arXiv preprint arXiv:1412.6980

KINGMA, D.P. AND P. DHARIWAL (2018): "Glow: Generative Flow with Invertible 1x1 Convolutions", arXiv preprint arXiv:1807.03039

KINGMA, D. P., SALIMANS, T., JOZEFOWICZ, R., CHEN, X., SUTSKEVER, I. AND M. WELLING (2016): "Improved Variational Inference with Inverse Autoregressive Flow", Neural Information Processing Systems, 4743–4751

KINGMA, D.P., SALIMANS, T. AND M. WELLING (2015): "Variational Dropout and the Local Reparametrization Trick", Neural Information Processing Systems

KINGMA, D.P. AND M. WELLING (2014): "Auto-encoding Variational Bayes", International Conference on Learning Representation

KOOP, G. AND D. KOROBILIS (2010): "Bayesian Multivariate Time Series Methods for Empirical Macroeconomics", Foundations and Trends in Econometrics, 3(4), 267–358

KOROBILIS, D. (2017): "Forecasting with many predictors using message passing algorithms", Essex Finance Center Working Paper

KOOP, G. AND D. KOROBILIS (2018): "Variational Bayes inference in high-dimensional time-varying parameter models", Essex Finance Center Working Paper

KRUEGER, D., HUANG, C., ISLAM, R., TURNER, R., LACOSTE, A. AND A. COURVILLE (2018): "Bayesian Hypernetworks", arXiv preprint arXiv:1710.04759

KUCUKELBIR, A., TRAN, D., RANGANATH, R., GELMAN, A. AND D.M. BLEI (2017): "Automatic Differentiation Variational Inference", Journal of Machine Learning Research, 1–45

KUSHNER, H. AND G.G. YIN (2003): "Stochastic Approximation and Recursive Algorithms and Applications", Springer-Verlag New York, 35

LI, Y. AND R.E. TURNER (2016): "Rényi Divergence Variational Inference", In Advances in Neural Information Processing Systems (pp. 1073–1081).

LITTERMAN, R. (1980): "A Bayesian Procedure for Forecasting with Vector Autoregressions", Working Paper, Massachusetts Institute of Technology

LOSCHILOV, I. AND F. HUTTER (2017): "SGDR: Stochastic Gradient Descent with Warm Restarts", International Conference on Learning Representations

LOUIZOS, C., ULLRICH K. AND M. WELLING (2017): "Bayesian Compression for Deep Learning", Neural Information Processing Systems

LUX, T. (2018): "Estimation of Agent-based Models using Sequential Monte Carlo Methods", Journal of Economic Dynamics and Control, 91, 391–408

MAROWKA, M., PETERS, D.W., KANTAS, N. AND G. BAGNAROSA (2017): "Some Recent Developments in Markov Chain Monte Carlo for Cointegrated Time Series", ESAIM Proceedings and Surveys, 59, 76–103

MCCRAKEN, M.W. AND S. NG (2016): "FRED-MD: A Monthly Database for Macroeconomic Research", Journal of Business and Economic Statistics, 34(4), 574–589

MINKA, T.P. (2001): "Expectation Propagation for Approximate Bayesian Inference", In Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence, 362–369

MOLCHANOV, D., ASHUKHA, A. AND D. VETROV (2017): "Variational Dropout Sparsifies Deep Neural Networks", arXiv preprint arXiv:1701.05369

NAESSETH, C.A., LINDERMAN, S.W., RANGANATH, R. AND D.M. BLEI (2018): "Variational Sequential Monte Carlo", International Conference on Artificial Intelligence and Statistics

NEAL, R.M. (2011): "MCMC Using Hamiltonian Dynamics", Handbook of Markov Chain Monte Carlo, 2(11), 2

NESTEROV, Y. (1983): "A Method of Solving a Convex Programming Problem with Convergence rate $O(1/k^2)$", Soviet Mathematics Doklady, 27, 372–376

ORMEROD, J.T. AND M.P. WAND (2010): "Explaining Variational Approximation", The American Statistician, 64(2), 140–153

OTROK, C. AND C.H. WHITEMAN (1998): "Bayesian Leading Indicators: Measuring and Predicting Economic Conditions in Iowa", International Economic Review, 997–1014

OWEN, A. (2013): "Monte Carlo Theory, Methods and Examples"

PAPAMAKARIOS, G., PAVLAKOU, T. AND I. MURRAY (2017): "Masked Autoregressive Flow for Density Estimation", Neural Information Processing Systems, 2338–2347

POLYAK, B.T. (1963): "Some Methods of Speed up the Convergence of Iteration Methods", USSR Computational Mathematics and Mathematical Physics, 4(5), 1–17

REZENDE, D. J. AND S. MOHAMED (2015): "Variational Inference with Normalizing Flows", arXiv preprint arXiv:1505.05770

TIPPING, M.E. (2001): "Sparse Bayesian Learning and Relevance Vector Machine", Journal of Machine Learning Research, 1, 211–244

TRAN, D., HOFFMAN, M.D., SAUROUS, R.A., BREVDO, E., MURPHY, K. AND D.M. BLEI (2017): "Deep Probabilistic Programming", International Conference on Learning Representation

TRAN, D., KUCUKELBIR, A., DIENG, A.B., RUDOLPH, M., LIANG, D. AND M. BLEI (2017): "Edward: a Library for Probabilistic Modeling, Inference, and Ctiticism", arXiv preprint arXiv:1610.09787

SALVATIER, J., WIECKI, T.V. AND C. FONNESBECK (2016): "Probabilistic Programming in Python using PyMC3", Peer J Computer Science

SELEZNEV, S. (2018): "Truncated Priors for tHDP-VAR", Manuscript

SIMS, C. (1983): "A Nine-Variable Probabilistic Macroeconomic Forecasting Model", Business Cycles, Indicators and Forecasting", NBER Chapters, 179–212

SMETS, F. AND R. WOUTERS (2003): "An Estimated Dynamic Stochastic General Equilibrium Model of the Euro Area", Journal of the European Economic Association, 1(5), 1123–1175

SMETS, F. AND R. WOUTERS (2007): "Shocks and Frictions in US Business Cycles: a Bayesian DSGE Approach", American Economic Review, 97(3), 586–606

STAN DEVELOPMENT TEAM (2016): "Stan Modeling Language Users Guide and Reference Manual"

STOCK, J.H. AND M.W. WATSON (2016): "Dynamic Factor Models, Factor-augmented Vector Autoregressions, and Structural Vector Autoregressions in Macroeconomics", Handbook of Macroeconomics, 2, 415–525

VAN DEN BERG, R., HASENCLEVER, L., TOMCZAK, J. M. AND M. WELLING (2018): "Sylvester Normalizing Flows for Variational Inference", arXiv preprint arXiv:1803.05649

VILLANI, M. (2009): "Steady-state Priors for Vector Autoregressions", Journal of Applied Econometrics, 24(4), 630–650

WAINWRIGHT, M.J. AND M.I. JORDAN (2008): "Graphical Models, Exponential Families, and Variational Inference", Foundations and Trends in Machine Learning, 1(1–2), 1–305

WIPF, D.P. AND S.S. NAGARAJAN (2007): "A New View of Automatic Relevance Determination", Neural Information Processing Systems

Appendix A

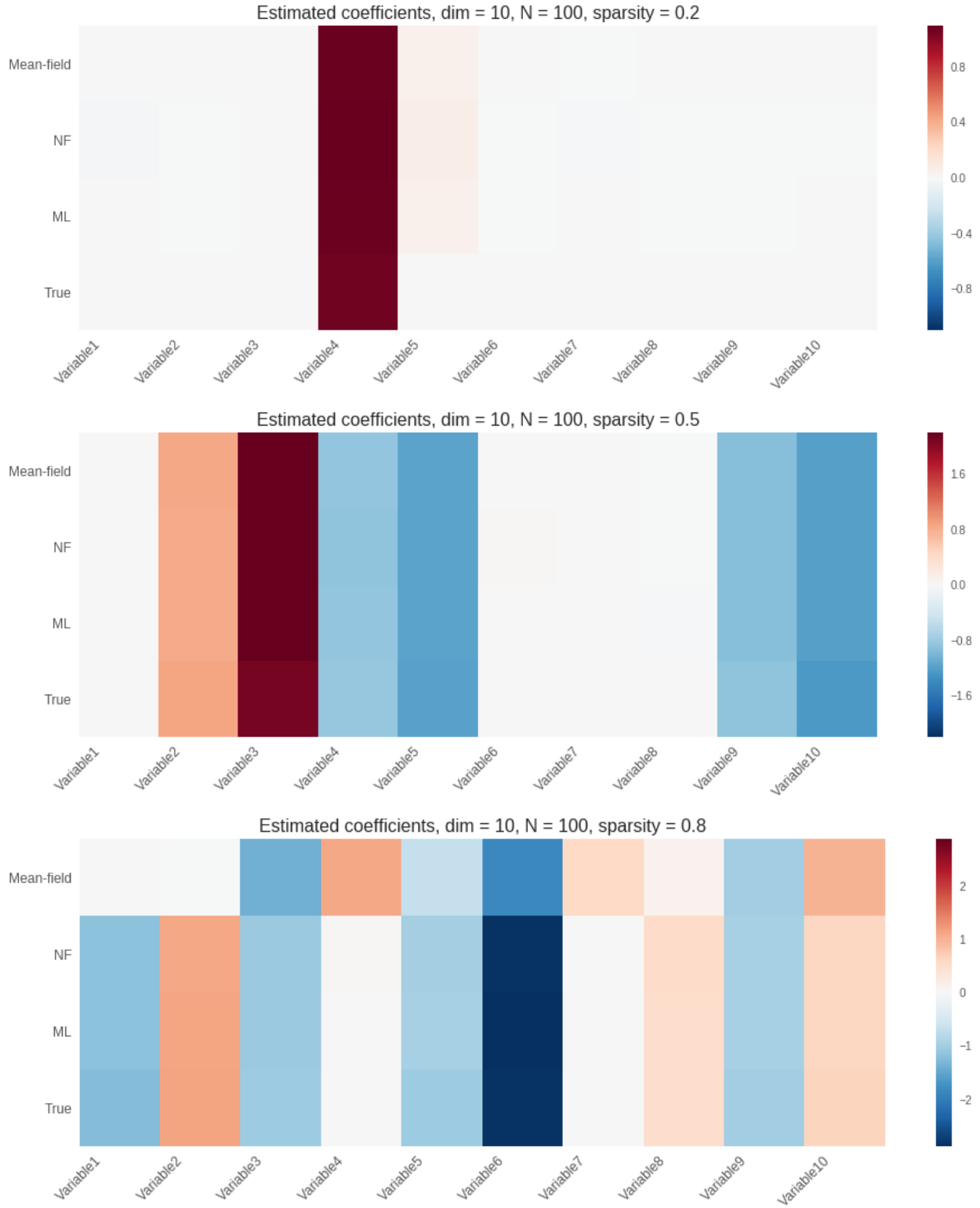

Figure 1. Artificial data estimates for sparse Bayesian learning regression, 10 covariates

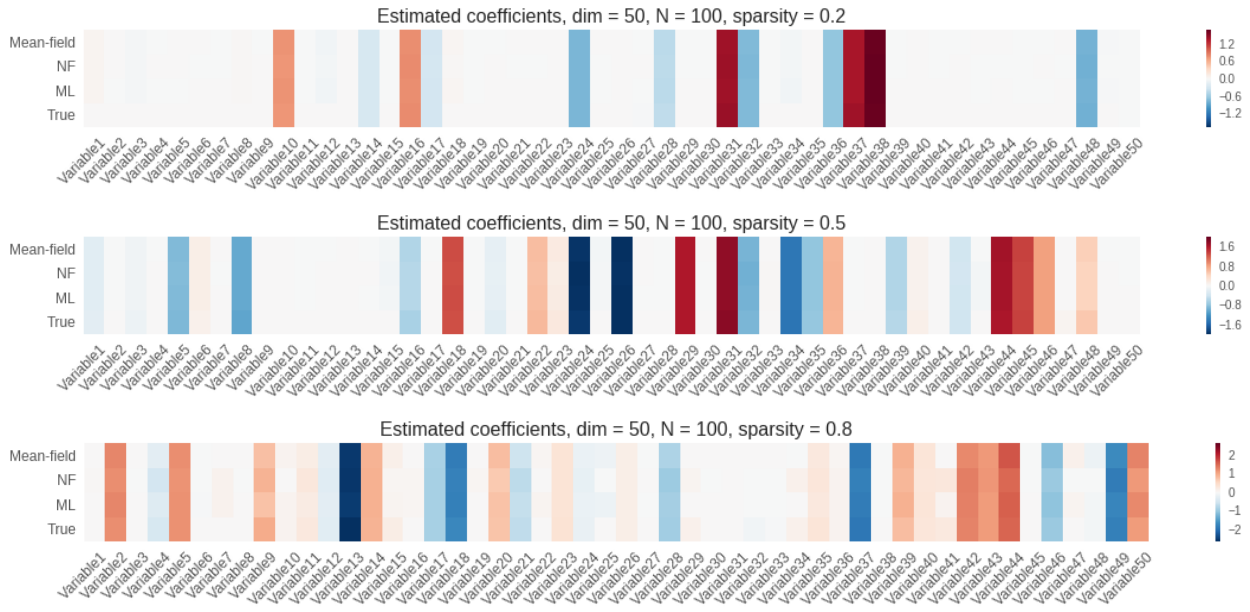

Figure 2. Artificial data estimates for sparse Bayesian learning regression, 50 covariates

Figure 3. Artificial data for Bayesian vector autoregression with sparse priors and t-Student errors

