Supplementary material: A Block Coordinate Descent Proximal Method for Simultaneous Filtering and Parameter Estimation

Ramin Raziperchikolaei [1 2] **Harish S. Bhat** [3 4]

Abstract

This supplementary material contains the following: 1) The equations and the true parameters of the ODEs that we used in the experiments, 2) Extension of our experiments with different types and magnitudes of the noise, and 3) An animation that shows how our method works and how the estimated and predicted states move closer to each other at each iteration.

1. ODEs in our experiments

We have used four benchmark datasets in our experiments. We only gave a brief explanation of each of them in the paper. Here, we introduce them in detail.

Lotka–Volterra model. This model is used to study the interaction between predator (variable x_0) and prey (variable x_1) in biology (Lotka, 1932). The model contains two nonlinear equations as follows:

$$\frac{dx_0}{dt} = \theta_0 x_0 - \theta_1 x_0 x_1 \qquad \frac{dx_1}{dt} = \theta_2 x_0 x_1 - \theta_3 x_1.$$

The state is two-dimensional and there are four unknown parameters. We use the same settings as in (Dondelinger et al., 2013). We set the parameters to $\theta_0 = 2$, $\theta_1 = 1$, $\theta_2 = 4$ and $\theta_3 = 1$. With initial condition $\mathbf{x}_{(1)} = [5, 3]$, we generate clean states in the time range of $[0, 2]$ with a spacing of $\Delta t = 0.1$.

FitzHugh–Nagumo model. This model describes spike generation in squid giant axons (FitzHugh, 1961; Nagumo et al., 1962). It has two nonlinear equations:

$$\frac{dx_0}{dt} = \theta_2(x_0 - \frac{(x_0)^3}{3} + x_1) \quad \frac{dx_1}{dt} = -\frac{1}{\theta_2}(x_0 - \theta_0 + \theta_1 x_1),$$

where x_0 is the voltage across an axon and x_1 is the outward current. The states are two-dimensional and there are three unknown parameters. We use the same settings as in (Ramsay et al., 2007). We set the parameters as $\theta_0 = 0.5, \theta_1 = 0.2$, and $\theta_3 = 3$. With initial condition $\mathbf{x}_{(1)} = [-1, 1]$, we generate clean states in the time range of $[0, 20]$ with a spacing of $\Delta t = 0.05$.

Rössler attractor. This three-dimensional nonlinear system has a chaotic attractor (Rössler, 1976):

$$\frac{dx_0}{dt} = -x_1 - x_2 \quad \frac{dx_1}{dt} = x_0 + \theta_0 x_1 \quad \frac{dx_2}{dt} = \theta_1 + x_2(x_0 - \theta_2).$$

The states are three-dimensional and there are three unknown parameters. We use the same settings as in (Ramsay et al., 2007). We set the parameters as $\theta_0 = 0.2, \theta_1 = 0.2$, and $\theta_3 = 3$. With the initial condition $\mathbf{x}_{(1)} = [1.13, -1.74, 0.02]$, we generate clean states in the time range of $[0, 20]$, with $\Delta t = 0.05$.

Lorenz-96 model. The goal of this model is to study weather predictability (Lorenz and Emanuel, 1998). The kth differential equation has the following form:

$$\frac{dx_k}{dt} = (x_{k+1} - x_{k-2})(x_{k-1}) - x_k + \theta_0,\ k = 0, \ldots, d-1$$

The model has one parameter θ_0 and d states, where d can be set by the user. This gives us the opportunity to test our method on larger ODEs. Note that to make this ODE meaningful, we have $x_{-1} = x_{d-1}$, $x_{-2} = x_{d-2}$, and $x_d = x_0$. As suggested in (Lorenz and Emanuel, 1998), we set $d = 40$ and $\theta_0 = 8$. The clean states are generated in the time range $[0, 4]$ with a spacing of $\Delta_i = 0.01$. The initial state is generated randomly from a Gaussian distribution with mean 0 and variance 1.

[1]Rakuten Institute of Technology, San Mateo, CA, USA [2]Department of Computer Science, University of California, Merced, USA [3]Department of Mathematics, University of Utah, USA [4]Department of Applied Mathematics, University of California, Merced, USA. Correspondence to: Ramin Raziperchikolaei <ramin.raziperchikola@rakuten.com>, Harish S. Bhat <hbhat@math.utah.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

2. Experimental results

Optimization of our objective function leads to a better estimation. In the Fig. 2 of our main paper, we reported the prediction error at each iteration of our algorithm for the Rössler and the Lorenz-96 models. Here, in Fig. 1, we add the FitzHugh–Nagumo and show the results for noisy observations with $\sigma^2 = 0.5$ and $\sigma^2 = 1$.

In all settings, our method decreases the error significantly for both Euler and three-step Adam-Bashforth methods. The three-step method performs better than the Euler method, specifically in the Lorenz-96 model.

