Scalable Variational Inference for Stochastic Differential Equations

Master Thesis
Ruifeng Xu
Supervisor: Prof. Dr. Joachim M. Buhmann
Advisors: Stefan Bauer & Nico S. Gorbach
Department of Computer Science, ETH Zurich

Statistical inference of states and parameters of dynamical systems based on noisy, sparse or even incomplete state observations.

Outline

- Dynamical systems
- Motivation & challenges
- Laplace mean-field approximation
- Extension to random dynamical systems
- Experiments
- Conclusion

Dynamical Systems

Ordinary differential equations (ODEs)

A K-dimensional real-valued ODE system is defined as

$$\dot{\boldsymbol{x}}(t) = \frac{d\boldsymbol{x}(t)}{dt} = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{\theta}) \tag{1}$$

where

$\boldsymbol{x}(t) = [x_1(t), \ldots, x_K(t)]^\top \in \mathbb{R}^K$ are the states at time t,

$\dot{\boldsymbol{x}}(t) = [f_1(\boldsymbol{x}(t), \boldsymbol{\theta}), \ldots, f_K(\boldsymbol{x}(t), \boldsymbol{\theta})]^\top \in \mathbb{R}^K$ are the state derivatives at time t,

$\boldsymbol{f} : \mathbb{R}^K \mapsto \mathbb{R}^K$ is the vector fields with parameter $\boldsymbol{\theta} \in \mathbb{R}^M$.

Initial states and parameters determine the future states.

$\boldsymbol{f}$ may have direct dependency on t, which is suppressed for uncluttered notations.

Stochastic differential equations (SDEs)

Given a probability space $(\mathbf{\Omega}, \mathcal{F}, \mathbb{P})$, a K-dimensional SDE system with state-specific, additive Gaussian noises is defined, in the *Itô* form, as

$$d\boldsymbol{x}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{\theta})dt + \mathbf{\Sigma}^{\frac{1}{2}} d\boldsymbol{W}_t \tag{2}$$

where

- $\boldsymbol{f} : \mathbb{R}^K \mapsto \mathbb{R}^K$ is the deterministic drift function with parameter $\boldsymbol{\theta} \in \mathbb{R}^M$
- $\mathbf{\Sigma} = diag(\rho_1^2, \ldots, \rho_K^2) \in \mathbb{R}^{K \times K}$ is the diagonal noise covariance matrix,
- $\boldsymbol{W}_t \in \mathbb{R}^K$ is a standard K-dimensional Wiener process.

Each realization is most likely a different *sample path*.

A class of multiplicative noise models can be mapped to this model.

Motivation & Challenges

Motivation

Dynamical systems model various natural phenomena in chemistry, physics, biology, economics, meteorology, etc. For example

- *Protein signalling transduction pathway* models the dynamics among protein species using a set of non-linear differential equations.
- Stochastic *Lorenz 96* model is commonly used in weather forecast.
- Many others . . .

The inference algorithm should be accurate, robust and performant.

Challenges

- Conventional methods requires explicit numerical integrations each time after parameter adaptation, which is slow and not scalable.
- The likelihood surfaces are likely to be multimodal due to nonlinearity within the dynamical systems, which makes parameter search difficult.
- In Bayesian statistics, the marginalization term is intractable and requires approximate inference techniques.

  Markov chain Monte Carlo (MCMC) sampling schemes are accurate but computationally expensive and requires onerous convergence analysis.

Laplace Mean-Field Approximation

Noisy observation

Usually, observations $\boldsymbol{y}(t) = [y_1(t), \ldots, y_K(t)]^\top \in \mathbb{R}^K$ are contaminated by noises $\boldsymbol{\varepsilon}(t) = [\varepsilon_1(t), \ldots, \varepsilon_K(t)] \in \mathbb{R}^K$ such that

$$\boldsymbol{y}(t) = \boldsymbol{x}(t) + \boldsymbol{\varepsilon}(t) \tag{3}$$

For a sequence of observations, we denote

$$\mathrm{Y} = [\boldsymbol{y}(t_1), \ldots, \boldsymbol{y}(t_N)] \in \mathbb{R}^{K \times N}$$
$$\mathrm{X} = [\boldsymbol{x}(t_1), \ldots, \boldsymbol{x}(t_N)] \in \mathbb{R}^{K \times N}$$
$$\mathrm{E} = [\boldsymbol{\varepsilon}(t_1), \ldots, \boldsymbol{\varepsilon}(t_N)] \in \mathbb{R}^{K \times N}$$