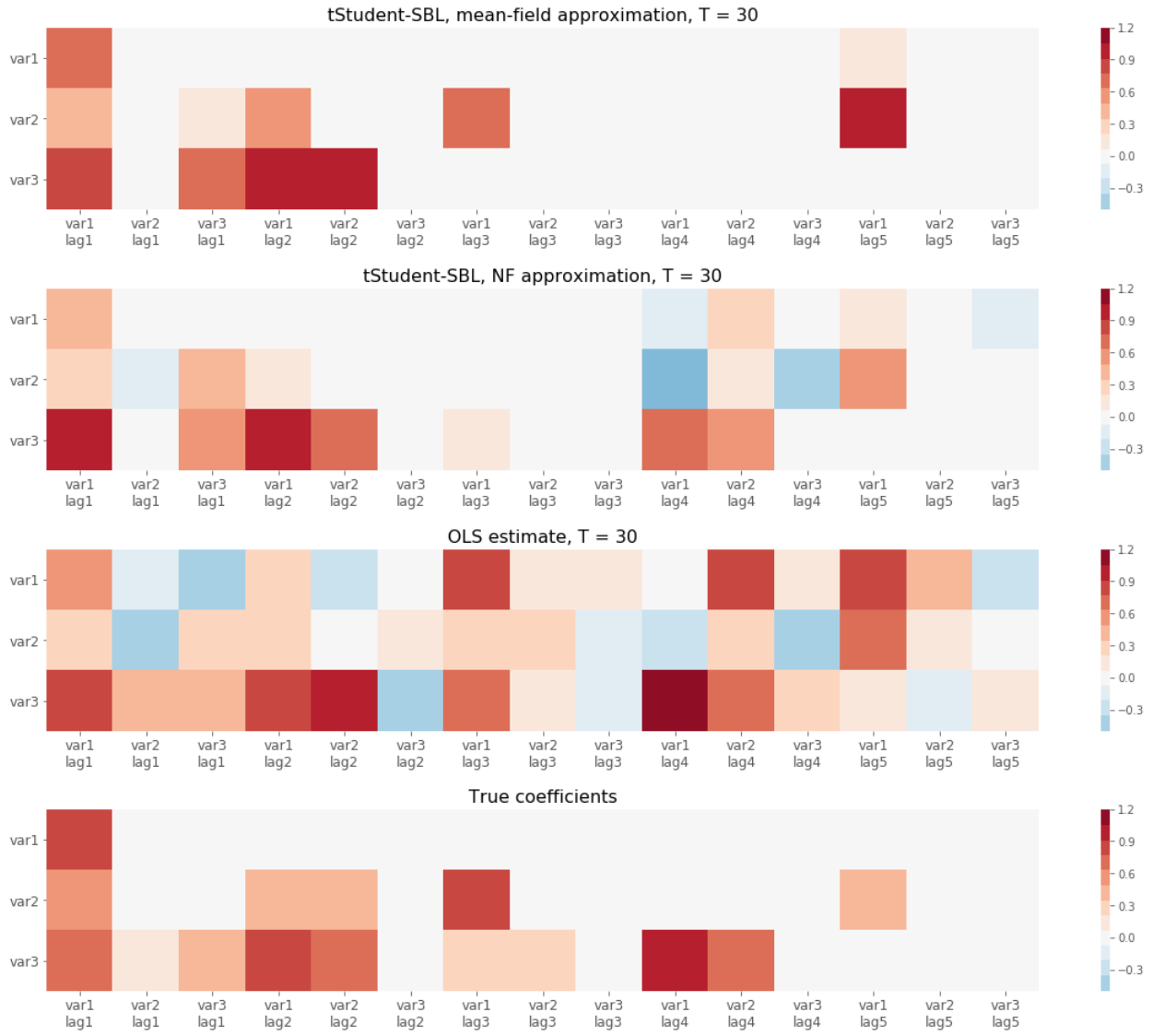

Figure 4. Estimation results for matrix B using 30 points from artificial data

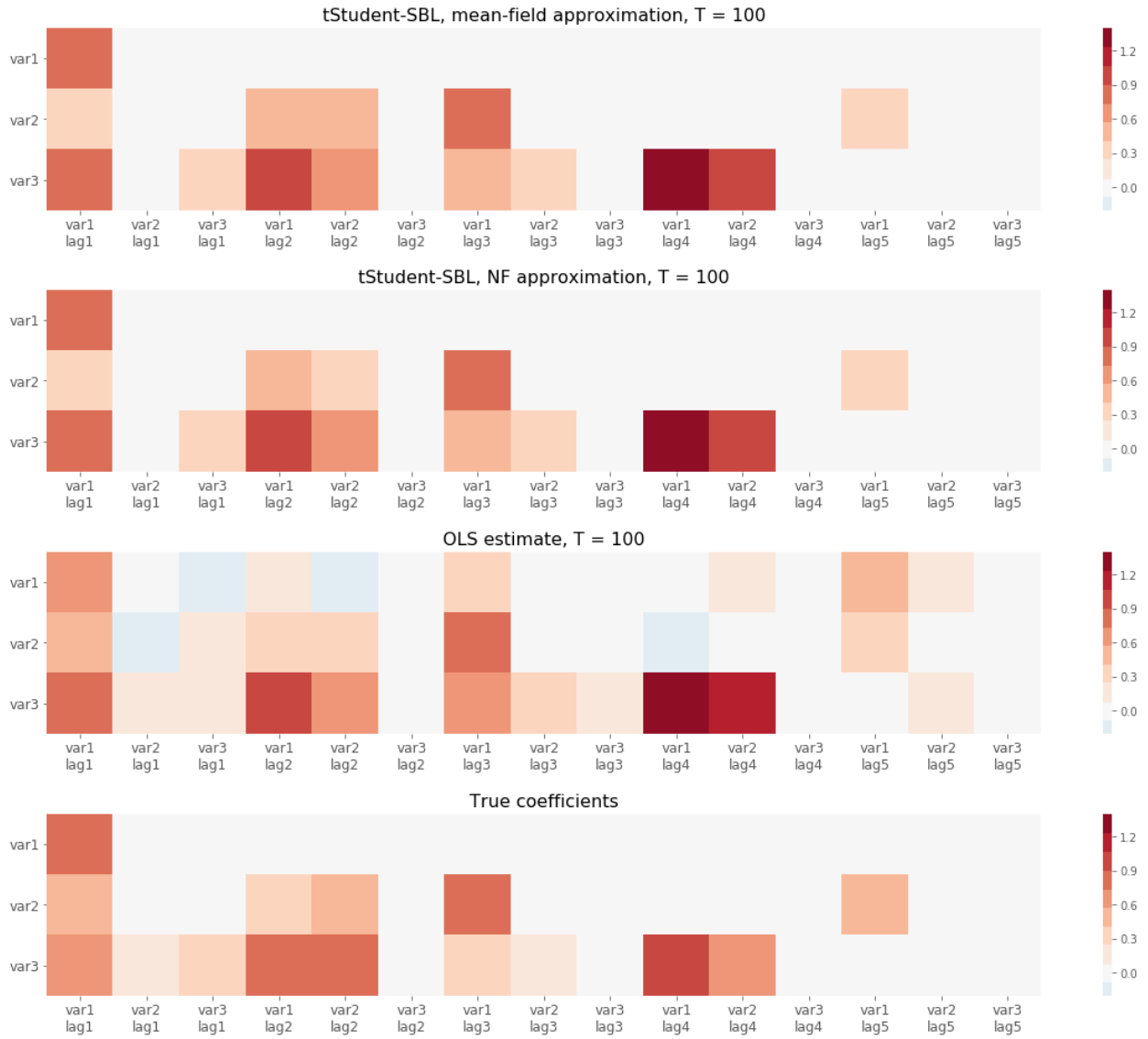

Figure 5. Estimation results for matrix B using 100 points from artificial data

Figure 6. Estimation results for matrix B using 1000 points from artificial data

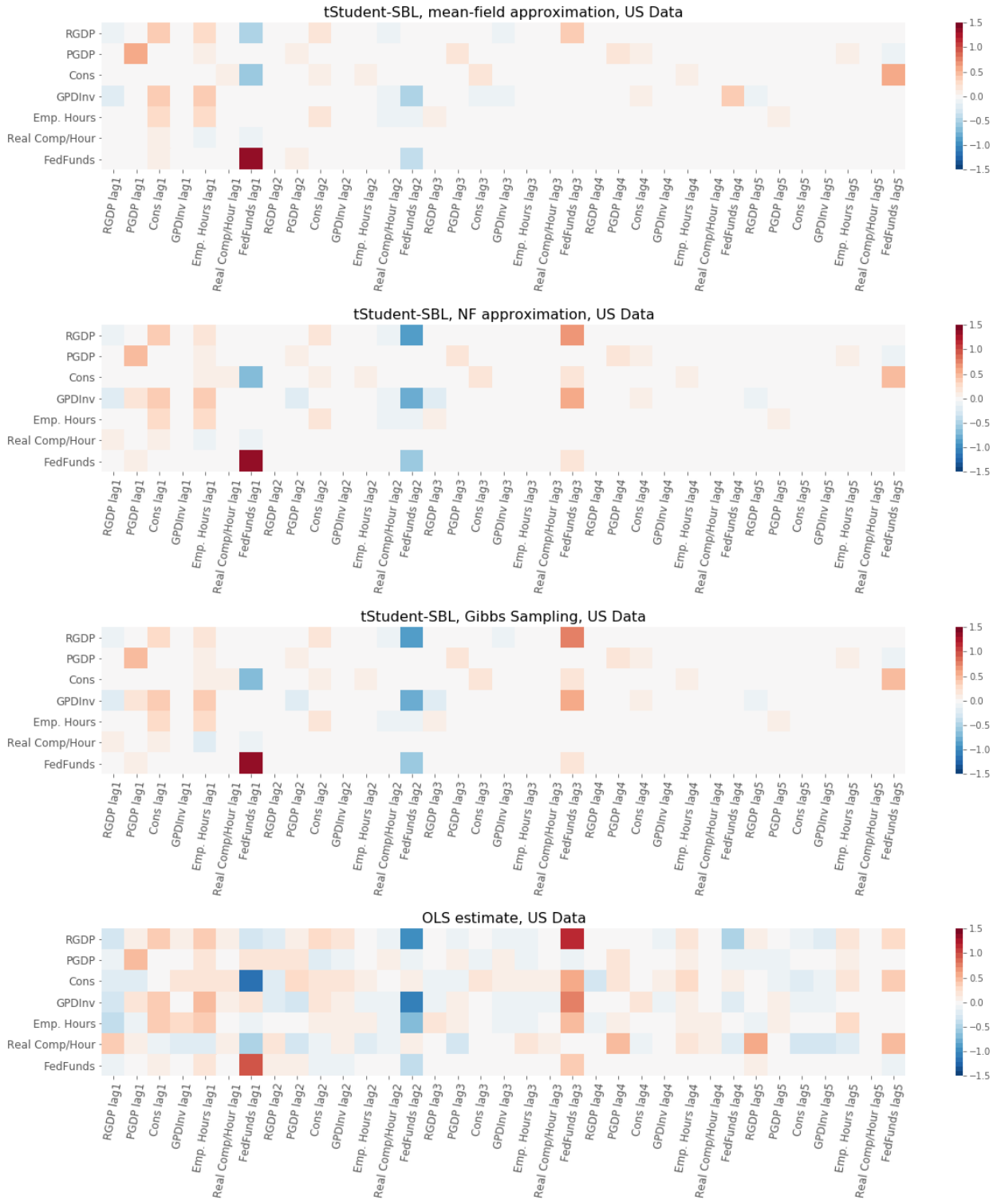

Figure 7. Estimation results for matrix B, US Data

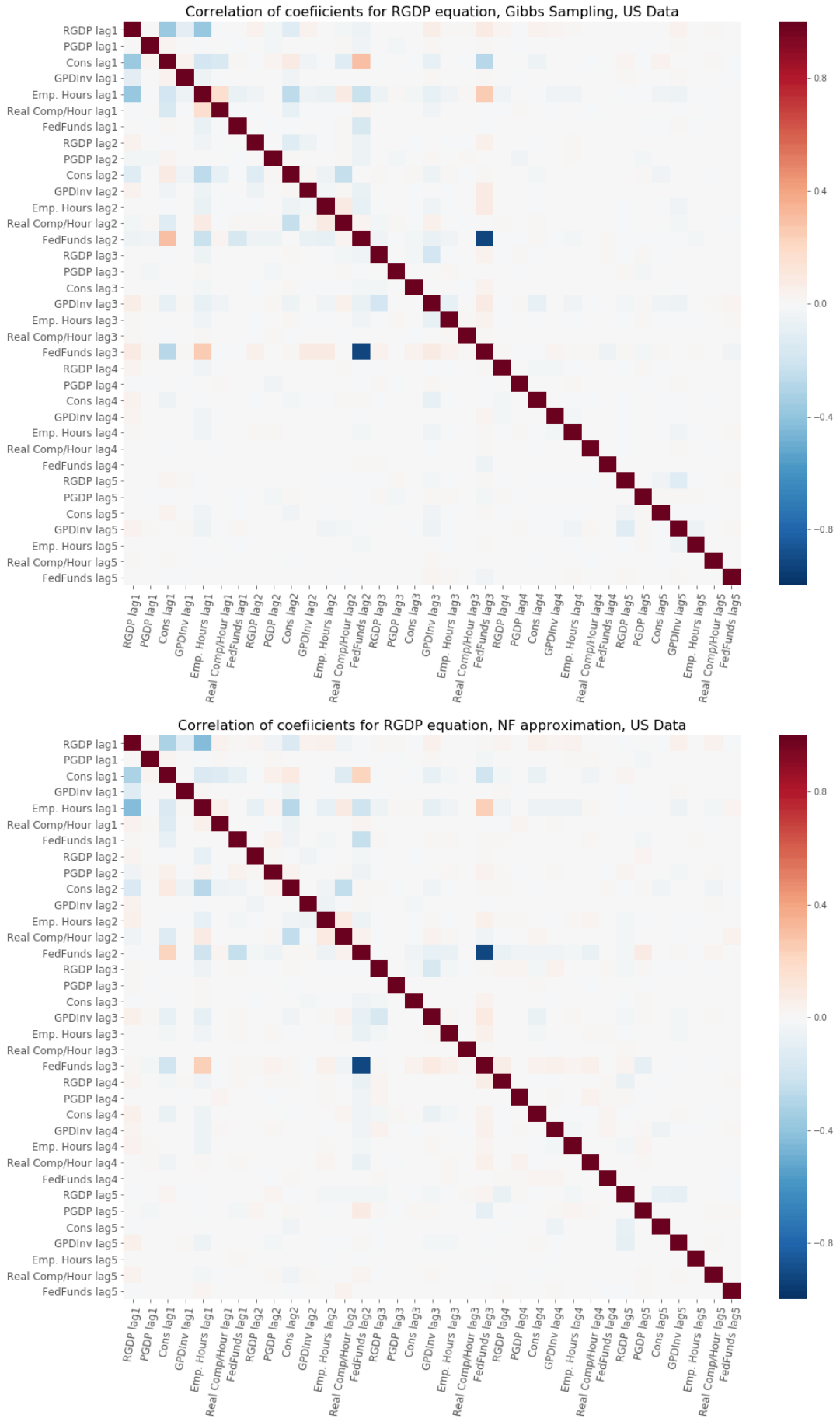

Figure 8. Correlation of coefficients, Bayesian vector autoregression with sparse priors and t-Student errors