Different types and amounts of noise in the observations. Our method does not assume anything about the type of noise. In reality, the noise could be from any distribution. In Fig. 2, we investigate the effect of the type of noise on the outcome of our algorithm. The red (blue) curves correspond to the case when we add Gaussian (Laplacian) noise to the observations. We set the mean to 0, change the variance of the noise, and report the prediction and parameter errors. Note that for each noise variance, we repeat the experiment 10 times and report the mean and standard deviation of the error.

In general, increasing the noise variance increases the error. We can see this in almost all plots. In both models, the error does not change much by changing the variance from 0 to 0.5. We can also see that the method performs almost as well for observations corrupted by Laplacian noise as in the Gaussian noise case. Note that the Laplacian noise has a heavier-than-Gaussian tail.

Comparison with other methods (robustness to the initialization). In Fig. (4) of the paper, we compared our method with three other methods in different categories on Rössler model. Fig 3 shows the comparison on the FitzHugh–Nagumo model. In both models, our method is robust with respect to the initialization and outperforms other methods significantly.

Comparison with the mean-field method (Gorbach et al., 2017). We compared our method with the mean-field method of (Gorbach et al., 2017) on the Lotka–Volterra model in Fig. (5) of the main paper, where the variance of the noise is $\sigma^2 = 1$. Fig. 4 compares the methods for $\sigma^2 = .5, 1$, and 1.5. Our method is more robust with respect to the noise and performs better.

Comparison with the extended Kalman filter (EKF). As we mentioned in the main paper, we follow (Sitz et al., 2002) in applying the Kalman filter to our problem of estimating the parameters and states. Here, we provide more information regarding our implementation.

We first need to write an equation that recursively finds the state $\mathbf{x}_{(t_{i+1})}$ in terms of $\mathbf{x}_{(t_i)}$. As suggested in (Sitz et al., 2002), this can be achieved by discretizing the ODE using the Euler discretization:

$$\mathbf{x}_{(t_{i+1})} = \mathbf{x}_{(t_i)} + \mathbf{f}(\mathbf{x}_{(t_i)}, \boldsymbol{\theta})\Delta_i. \tag{1}$$

Let us define $\boldsymbol{\theta}_{(t_i)}$ as the parameter estimated at time t_i by the Kalman filter. We define a joint state variable $\boldsymbol{\xi}_{(t_i)}$, which merges the states $\mathbf{x}_{(t_i)}$ and the parameters $\boldsymbol{\theta}_{(t_i)}$ as follows:

$$\boldsymbol{\xi}_{(t_i)} = \begin{pmatrix} \mathbf{x}_{(t_i)} \\ \boldsymbol{\theta}_{(t_i)} \end{pmatrix}, \quad \boldsymbol{\xi}_{(t_i)} \in \mathbb{R}^{d+p}. \tag{2}$$

The process model to predict the next state variable can be written as:

$$\boldsymbol{\xi}_{(t_{i+1})} = \begin{pmatrix} \mathbf{x}_{(t_{i+1})} \\ \boldsymbol{\theta}_{(t_{i+1})} \end{pmatrix} = \begin{pmatrix} \mathbf{x}_{(t_i)} + \mathbf{f}(\mathbf{x}_{(t_i)}, \boldsymbol{\theta})\Delta_i \\ \boldsymbol{\theta}_{(t_i)} \end{pmatrix}. \tag{3}$$

We define the observation model as follows:

$$\mathbf{y}_{(t_i)} = \mathbf{H}\boldsymbol{\xi}_{(t_i)}, \quad \mathbf{H} = \begin{pmatrix} \mathbf{I} & \mathbf{0} \end{pmatrix}_{d\times(d+p)}, \tag{4}$$

where $\mathbf{H}$ is a $d \times (d+p)$ matrix, $\mathbf{I}$ is a $d \times d$ identity matrix, and $\mathbf{0}$ is a $d \times p$ matrix where all elements are 0.

In most cases, the function $\mathbf{f}(\cdot)$ is nonlinear, which makes the process model nonlinear. For this reason, we use the extended Kalman filter (EKF), which linearizes the model.

We use an open-source Python code in (Labbe, 2014) to implement the EKF. We set the state covariance (noise covariance) to a diagonal matrix with elements equal to 1 000 (0.1). We set the process covariance using the function Q_discrete_white_noise() provided in (Labbe, 2014), where the variance is set to 1. Note that these parameters must be carefully tuned to obtain reasonable results; changing the state or noise covariance yields significantly worse results.

We compare our method with the EKF in Fig. (5) of the main paper on the Lotka–Volterra model. In that experiment, we set the number of samples to $T = 10\,000$ (time range $[0, 2]$). Here, we show the results for both $T = 20$ (time range $[0, 2]$) and $T = 10\,000$ (time range $[0, 1\,000]$).

As we can see in Fig. 5, the only setting in which the EKF performs comparably to our method is the case of $T = 10\,000$ and $\sigma^2 = 0.1$. In more realistic settings, our method significantly outperforms the EKF.