Noisy observation

Assuming *independent and identically distributed (i.i.d.)* state-specific, additive Gaussian noise $\varepsilon(t) \sim \mathcal{N}(\mathbf{0}, \boldsymbol{D})$ with $\boldsymbol{D} = diag(\sigma_1^2, \ldots, \sigma_K^2) \in R^{K \times K}$, then

$$\begin{aligned} p(\mathrm{Y}|\mathrm{X}, \boldsymbol{\sigma}) &= \prod_k p(\boldsymbol{y}_k|\boldsymbol{x}_k, \sigma_k) \\ &= \prod_k \mathcal{N}(\boldsymbol{y}_k|\boldsymbol{x}_k, \sigma_k^2 \boldsymbol{I}) \end{aligned} \tag{4}$$

where

$\boldsymbol{y}_k = [y_k(t_1), \ldots, y_k(t_N)]^\top \in \mathbb{R}^N$ are the observations for the k-th state over time.

$\boldsymbol{x}_k = [x_k(t_1), \ldots, x_k(t_N)]^\top \in \mathbb{R}^N$ are the values of the k-th state over time.

State prior

Introducing state-specific, independent Gaussian process priors on each $\boldsymbol{x}_k$ for $k = 1, \ldots, K$, then

$$\begin{aligned} p(\mathrm{X}|\boldsymbol{\varphi}) &= \prod_k p(\boldsymbol{x}_k|\boldsymbol{\varphi}_k) \\ &= \prod_k \mathcal{N}(\boldsymbol{x}_k|\mathbf{0}, \boldsymbol{C}_{\boldsymbol{\varphi}_k}) \end{aligned} \tag{5}$$

where $\boldsymbol{C}_{\boldsymbol{\varphi}_k}$ is the covariance matrix induced by the kernel function $\mathcal{K}_{\boldsymbol{\varphi}_k}$ with hyperparemeter $\boldsymbol{\varphi}_k$.

State posterior

Using *Bayes' theorem*, the posterior on X is obtained as

$$\begin{aligned} p(\mathrm{X}|\mathrm{Y}, \boldsymbol{\varphi}, \boldsymbol{\sigma}) &= \frac{p(\mathrm{X}|\boldsymbol{\varphi}) p(\mathrm{Y}|\mathrm{X}, \boldsymbol{\sigma})}{\int p(\mathrm{X}|\boldsymbol{\varphi}) p(\mathrm{Y}|\mathrm{X}, \boldsymbol{\sigma}) d\mathrm{X}} \\ &= \prod_k p(\boldsymbol{x}_k|\boldsymbol{y}_k, \boldsymbol{\varphi}_k, \sigma_k) \\ &= \prod_k \mathcal{N}(\boldsymbol{x}_k|\boldsymbol{\mu}_k(\boldsymbol{y}_k), \boldsymbol{\Sigma}_k) \end{aligned} \tag{6}$$

where

$$\begin{aligned} \boldsymbol{\mu}_k(\boldsymbol{y}_k) &= \boldsymbol{C}_{\boldsymbol{\varphi}_k} (\boldsymbol{C}_{\boldsymbol{\varphi}_k} + \sigma_k^2 \boldsymbol{I})^{-1} \boldsymbol{y}_k \\ \boldsymbol{\Sigma}_k &= \sigma_k^2 \boldsymbol{C}_{\boldsymbol{\varphi}_k} (\boldsymbol{C}_{\boldsymbol{\varphi}_k} + \sigma_k^2 \boldsymbol{I})^{-1} \end{aligned}$$

Gaussian process response model

Because Gaussian process is closed under differentiation, the joint distribution of $\boldsymbol{x}_k$ and $\dot{\boldsymbol{x}}_k$, for $k = 1, \ldots, K$, within a finite amount of time points is also Gaussian:

$$\begin{bmatrix} \boldsymbol{x}_k \\ \dot{\boldsymbol{x}}_k \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbf{0} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} \boldsymbol{C}_{\boldsymbol{\varphi}_k} & \boldsymbol{C}'_{\boldsymbol{\varphi}_k} \\ {}'\boldsymbol{C}_{\boldsymbol{\varphi}_k} & \boldsymbol{C}''_{\boldsymbol{\varphi}_k} \end{bmatrix} \right) \tag{7}$$

where

$$C_{\boldsymbol{\varphi}_k i,j} = \mathcal{K}_{\boldsymbol{\varphi}_k}(t_i, t_j) \qquad C'_{\boldsymbol{\varphi}_k i,j} = \frac{\partial \mathcal{K}_{\boldsymbol{\varphi}_k}(t_i, t_j)}{\partial t_j}$$

$${}'C_{\boldsymbol{\varphi}_k i,j} = \frac{\partial \mathcal{K}_{\boldsymbol{\varphi}_k}(t_i, t_j)}{\partial t_i} \qquad C''_{\boldsymbol{\varphi}_k i,j} = \frac{\partial^2 \mathcal{K}_{\boldsymbol{\varphi}_k}(t_i, t_j)}{\partial t_i \partial t_j}$$