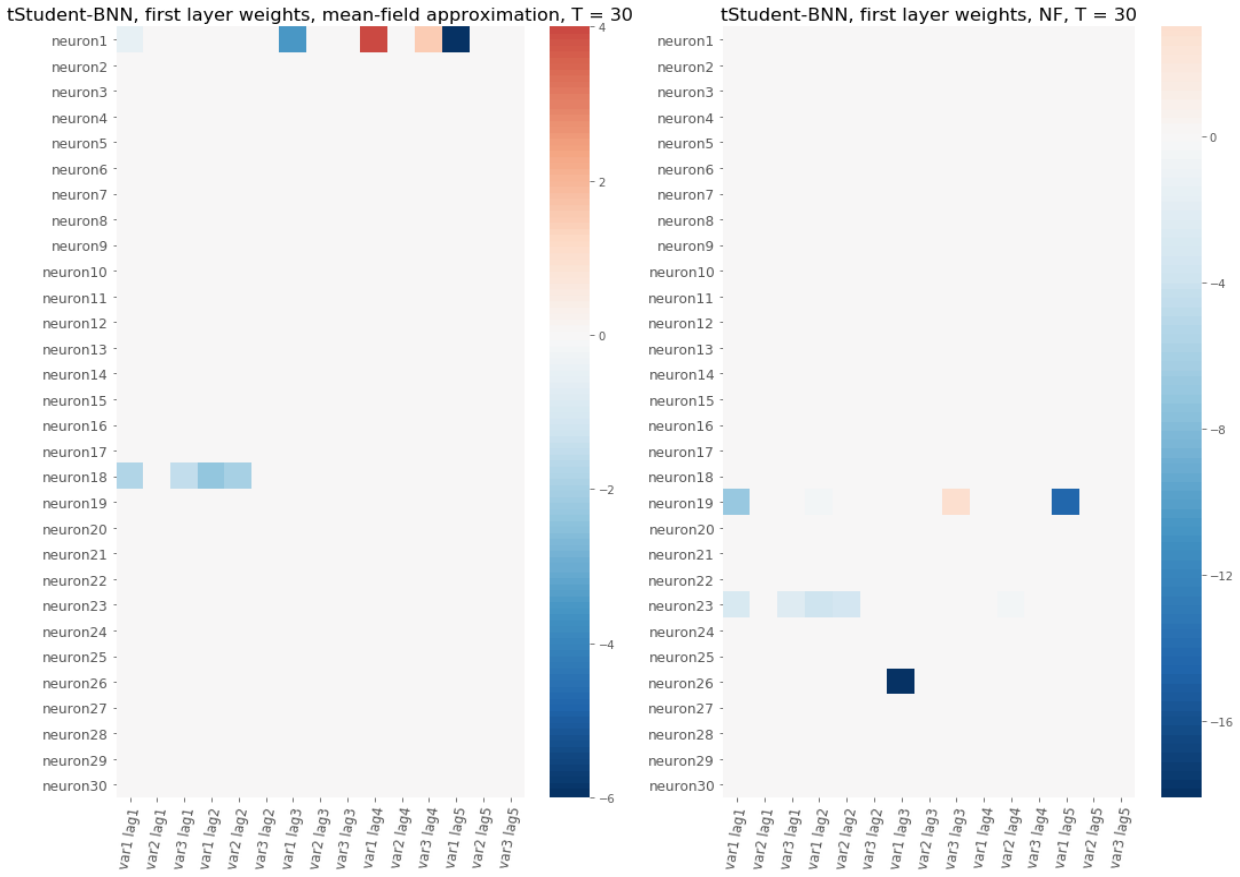

Figure 9. Estimation results for matrix W_1 using 30 points from artificial data

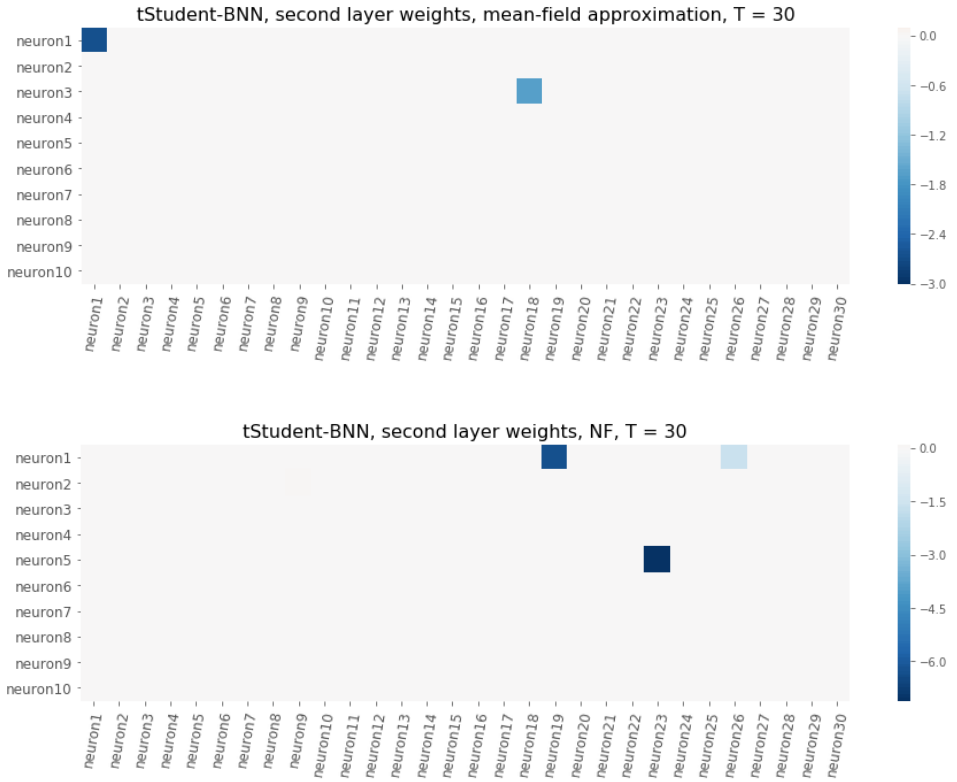

Figure 10. Estimation results for matrix W_2 using 30 points from artificial data

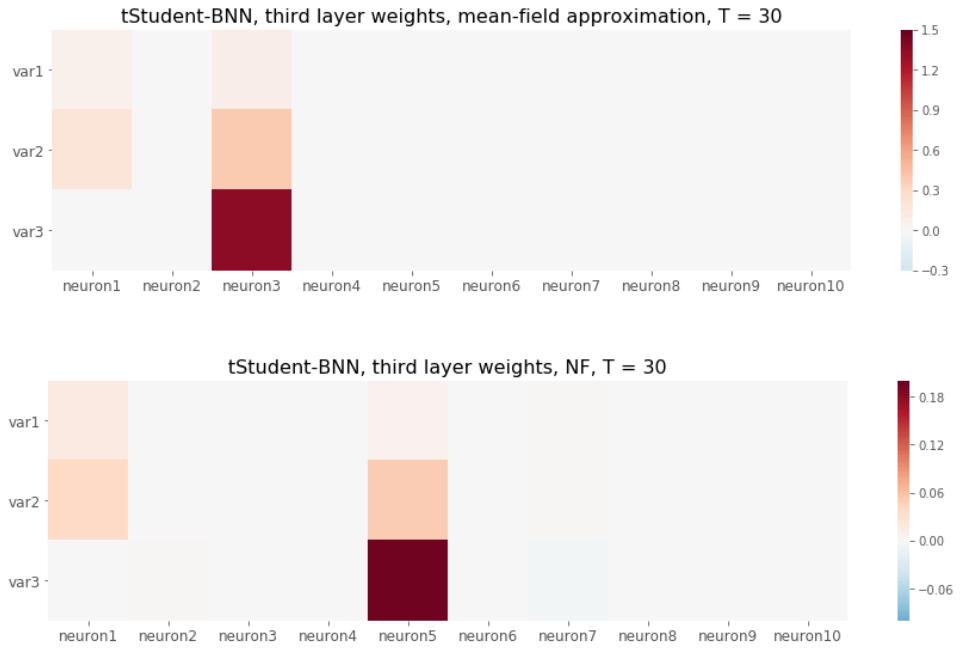

Figure 11. Estimation results for matrix W_3 using 30 points from artificial data

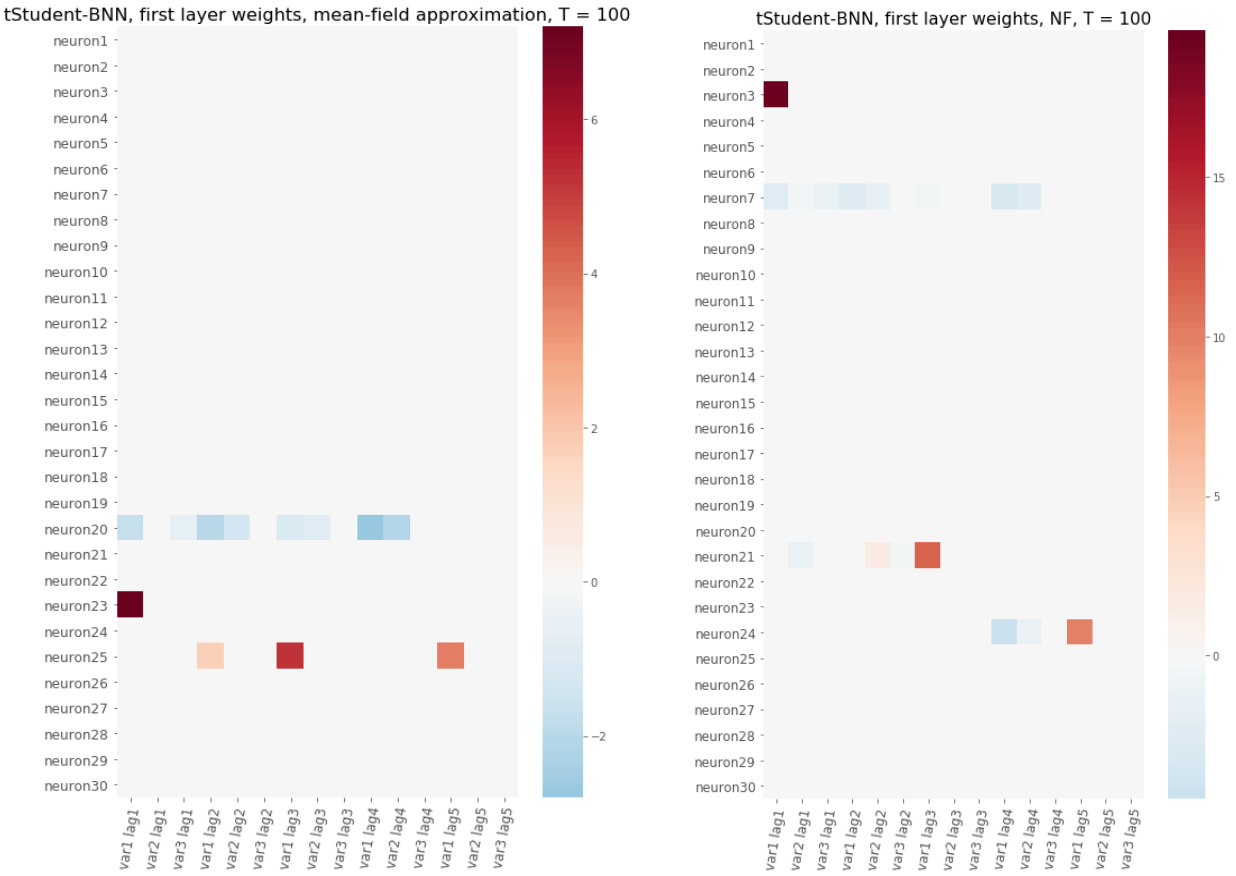

Figure 12. Estimation results for matrix W_1 using 100 points from artificial data

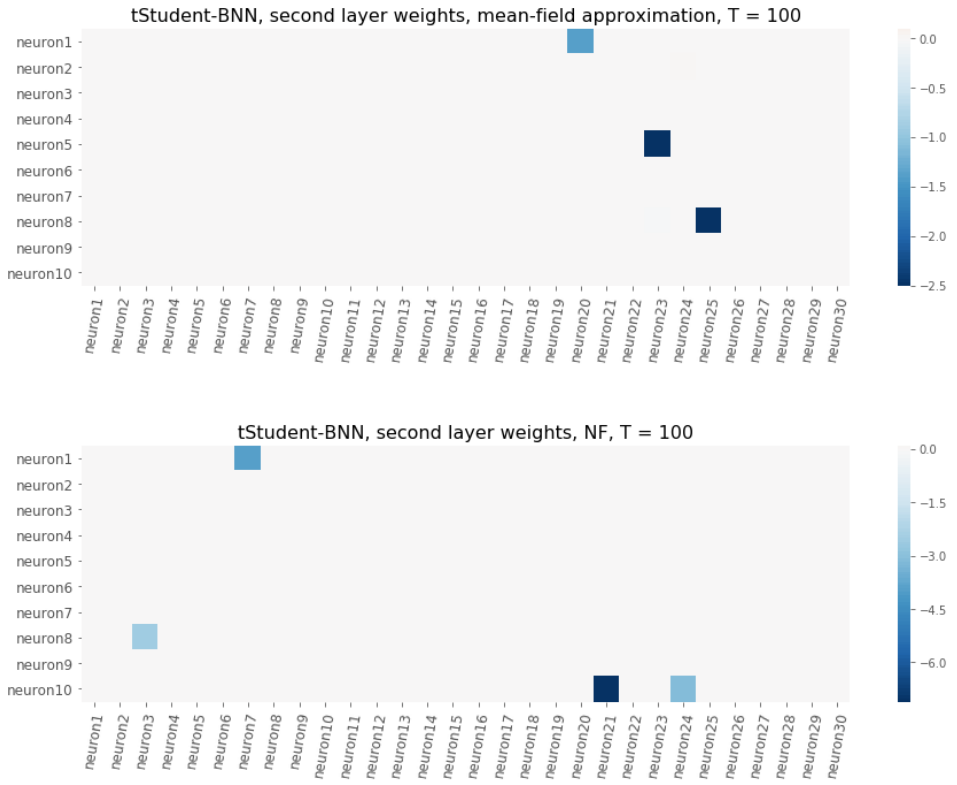

Figure 13. Estimation results for matrix W_2 using 100 points from artificial data