Animation to show how our method works. We consider the FitzHugh–Nagumo model, with settings as explained at beginning of this section, except that we consider the first 10 seconds instead of 20. We add Gaussian noise with variance 0.5 to the clean states to create the noisy observations. In Fig. 6, we show how our algorithm works in

the first 250 iterations. Acrobat Reader is required to play the animation. In this animation, $\mathbf{X}$ denotes clean states (green circles), $\mathbf{X}^*$ denotes estimated states, and $\hat{\mathbf{X}}$ denotes predicted states. Note that initially, $\mathbf{X}^*$ is the same as the noisy observations. Fig. 6 shows the two dimensions separately. At the top of each figure, we show the estimated parameters at each iteration. Note that the true parameters are $\theta_0 = 0.5, \theta_1 = 0.2$, and $\theta_2 = 3$. As explained before, the estimated and predicted states move closer to each other at each iteration. This helps the estimated parameters converge to the true parameters.

References

F. Dondelinger, D. Husmeier, S. Rogers, and M. Filippone. ODE parameter inference using adaptive gradient matching with Gaussian processes. In *Artificial Intelligence and Statistics*, pages 216–228, 2013.

R. FitzHugh. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical journal*, 1 (6):445–466, 1961.

N. S. Gorbach, S. Bauer, and J. M. Buhmann. Scalable variational inference for dynamical systems. In *Advances in Neural Information Processing Systems*, pages 4809–4818, 2017.

R. Labbe. Kalman and Bayesian filters in Python, 2014. URL https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python.

E. N. Lorenz and K. A. Emanuel. Optimal sites for supplementary weather observations: Simulation with a small model. *Journal of the Atmospheric Sciences*, 55(3):399–414, 1998.

A. J. Lotka. The growth of mixed populations: two species competing for a common food supply. *Journal of the Washington Academy of Sciences*, 22(16/17):461–469, 1932.

J. Nagumo, S. Arimoto, and S. Yoshizawa. An active pulse transmission line simulating nerve axon. *Proceedings of the IRE*, 50(10):2061–2070, 1962.

J. O. Ramsay, G. Hooker, D. Campbell, and J. Cao. Parameter estimation for differential equations: a generalized smoothing approach. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(5):741–796, 2007.

O. E. Rössler. An equation for continuous chaos. *Physics Letters A*, 57(5):397–398, 1976.

A. Sitz, U. Schwarz, J. Kurths, and H. U. Voss. Estimation of parameters and unobserved components for nonlinear systems from noisy time series. *Phys. Rev. E*, 66(1): 016210, 2002. doi: 10.1103/PhysRevE.66.016210.