Gaussian process response model

The conditional distribution over $\dot{\mathrm{X}}$ is given by

$$\begin{aligned} p(\dot{\mathrm{X}}|\mathrm{X}, \boldsymbol{\varphi}) &= \prod_k p(\dot{\boldsymbol{x}}_k|\boldsymbol{x}_k, \boldsymbol{\varphi}_k) \\ &= \prod_k \mathcal{N}(\dot{\boldsymbol{x}}_k|\boldsymbol{m}_k, \boldsymbol{A}_k) \end{aligned} \tag{8}$$

where

$$\begin{aligned} \boldsymbol{m}_k &= {}'\boldsymbol{C}_{\boldsymbol{\varphi}_k} \boldsymbol{C}_{\boldsymbol{\varphi}_k}^{-1} \boldsymbol{x}_k \\ \boldsymbol{A}_k &= \boldsymbol{C}''_{\boldsymbol{\varphi}_k} - {}'\boldsymbol{C}_{\boldsymbol{\varphi}_k} \boldsymbol{C}_{\boldsymbol{\varphi}_k}^{-1} \boldsymbol{C}'_{\boldsymbol{\varphi}_k} \end{aligned}$$

ODE response model

Assuming state-specific, additive Gaussian errors between $\dot{\boldsymbol{x}}(t)$ and the response from $\boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{\theta})$, we have

$$\begin{aligned} p(\dot{\mathrm{X}}|\mathrm{X}, \boldsymbol{\theta}, \boldsymbol{\gamma}) &= \prod_k p(\dot{\boldsymbol{x}}_k|\mathrm{X}, \boldsymbol{\theta}, \gamma_k) \\ &= \prod_k \mathcal{N}(\dot{\boldsymbol{x}}_k|\boldsymbol{f}_k(\mathrm{X}, \boldsymbol{\theta}), \gamma_k \boldsymbol{I}) \end{aligned} \tag{9}$$

where

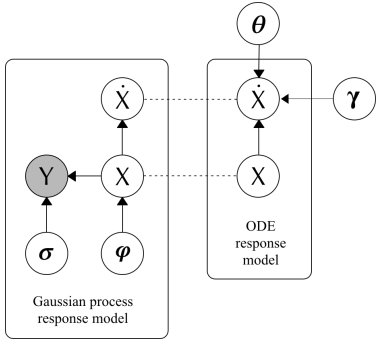
- $\dot{\boldsymbol{x}}_k = [\dot{x}_k(t_1), \ldots, \dot{x}_k(t_N)] \in \mathbb{R}^N$ are the derivatives for the k-th state over time.
- $\boldsymbol{\gamma} = [\gamma_1, \ldots, \gamma_K]^T \in \mathbb{R}^K$ contains the error variances.

Product of experts

The *product of experts* technique combines (Eq. 8) and (Eq. 9) to obtain

$$p(\dot{\mathrm{X}}|\mathrm{X}, \boldsymbol{\varphi}, \boldsymbol{\theta}, \boldsymbol{\gamma}) \propto p(\dot{\mathrm{X}}|\mathrm{X}, \boldsymbol{\varphi}) p(\dot{\mathrm{X}}|\mathrm{X}, \boldsymbol{\theta}, \boldsymbol{\gamma}) \qquad (10)$$

which attains high densities where both $p(\dot{\mathrm{X}}|\mathrm{X}, \boldsymbol{\varphi})$ and $p(\dot{\mathrm{X}}|\mathrm{X}, \boldsymbol{\theta}, \boldsymbol{\gamma})$ have strong support.