Figure 14. Estimation results for matrix W_3 using 100 points from artificial data

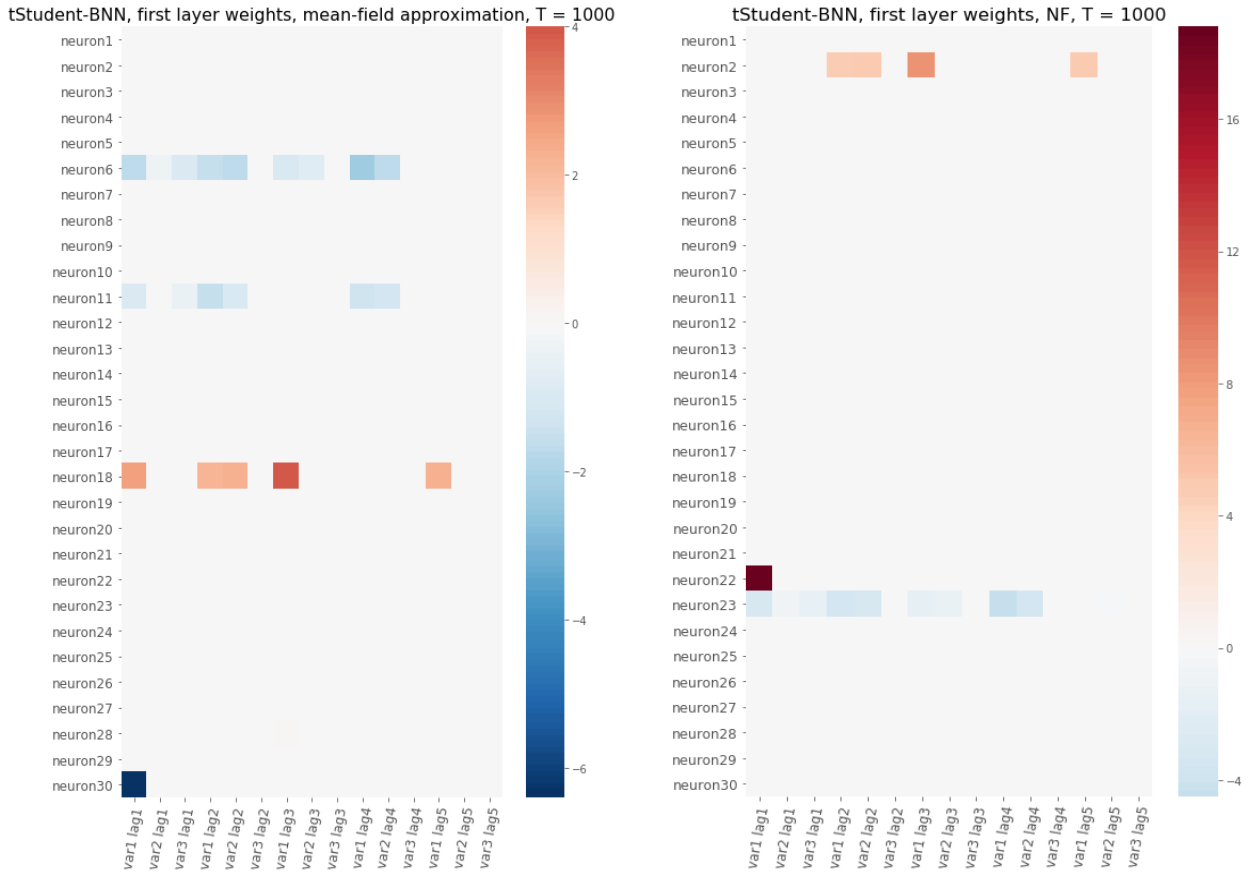

Figure 15. Estimation results for matrix W_1 using 1000 points from artificial data

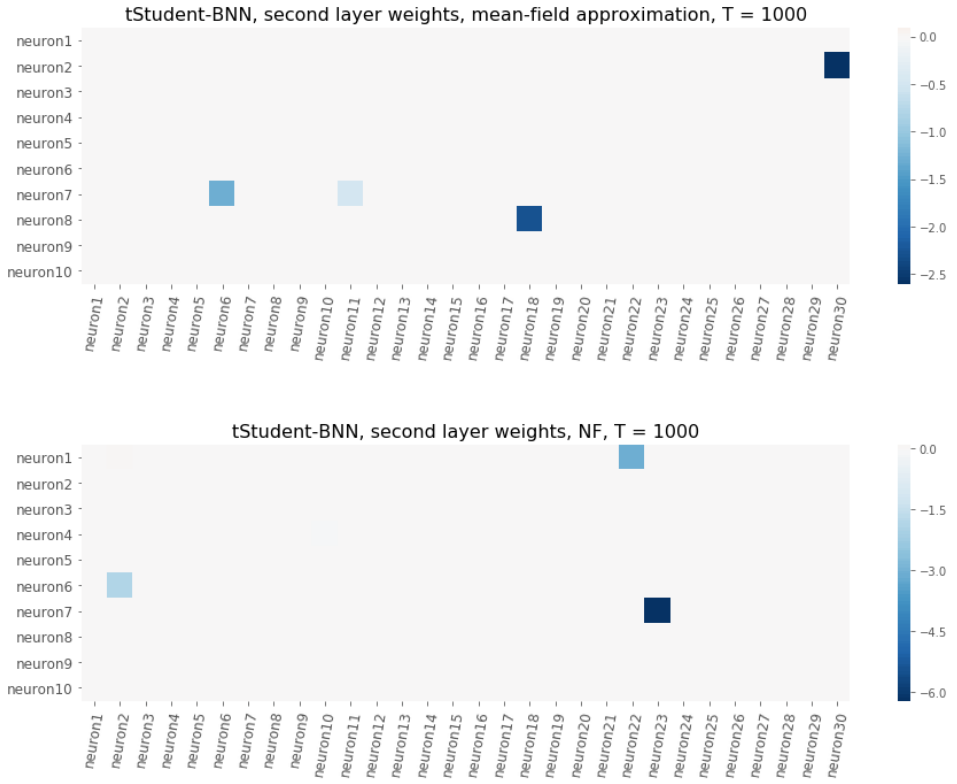

Figure 16. Estimation results for matrix W_2 using 1000 points from artificial data

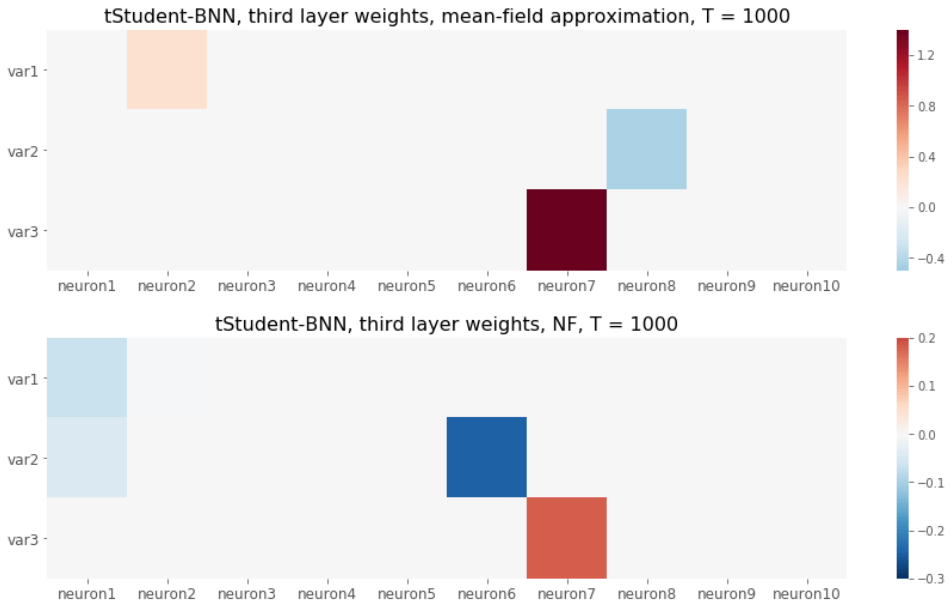

Figure 17. Estimation results for matrix W_3 using 1000 points from artificial data