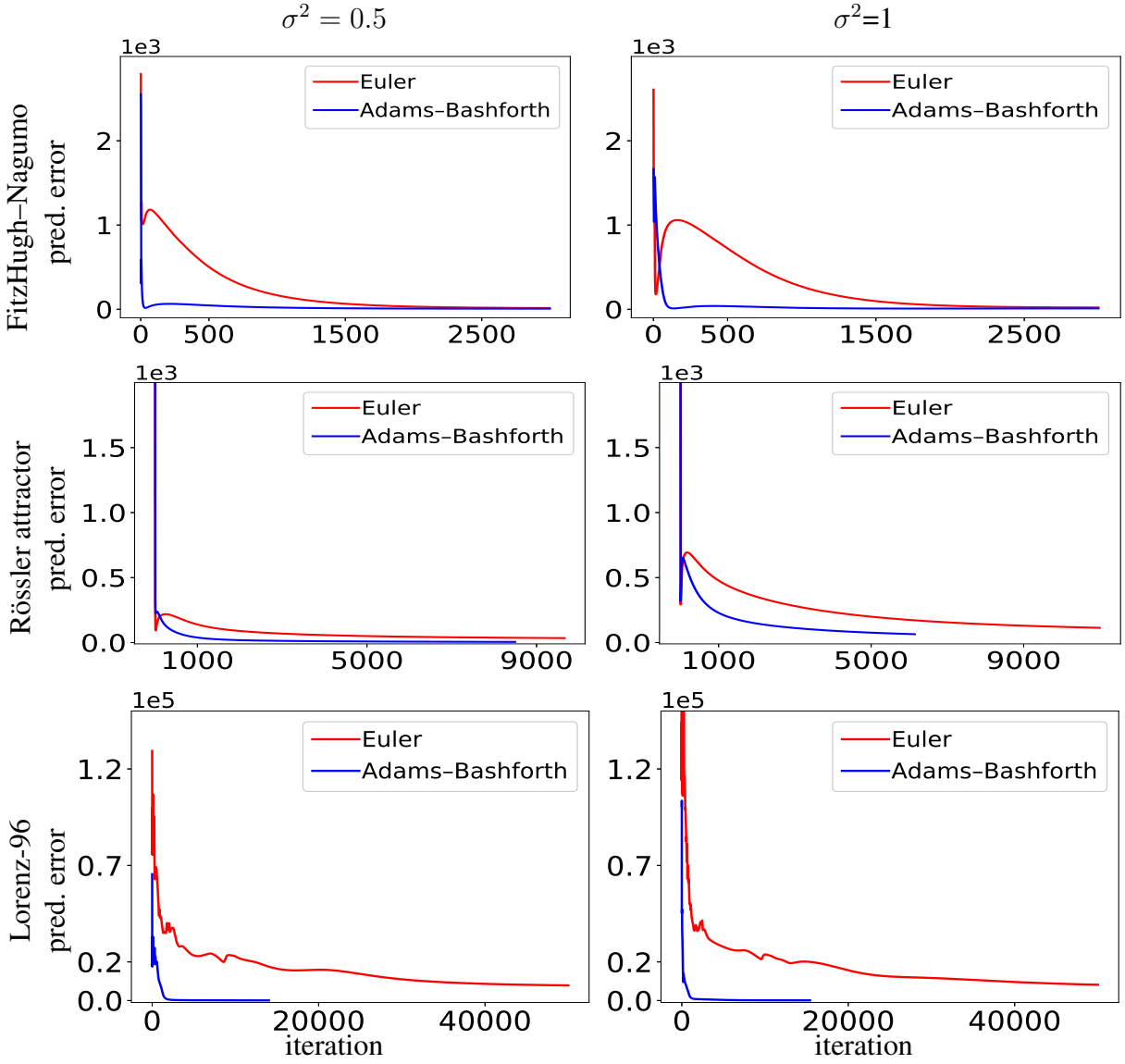


Figure 1. Similar to Fig. (2) of the main paper. We added FitzHugh–Nagumo and noisy observations with $\sigma^2 = 0.5$. Our learning strategy decreases the error in both cases and in all models.