Joint posterior

The joint posterior $p(\mathrm{X}, \boldsymbol{\theta}|\mathrm{Y}, \boldsymbol{\varphi}, \boldsymbol{\sigma}, \boldsymbol{\gamma})$ is obtained by

$$\begin{aligned} p(\mathrm{X}, \boldsymbol{\theta}|\mathrm{Y}, \boldsymbol{\varphi}, \boldsymbol{\sigma}, \boldsymbol{\gamma}) &= \int p(\boldsymbol{\theta}) p(\mathrm{X}|\mathrm{Y}, \boldsymbol{\varphi}, \boldsymbol{\sigma}) p(\dot{\mathrm{X}}|\mathrm{X}, \boldsymbol{\theta}, \boldsymbol{\varphi}, \boldsymbol{\gamma}) d\dot{\mathrm{X}} \\ &\propto p(\boldsymbol{\theta}) \prod_k [\mathcal{N}(\boldsymbol{x}_k|\boldsymbol{\mu}_k(\boldsymbol{y}_k), \boldsymbol{\Sigma}_k) \mathcal{N}(\boldsymbol{f}_k(\mathrm{X}, \boldsymbol{\theta})|\boldsymbol{m}_k, \boldsymbol{\Lambda}_k^{-1})] \end{aligned} \tag{11}$$

where

$$\boldsymbol{\Lambda}_k^{-1} = \boldsymbol{A}_k + \gamma_k \boldsymbol{I}$$

The “best” parameters $\boldsymbol{\theta}^*$ could be estimated using *Maximum a posteriori (MAP)* to yield

$$\begin{aligned} \boldsymbol{\theta}^* &= \arg\max_{\boldsymbol{\theta}} \int p(\mathrm{X}, \boldsymbol{\theta}|\mathrm{Y}, \boldsymbol{\varphi}, \boldsymbol{\sigma}, \boldsymbol{\gamma}) d\mathrm{X} \\ &= \arg\max_{\boldsymbol{\theta}} p(\boldsymbol{\theta}|\mathrm{Y}, \boldsymbol{\varphi}, \boldsymbol{\sigma}, \boldsymbol{\gamma}) \end{aligned} \tag{12}$$

which is intractable due to strong non-linear couplings of the states inside the ODEs.

Laplace mean-field approximation

Positing the following factorized proxy distribution:

$$
\begin{aligned}
Q(\mathrm{X}, \boldsymbol{\theta}) &= q(\boldsymbol{\theta}|\boldsymbol{\eta}_{\boldsymbol{\theta}}, \boldsymbol{\Xi}_{\boldsymbol{\theta}}) \prod_u q(\boldsymbol{x}_u|\boldsymbol{\eta}_{\boldsymbol{x}_u}, \boldsymbol{\Xi}_{\boldsymbol{x}_u}) \\
&= \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\eta}_{\boldsymbol{\theta}}, \boldsymbol{\Xi}_{\boldsymbol{\theta}}) \prod_u \mathcal{N}(\boldsymbol{x}_u|\boldsymbol{\eta}_{\boldsymbol{x}_u}, \boldsymbol{\Xi}_{\boldsymbol{x}_u}) \qquad (13)
\end{aligned}
$$

Conditional probability $p(\boldsymbol{x}_u|\mathrm{Y}, \mathrm{X}_{/\{u\}}, \boldsymbol{\varphi}, \boldsymbol{\theta}, \boldsymbol{\sigma}, \boldsymbol{\gamma})$

Denoting $\mathrm{X}_{/\{u\}} = \{\boldsymbol{x}_o | o = 1, \ldots, K \text{ and } o \neq u\}$, for $u = 1, \ldots, K$, we have

$$
\begin{aligned}
p(\boldsymbol{x}_u|\mathrm{Y}, \mathrm{X}_{/\{u\}}, \boldsymbol{\varphi}, \boldsymbol{\theta}, \boldsymbol{\sigma}, \boldsymbol{\gamma}) &= \int p(\boldsymbol{x}_u|\mathrm{Y}, \mathrm{X}_{/\{u\}}, \boldsymbol{\varphi}, \boldsymbol{\sigma}) p(\dot{\mathrm{X}}|\boldsymbol{x}_u, \mathrm{X}_{/\{u\}}, \boldsymbol{\varphi}, \boldsymbol{\theta}, \boldsymbol{\gamma}) d\dot{\mathrm{X}} \\
&\overset{(b)}{=} \int p(\boldsymbol{x}_u|\boldsymbol{y}_u, \boldsymbol{\varphi}_k, \sigma_k) p(\dot{\mathrm{X}}|\mathrm{X}, \boldsymbol{\varphi}, \boldsymbol{\theta}, \boldsymbol{\gamma}) d\dot{\mathrm{X}} \\
&\propto \mathcal{N}(\boldsymbol{x}_u|\boldsymbol{\mu}_u(\boldsymbol{y}_u), \boldsymbol{\Sigma}_u) \prod_k \mathcal{N}(\boldsymbol{f}_k(\mathrm{X}, \boldsymbol{\theta})|\boldsymbol{m}_k, \boldsymbol{\Lambda}_k^{-1}) \qquad (14)
\end{aligned}
$$

where (b) holds because

- $p(\boldsymbol{x}_u|\mathrm{Y}, \mathrm{X}_{/\{u\}}, \boldsymbol{\varphi}, \boldsymbol{\sigma})$ depends only on $\boldsymbol{y}_u$ due to independent prior assumption,
- $p(\dot{\mathrm{X}}|\boldsymbol{x}_u, \mathrm{X}_{/\{u\}}, \boldsymbol{\varphi}, \boldsymbol{\theta}, \boldsymbol{\gamma})$ is equivalent to $p(\dot{\mathrm{X}}|\mathrm{X}, \boldsymbol{\varphi}, \boldsymbol{\theta}, \boldsymbol{\gamma})$.