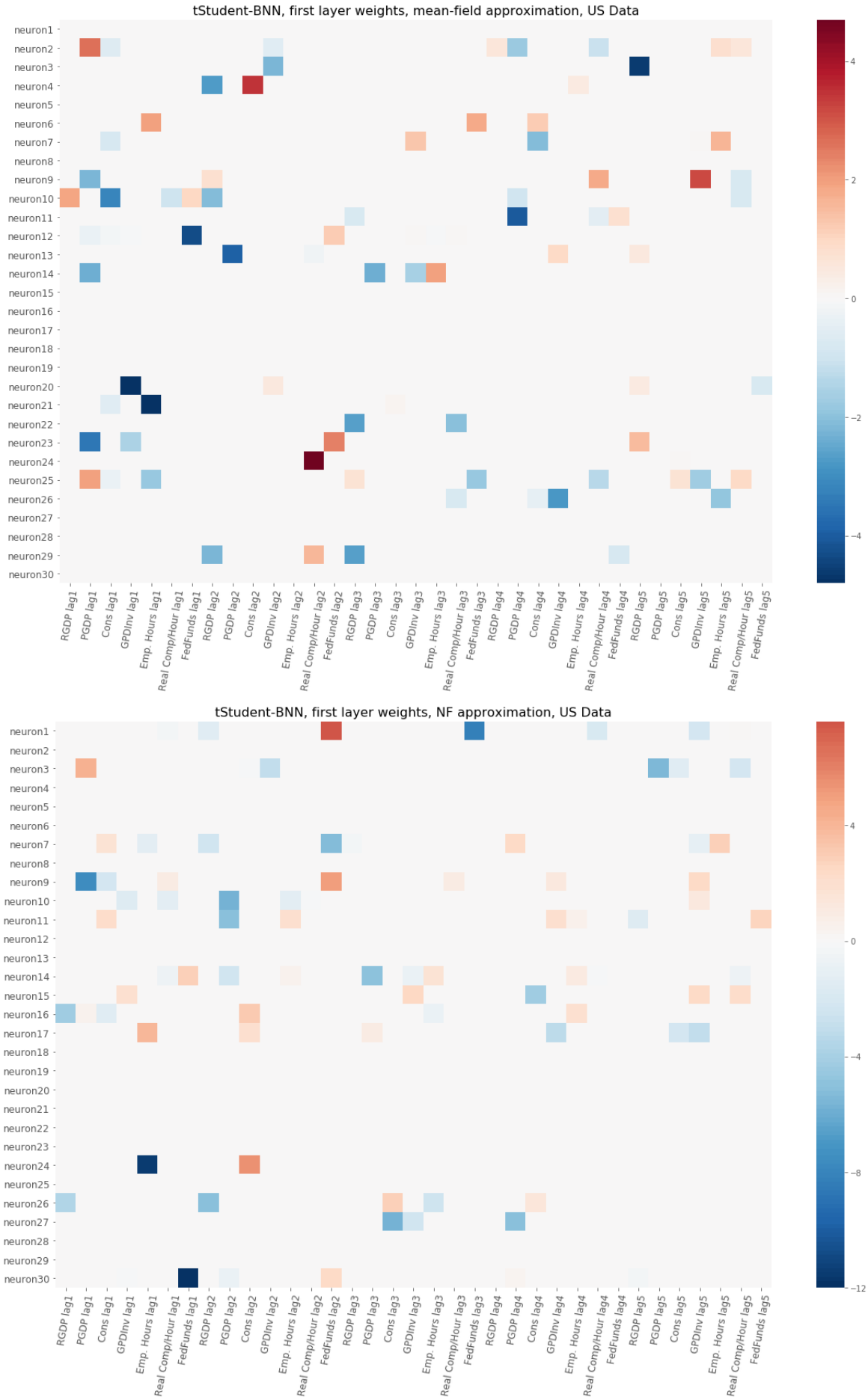

Figure 18. Estimation results for matrix W_1, US Data

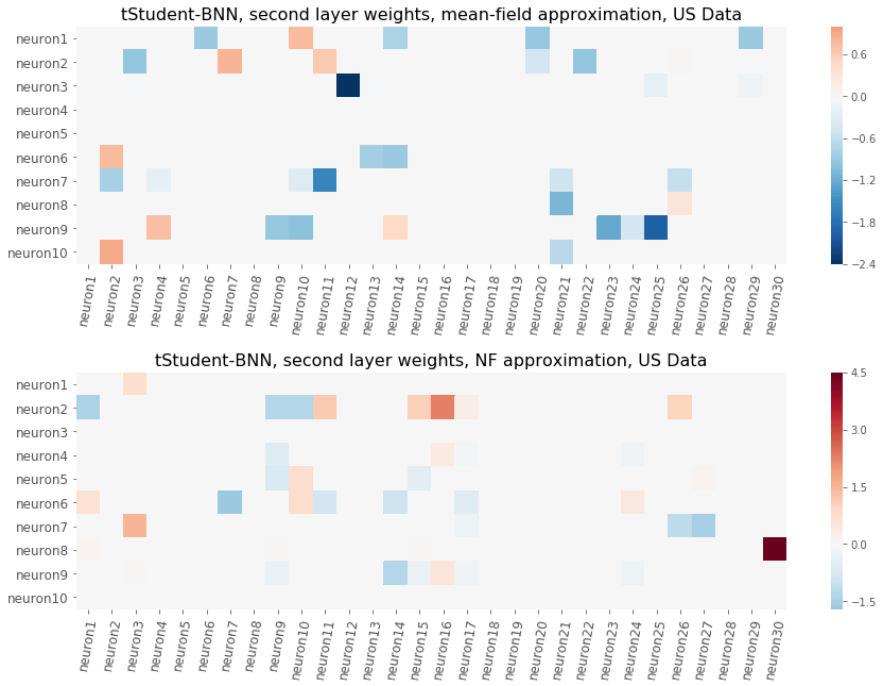

Figure 19. Estimation results for matrix W_2, US Data

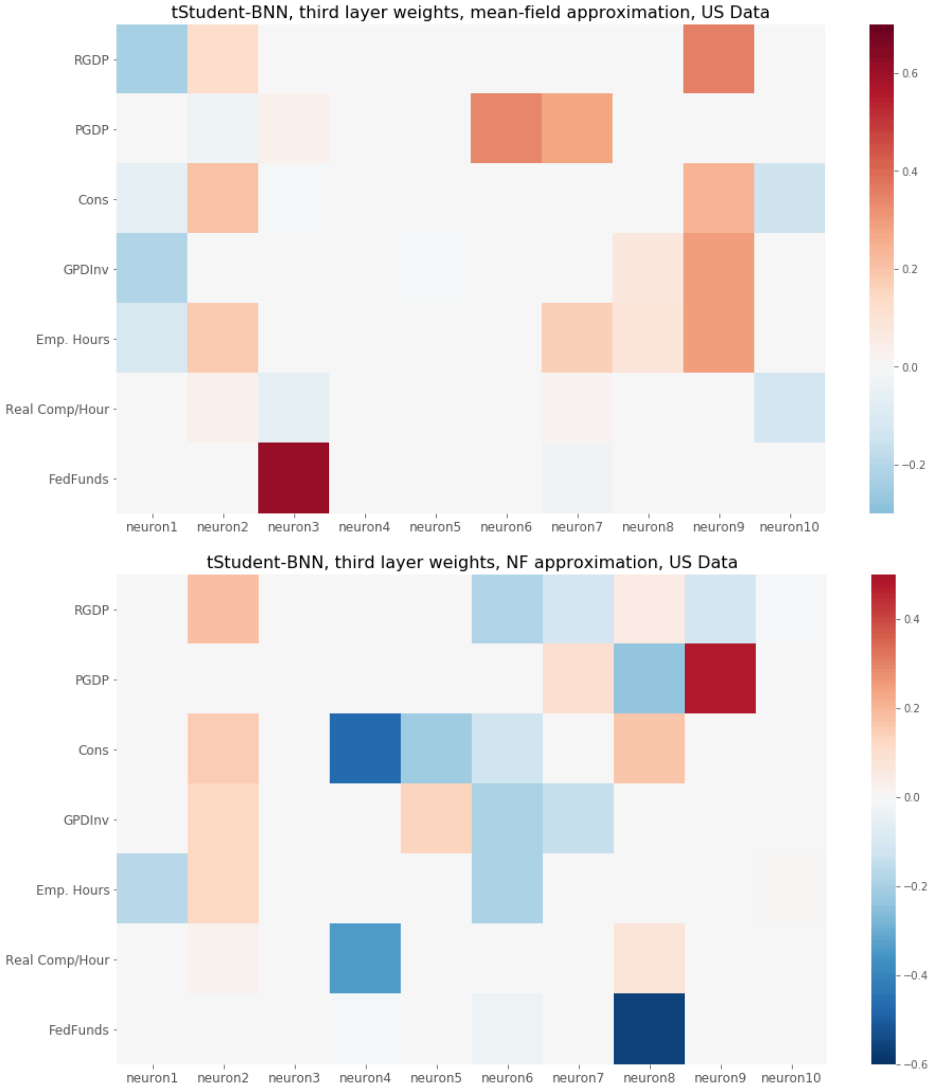

Figure 20. Estimation results for matrix W_3, US Data

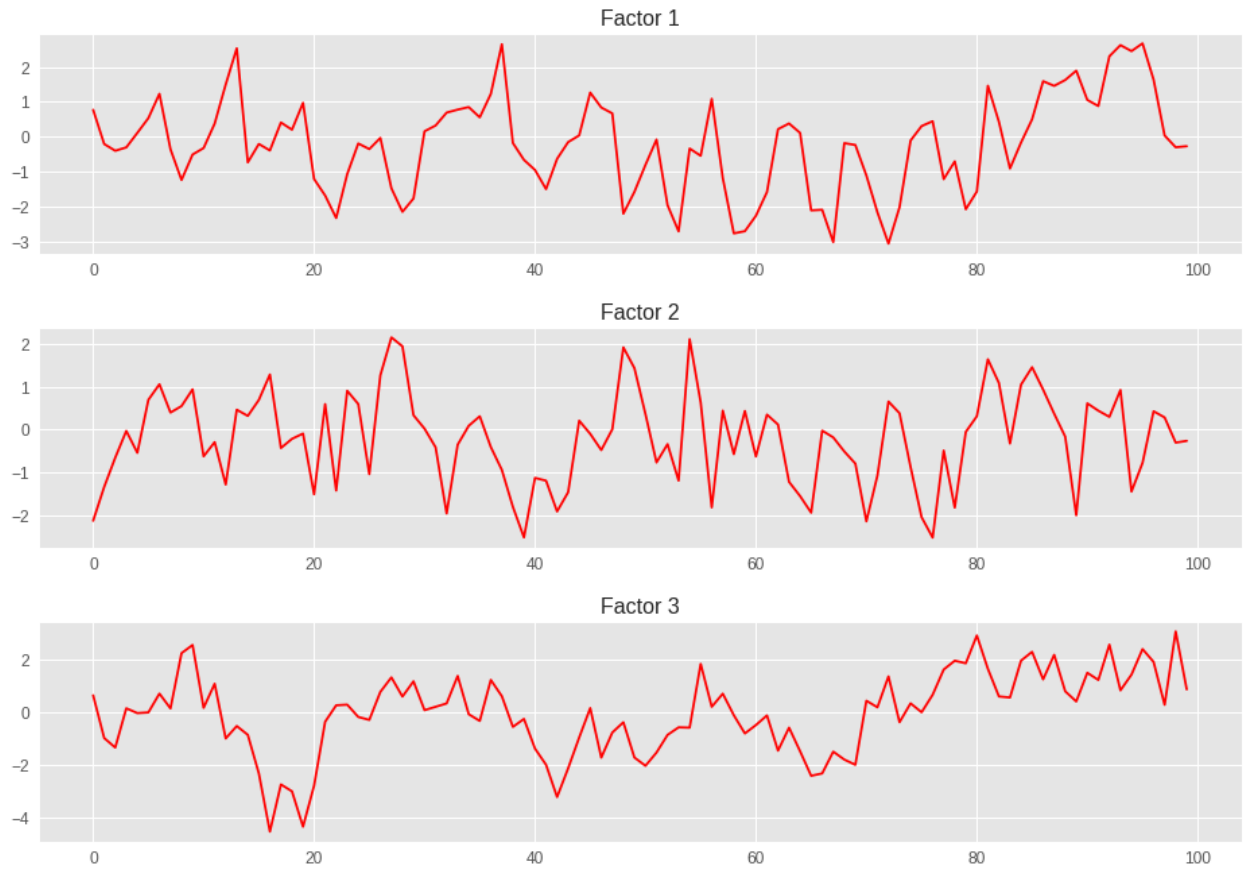

Figure 21. Artificial factors for DFM model

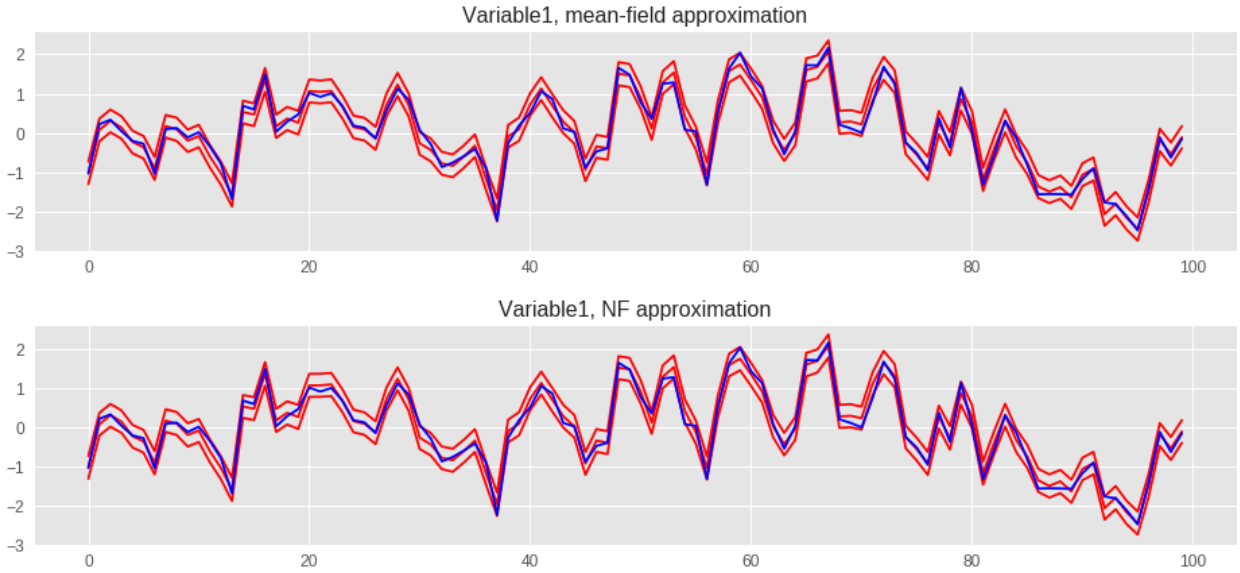