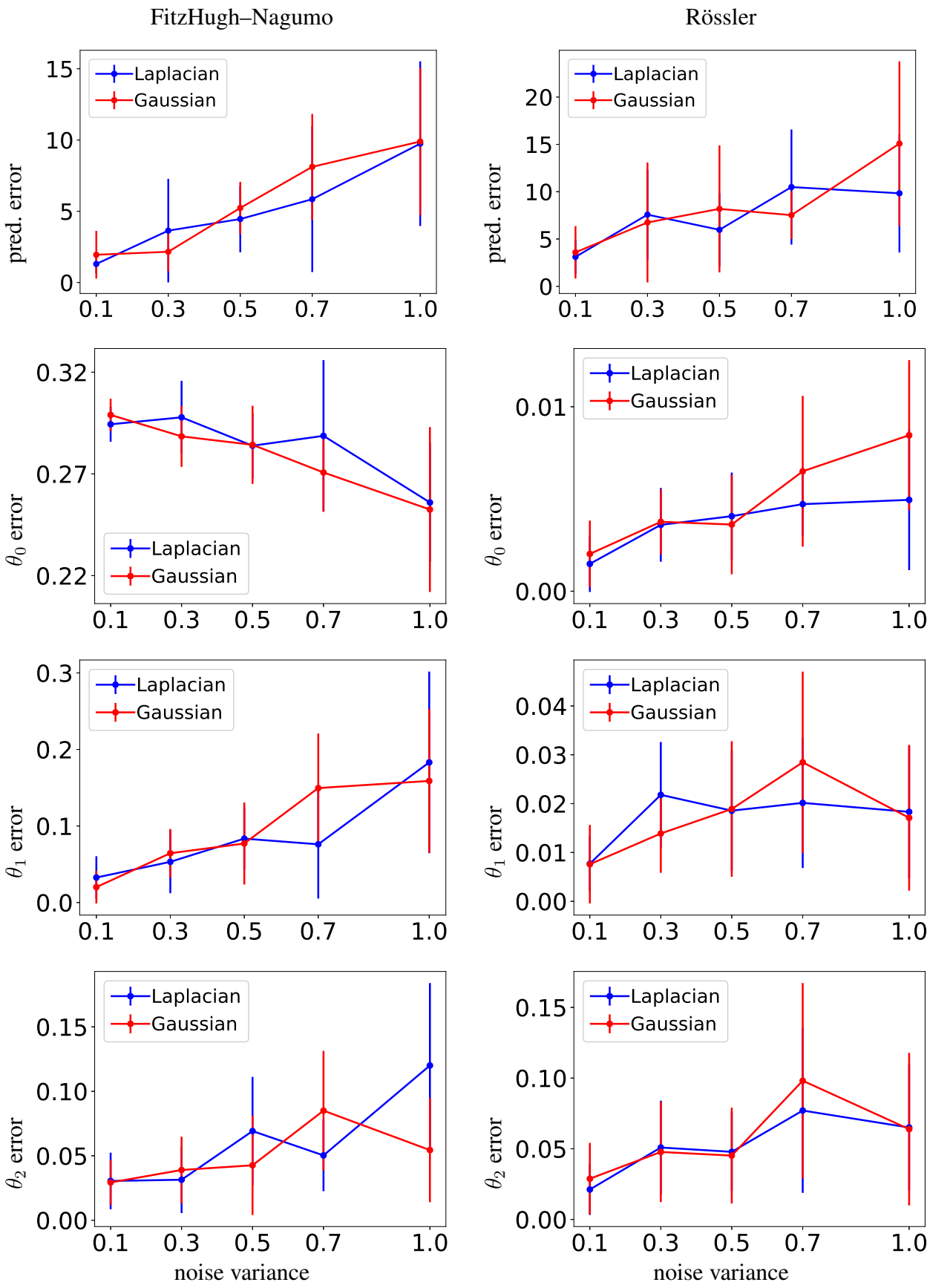


Figure 2. We change the amount and type of noise in the observations, and report the prediction and parameter errors on the FitzHugh–Nagumo (first column) and Rössler (second column) models.

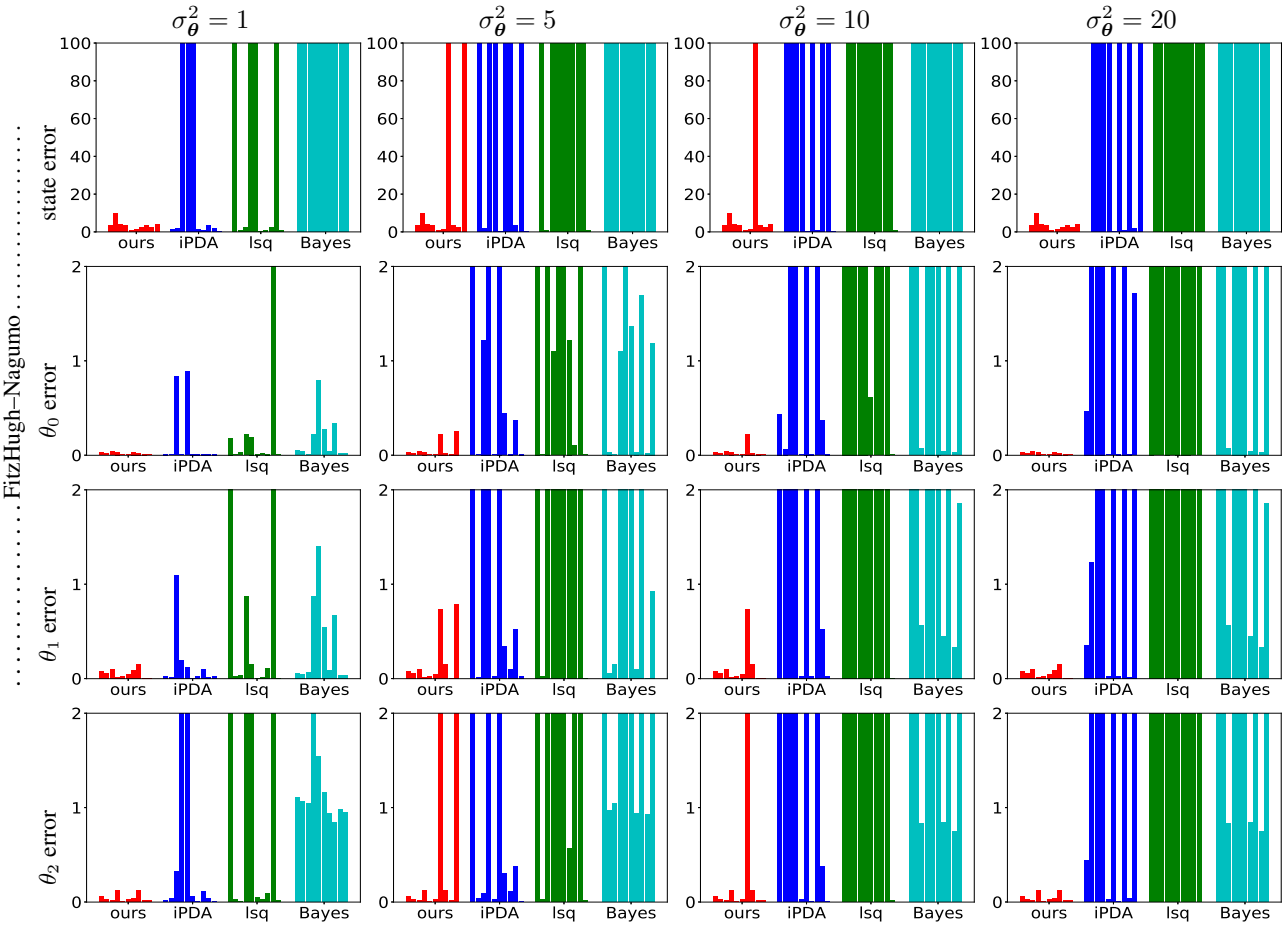


Figure 3. Similar to Fig. (4) of the paper, but on the FitzHugh–Nagumo model. Our method outperforms other methods significantly.