Figure 22. Recovered artificial data (5th, 50th and 95th quantities), DFM

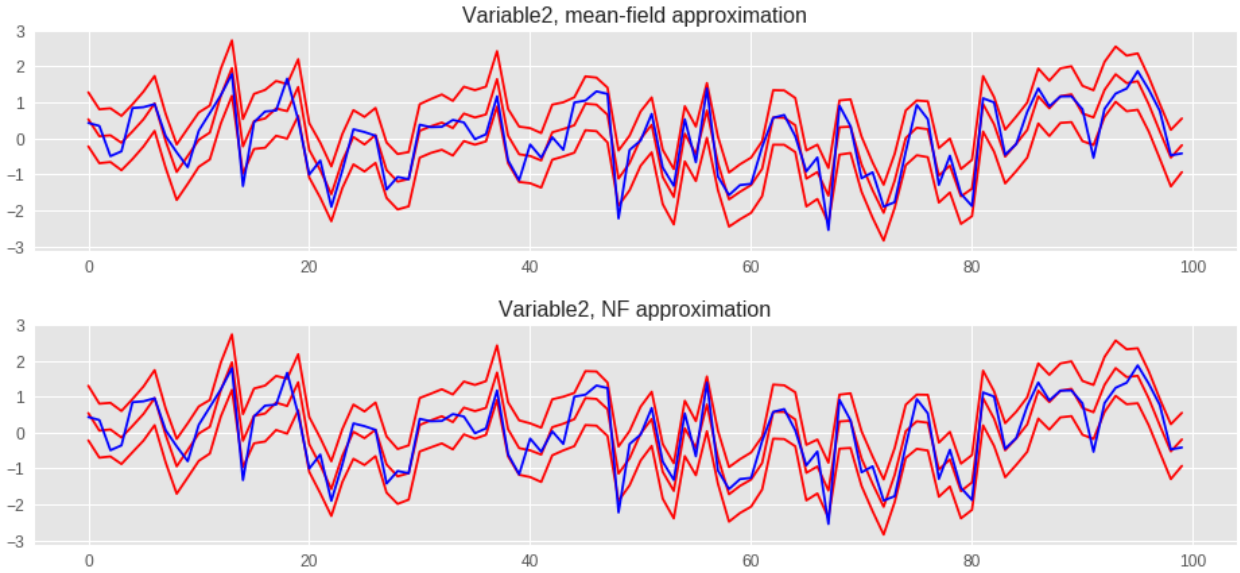

Figure 23. Recovered artificial data (5th, 50th and 95th quantities), DFM

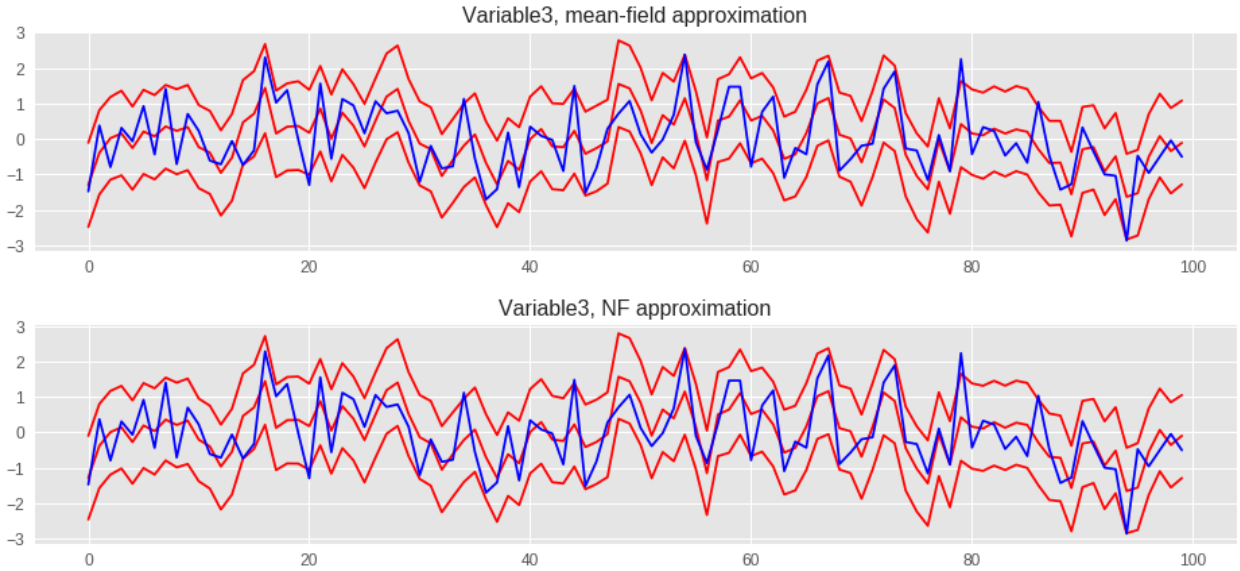

Figure 24. Recovered artificial data (5th, 50th and 95th quantities), DFM

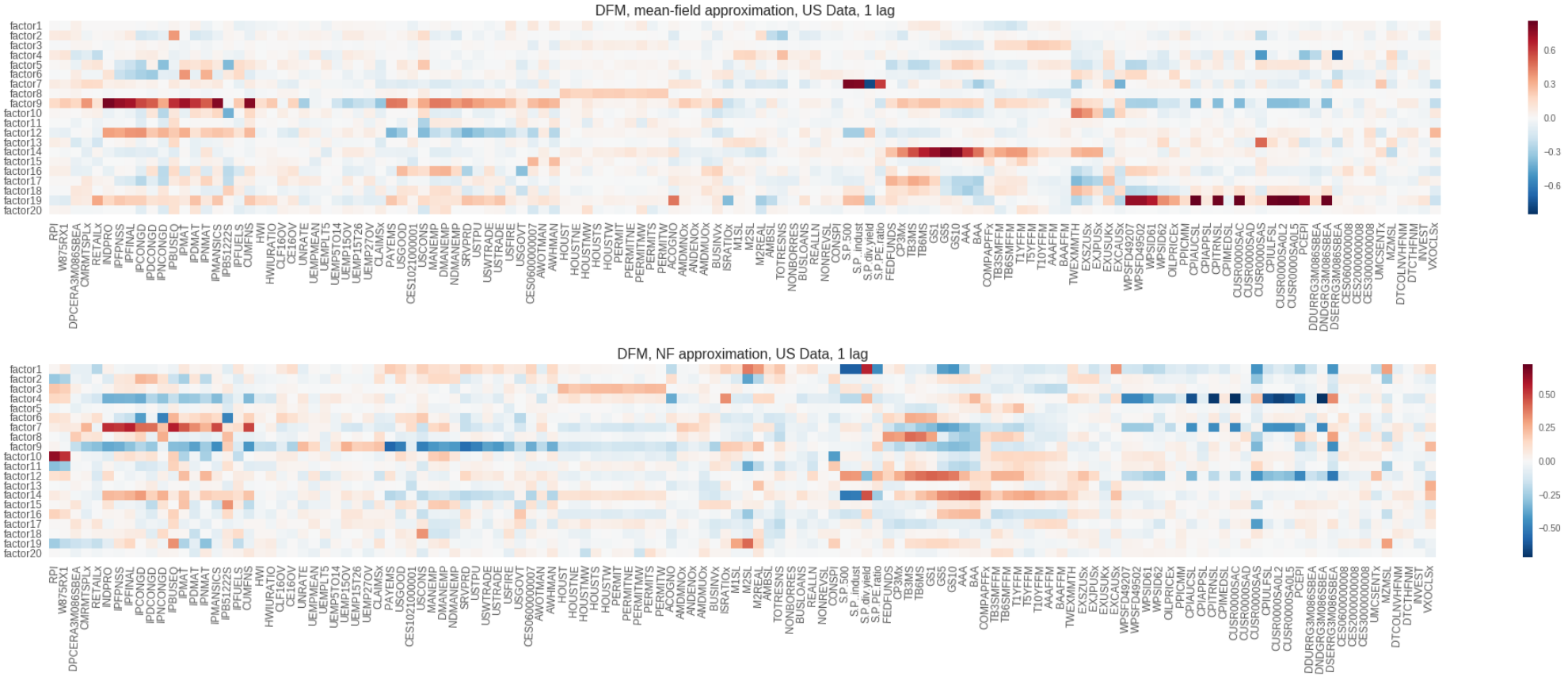

Figure 25. Factor loadings, US Data

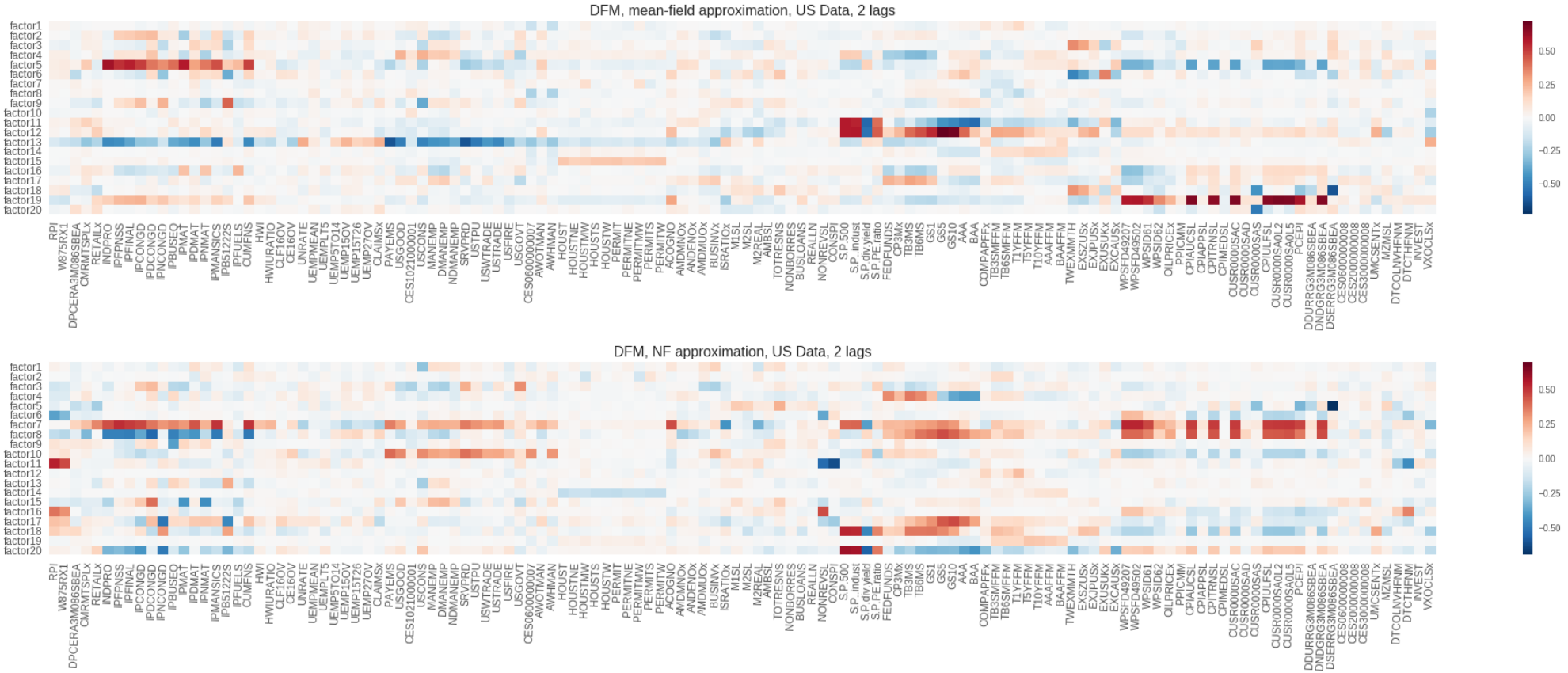

Figure 26. Factor loadings, US Data

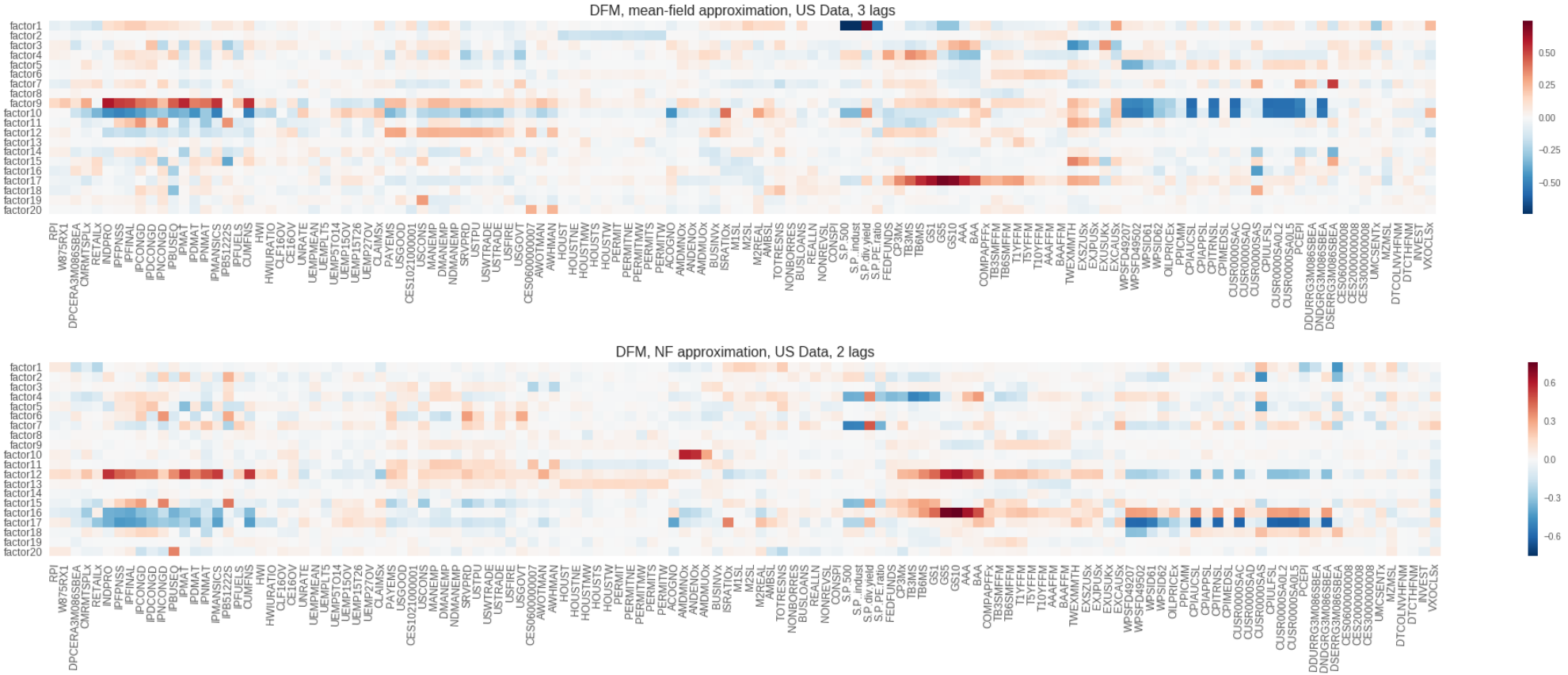

Figure 27. Factor loadings, US Data

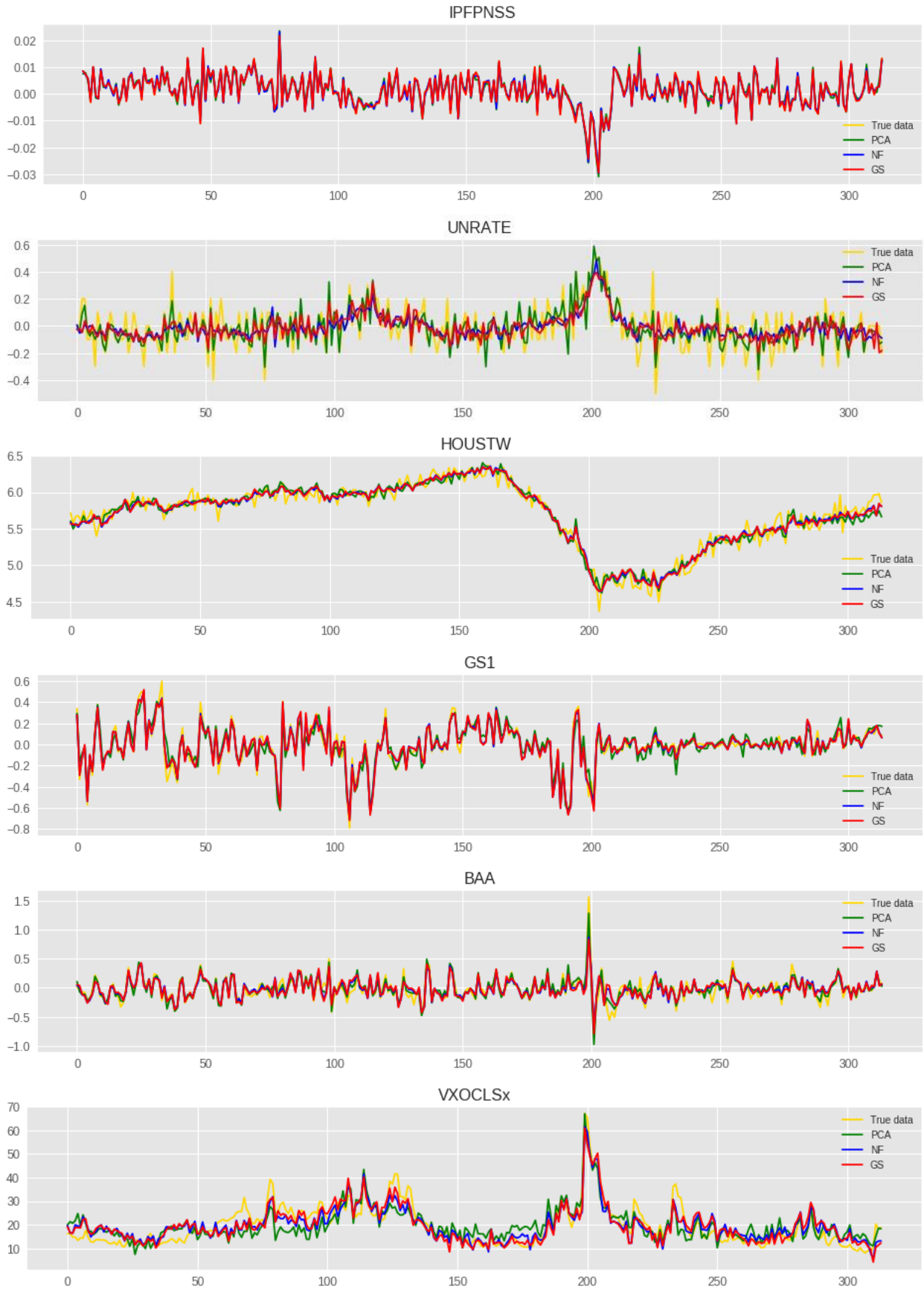

Figure 28. Recovered US data (50th quantity), DFM

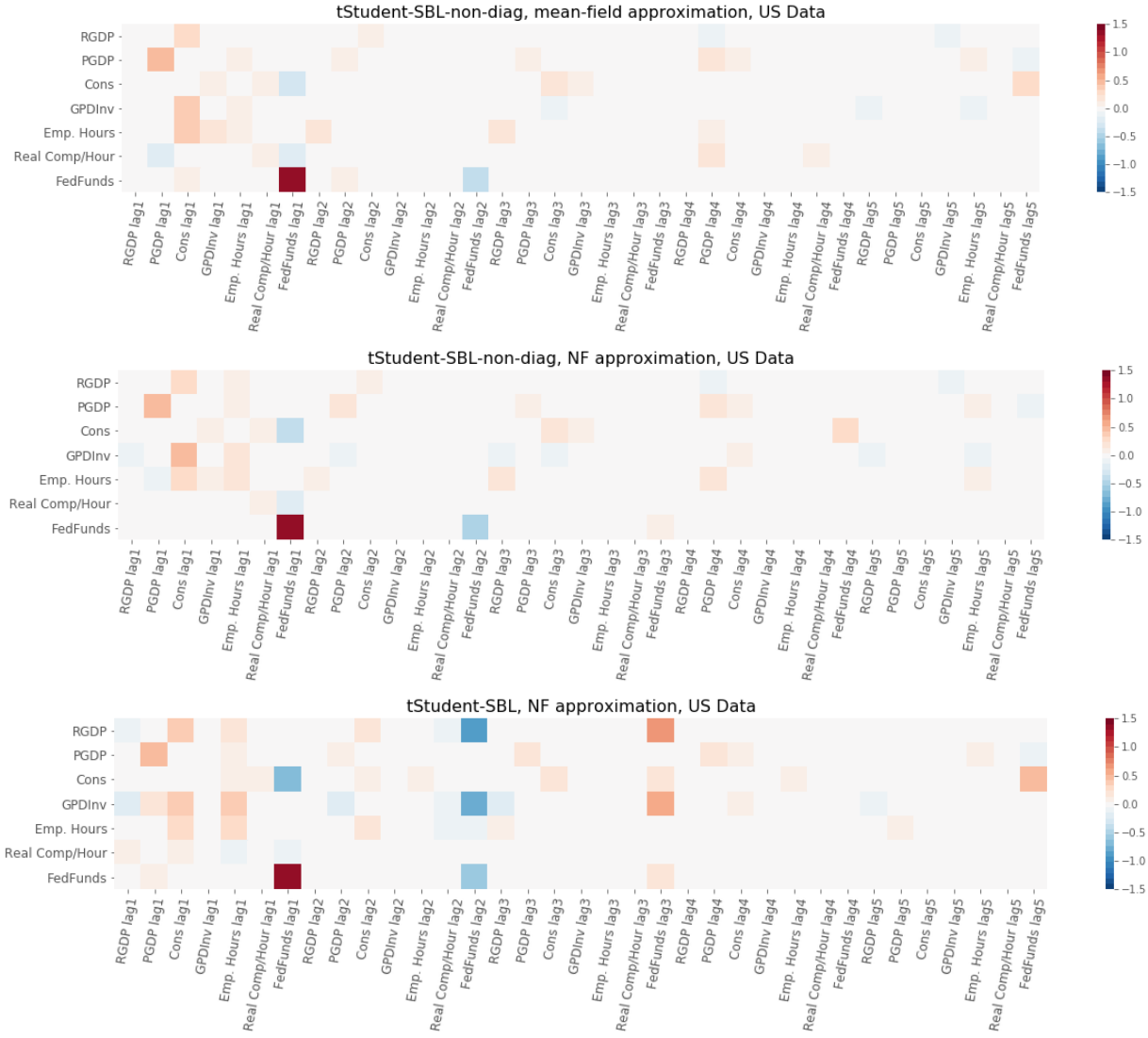

Figure 29. Estimation results for matrix B, non-diagonal, US Data

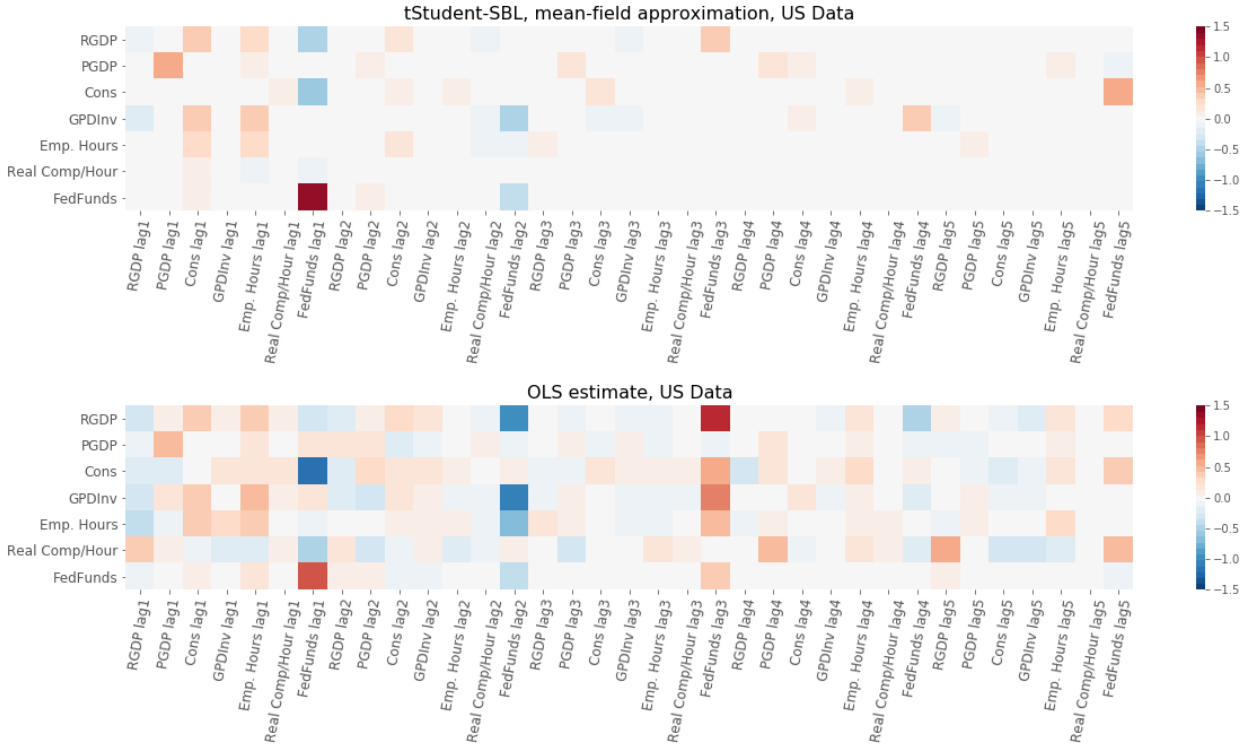

Figure 29. Estimation results for matrix B, non-diagonal, US Data (continues)