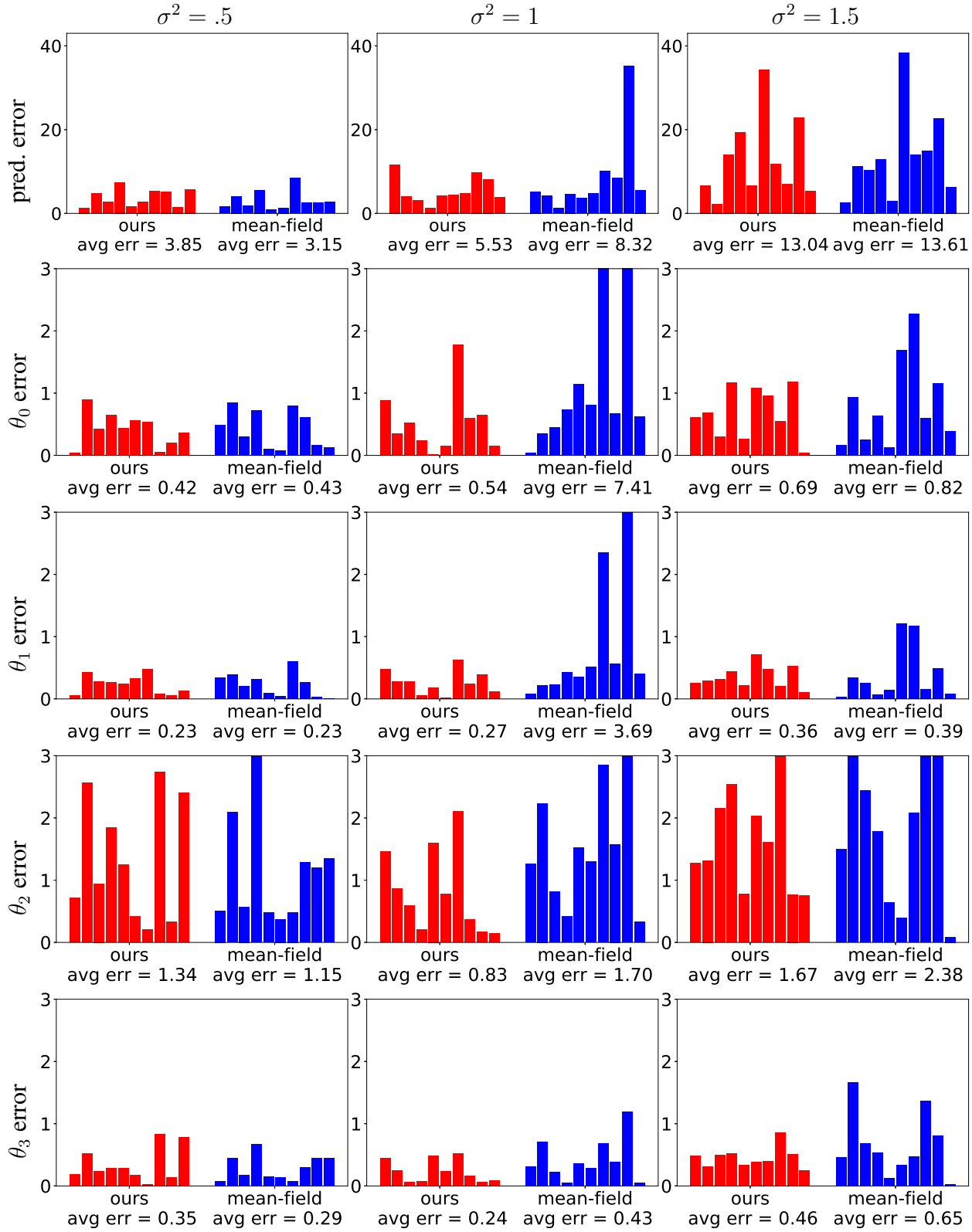


Figure 4. Comparison with the mean-field method. Similar to the first row of Fig. (5) in the paper, but for a set of noise variances: $\sigma^2 = .5, 1$, and 1.5. Our method is more robust with respect to the noise and performs better.

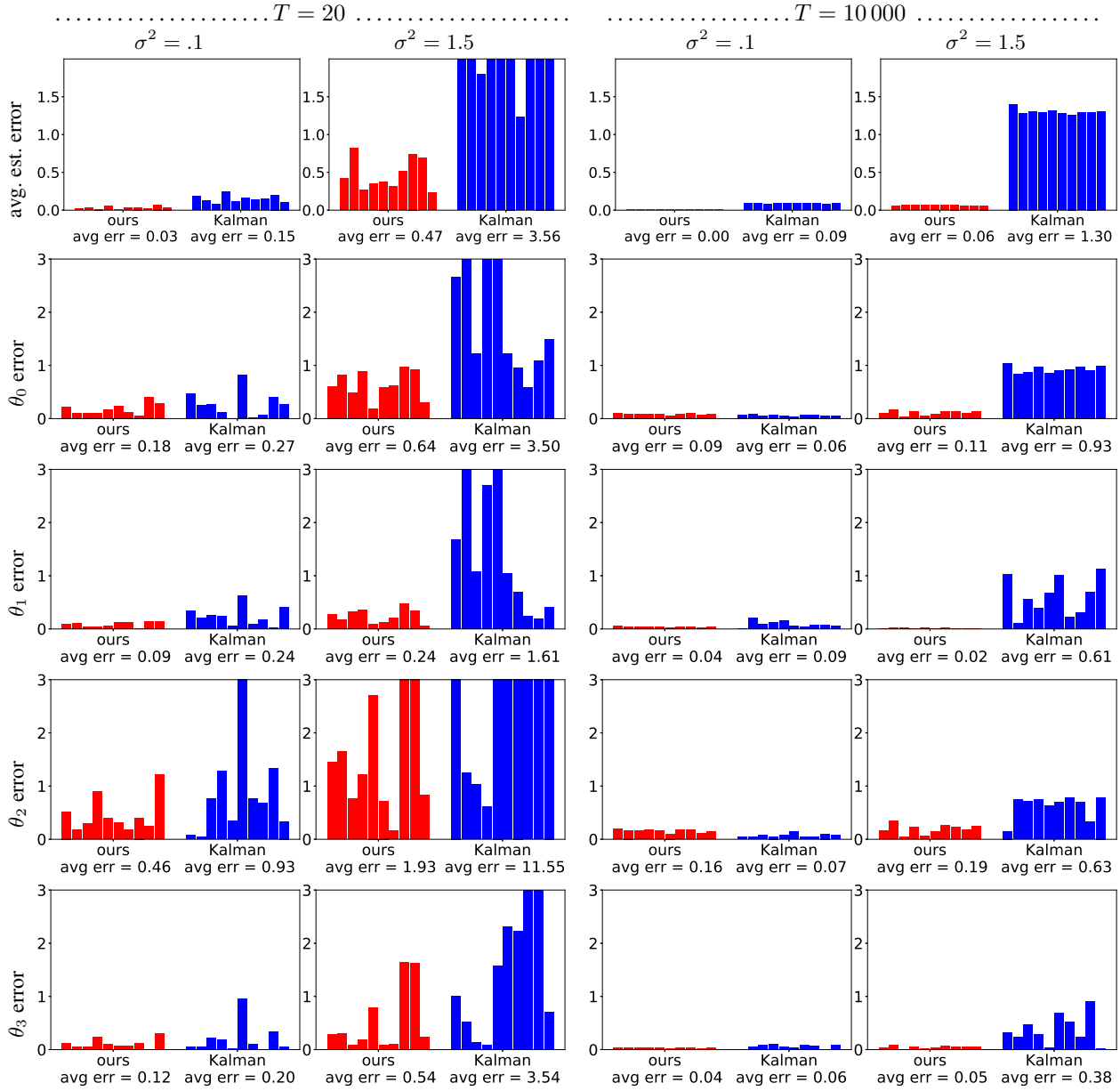
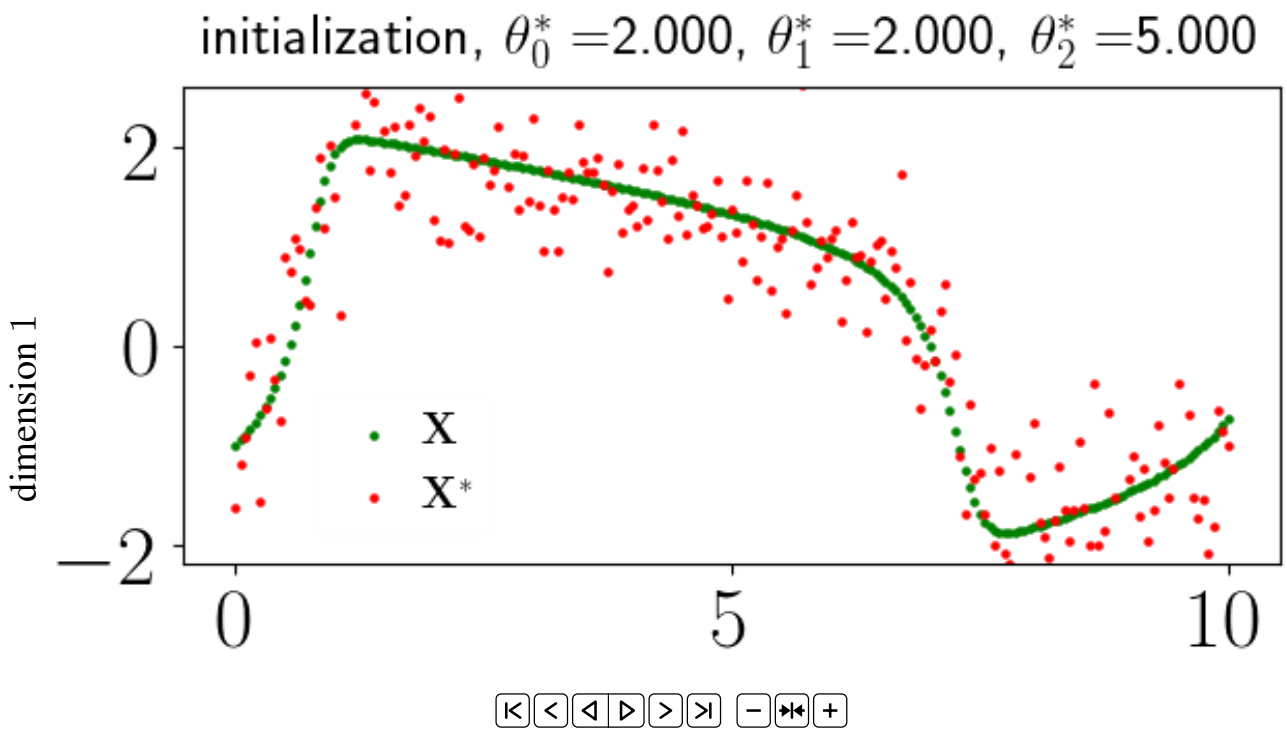