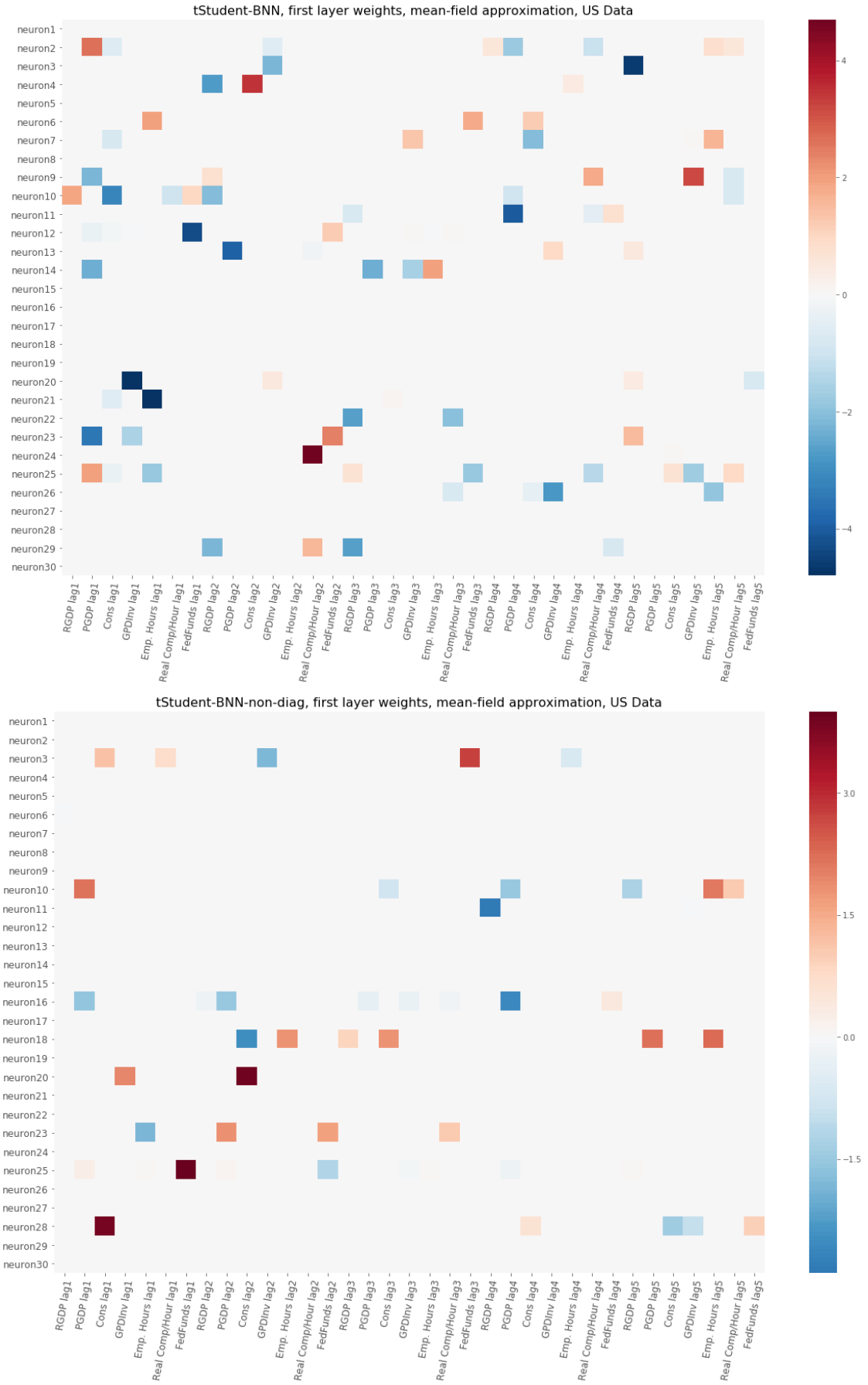

Figure 30. Estimation results for matrix W_1, mean-field , non-diagonal, US Data

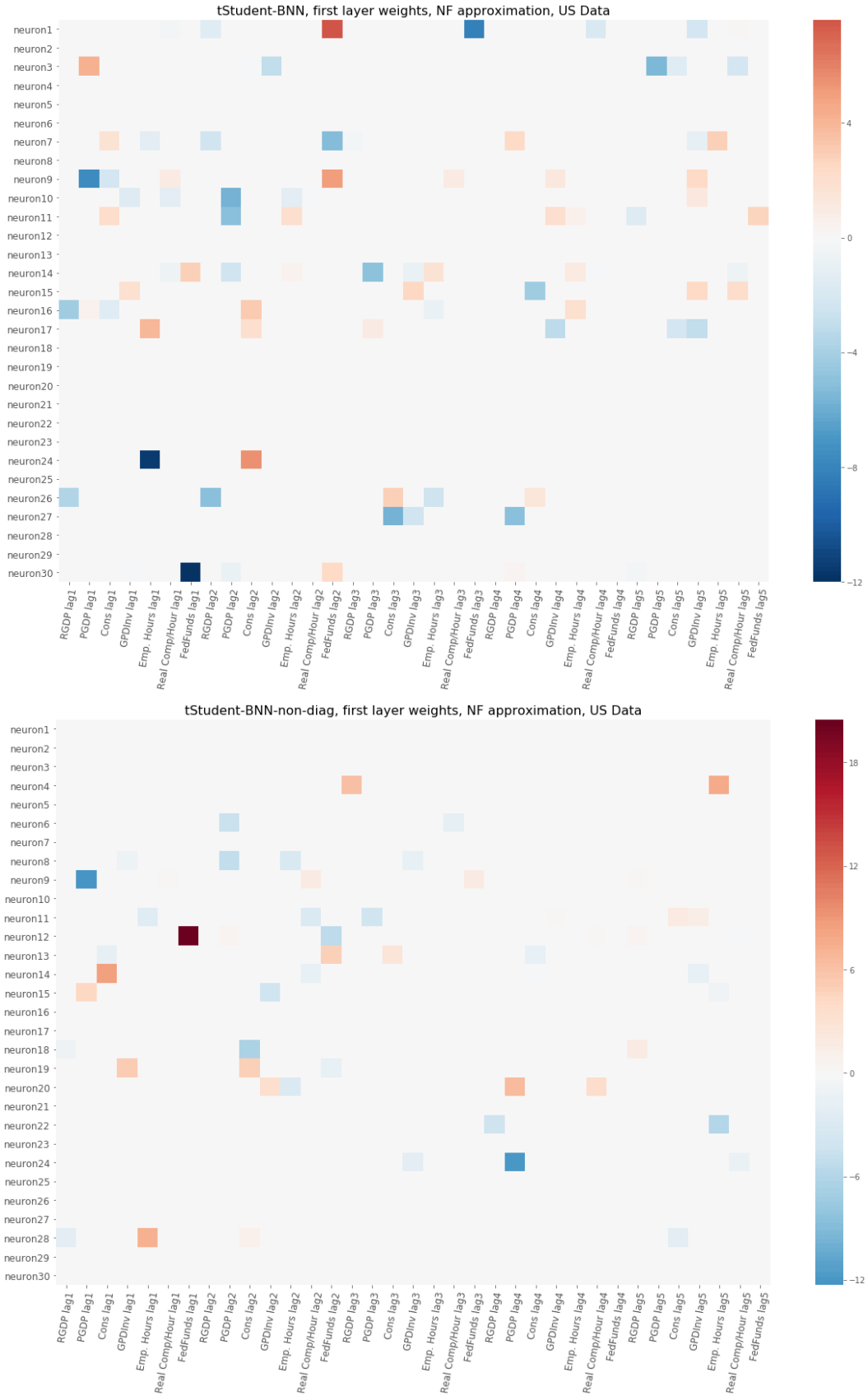

Figure 31. Estimation results for matrix W_1, NF , non-diagonal, US Data

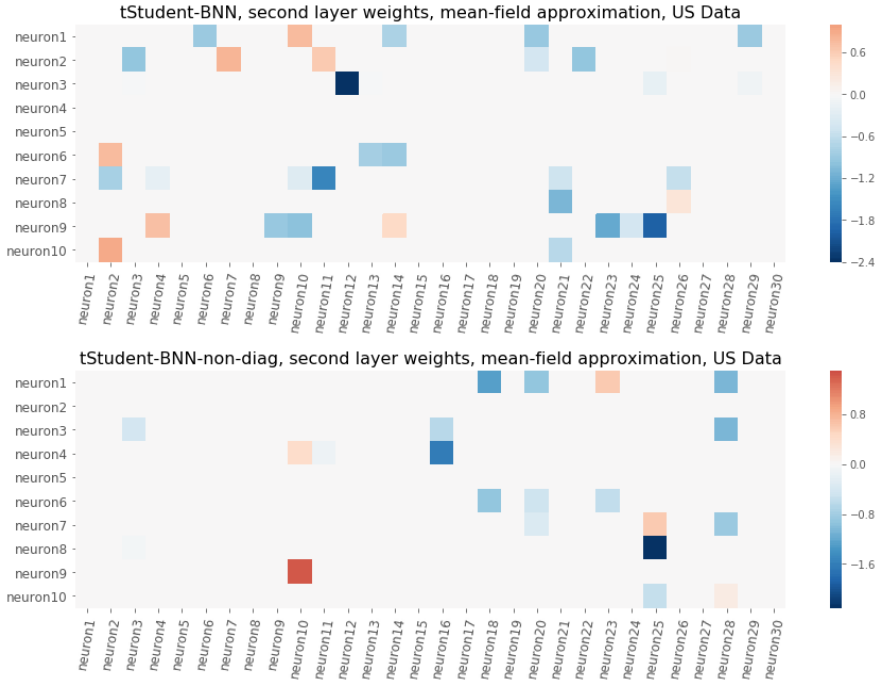

Figure 32. Estimation results for matrix W_2, mean-field , non-diagonal, US Data

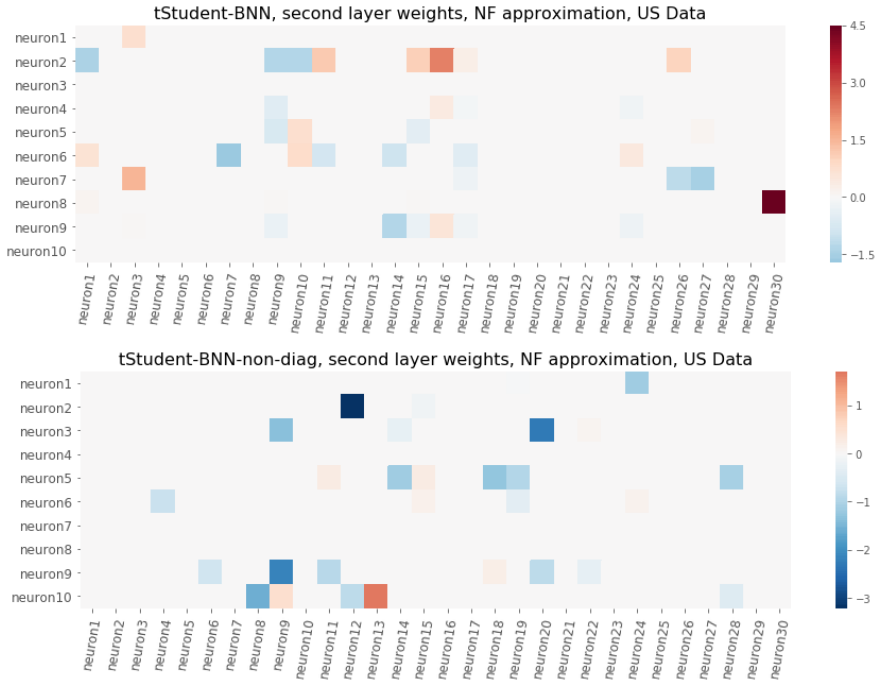

Figure 33. Estimation results for matrix W_2, NF , non-diagonal, US Data

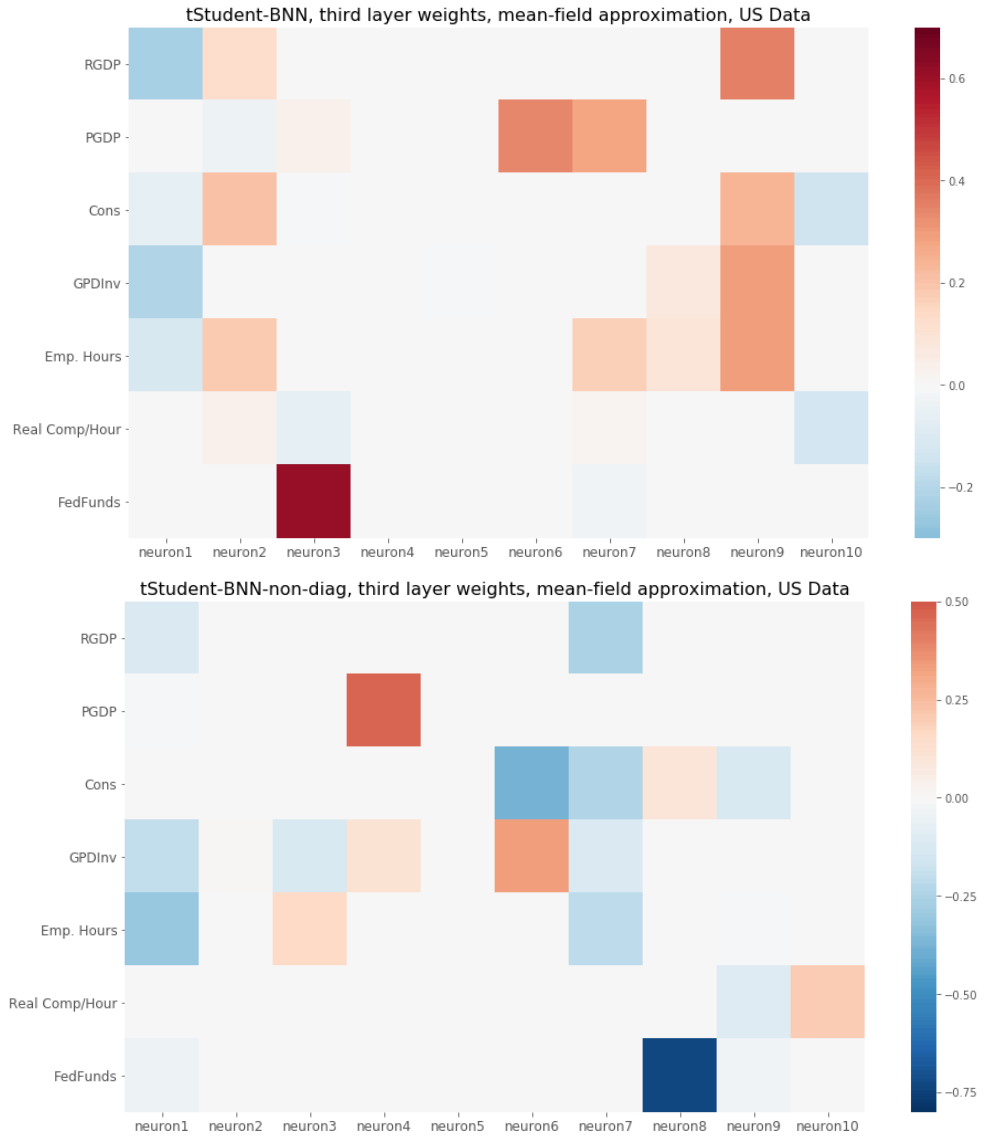

Figure 34. Estimation results for matrix W_3, mean-field , non-diagonal, US Data

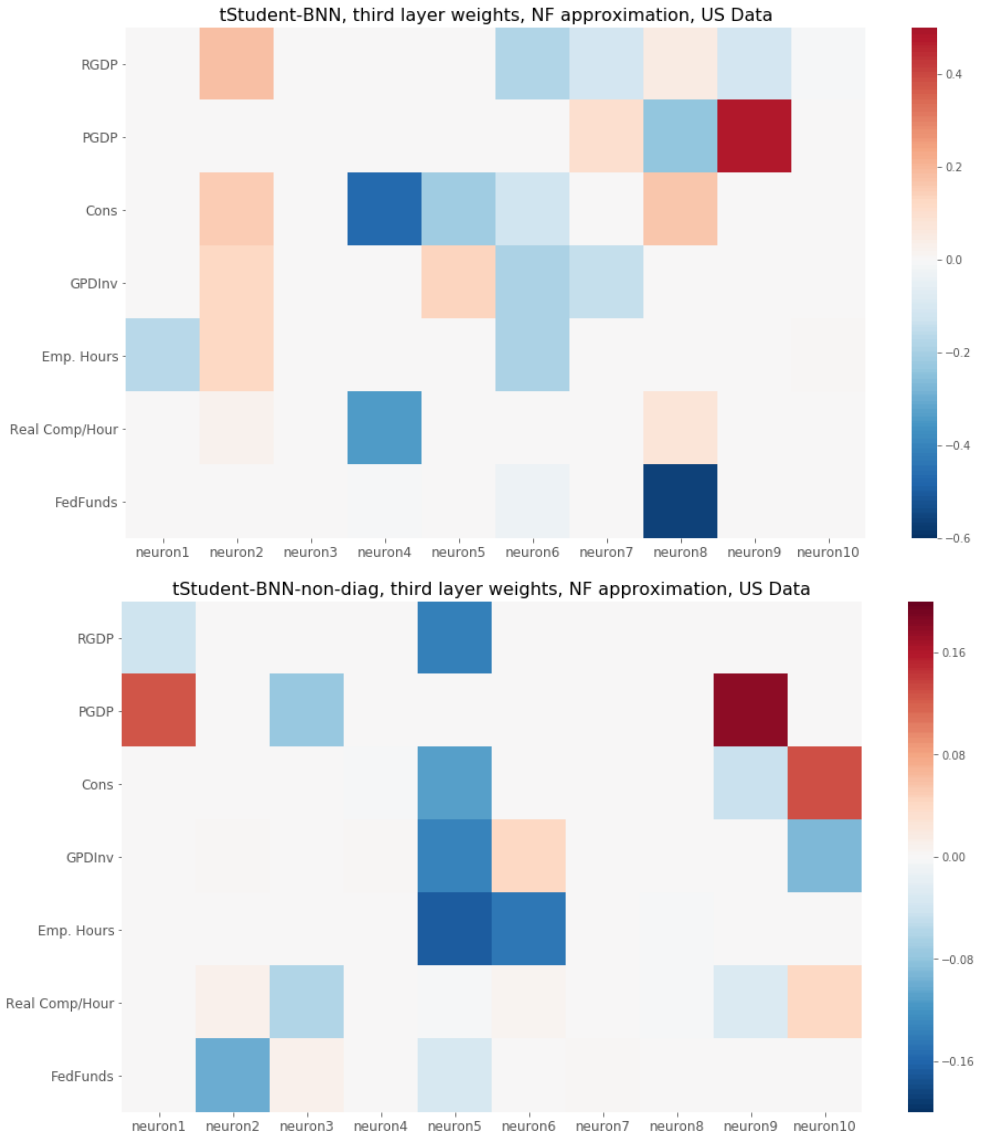

Figure 35. Estimation results for matrix W_3, NF , non-diagonal, US Data

Appendix B

The matrix was estimated in the main part of the paper for models with diagonal covariance. Here we show results for Bayesian vector autoregression with sparse priors and t-Student errors, and the Bayesian neural network with a non-diagonal covariance matrix.

Equations (14) and (19) do not restrict matrix C, but we faced computational problems calculating its log-determinant in Tensorflow. To avoid this problem, we set C to be triangular with ones in diagonal, so the log-determinant is zero. Estimation results for US Data are shown in Figures 29–35. Table 8 demonstrates that the non-diagonal covariance matrix significantly improves ELBO and marginal likelihood of the models.

	ELBO		Marginal likelihood	
	MF	NF	MF	NF
BVAR, US Data	-1374.5	-1362.9	-1364.1	-1352.7
BVAR, US Data, non-diag	-1131.1	-1122.3	-1118.1	-1114.4
BNN, US Data	-1240.2	-1236.9	-1203.1	-1192.1
BNN, US Data, non-diag	-1077.3	-1042.5	-1046.9	-1013.8

Table 8. ELBO and marginal likelihood, non-diagonal