Figure 5. Comparison with the EKF. Similar to the second and third rows of Fig. (5) of the paper, but includes both $T = 20$ and $T = 10\,000$ observations.



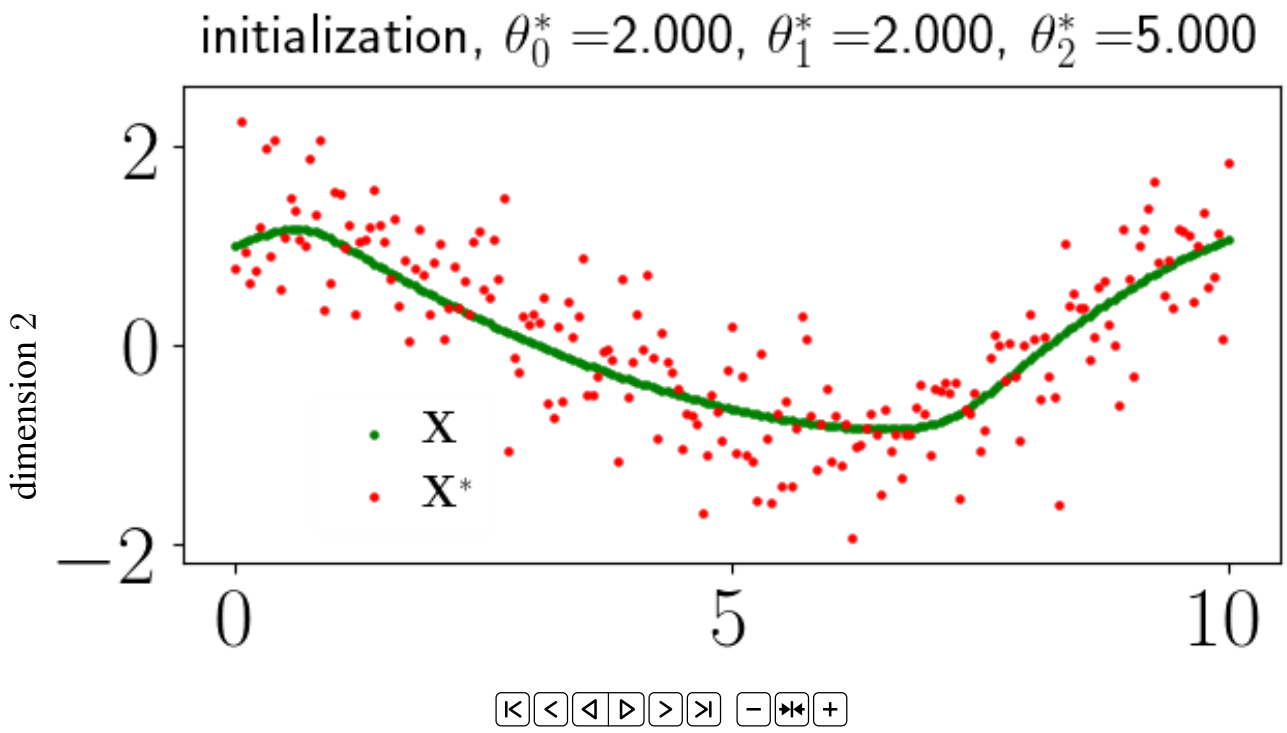


Figure 6. FitzHugh–Nagumo model, where the true parameters are $\theta_0 = 0.5, \theta_1 = 0.2$, and $\theta_2 = 3$. Noisy observations are achieved by adding Gaussian noise to the clean states. In this figure, $\mathbf{X}$ is the clean states (green circles), $\mathbf{X}^*$ is the estimated states, and $\hat{\mathbf{X}}$ is the predicted states. Note that at the initialization, $\mathbf{X}^*$ is the same as the noisy observations. The estimated parameters at each iteration of our algorithm are shown at the top of the figure.