Cost minimization for states

The mean vector and precision matrix of $q(\boldsymbol{x}_u|\boldsymbol{\eta}_{\boldsymbol{x}_u}, \boldsymbol{\Xi}_{\boldsymbol{x}_u})$ for $u = 1, \ldots, K$ are given by

$$\begin{aligned}
\boldsymbol{\eta}_{\boldsymbol{x}_u} &= \arg\max_{\boldsymbol{x}_u} \ln \left[\mathcal{N}(\boldsymbol{x}_u|\boldsymbol{\mu}_u(\boldsymbol{y}_u), \boldsymbol{\Sigma}_u) \prod_k \mathcal{N}(\boldsymbol{f}_k(\mathrm{X}, \boldsymbol{\theta})|\boldsymbol{m}_k, \boldsymbol{\Lambda}_k^{-1})\right] \\
&= \arg\max_{\boldsymbol{x}_u} \left[\ln \mathcal{N}(\boldsymbol{x}_u|\boldsymbol{\mu}_u(\boldsymbol{y}_u), \boldsymbol{\Sigma}_u) + \sum_k \ln \mathcal{N}(\boldsymbol{f}_k(\mathrm{X}, \boldsymbol{\theta})|\boldsymbol{m}_k, \boldsymbol{\Lambda}_k^{-1})\right] \\
&= \arg\min_{\boldsymbol{x}_u} \frac{1}{2}[(\boldsymbol{x}_u - \boldsymbol{\mu}_u(\boldsymbol{y}_u))^T \boldsymbol{\Sigma}_u^{-1}(\boldsymbol{x}_u - \boldsymbol{\mu}_u(\boldsymbol{y}_u)) + \sum_k (\boldsymbol{f}_k - \boldsymbol{m}_k)^T \boldsymbol{\Lambda}_k (\boldsymbol{f}_k - \boldsymbol{m}_k)] \\
&= \arg\min_{\boldsymbol{x}_u} cost_{\boldsymbol{x}_u}(\boldsymbol{x}_u, \mathrm{X}_{/\{u\}}, \boldsymbol{\theta}, \boldsymbol{\mu}_u(\boldsymbol{y}_u), \boldsymbol{\Sigma}_u, \boldsymbol{m}, \boldsymbol{\Lambda}) \qquad (15) \\
\boldsymbol{\Xi}_{\boldsymbol{x}_u}^{-1} &= \nabla\nabla cost_{\boldsymbol{x}_u}|_{\boldsymbol{x}_u = \boldsymbol{\eta}_{\boldsymbol{x}_u}} \qquad (16)
\end{aligned}$$

Conditional probability $p(\boldsymbol{\theta}|\mathrm{Y}, \mathrm{X}, \boldsymbol{\varphi}, \boldsymbol{\gamma})$

For $p(\boldsymbol{\theta}|\mathrm{Y}, \mathrm{X}, \boldsymbol{\varphi}, \boldsymbol{\gamma})$, we have

$$\begin{aligned} p(\boldsymbol{\theta}|\mathrm{Y}, \mathrm{X}, \boldsymbol{\varphi}, \boldsymbol{\gamma}) &\overset{(a)}{=} p(\boldsymbol{\theta}|\mathrm{X}, \boldsymbol{\varphi}, \boldsymbol{\gamma}) \\ &= \int p(\boldsymbol{\theta}) p(\dot{\mathrm{X}}|\mathrm{X}, \boldsymbol{\varphi}, \boldsymbol{\theta}, \boldsymbol{\gamma}) d\dot{\mathrm{X}} \\ &\propto \prod_k \mathcal{N}(\boldsymbol{f}_k(\mathrm{X}, \boldsymbol{\theta})|\boldsymbol{m}_k, \boldsymbol{\Lambda}_k^{-1}) \end{aligned} \tag{17}$$

where (a) holds since $\boldsymbol{\theta}$ depends indirectly on Y through X.

Cost minimization for parameters

Denoting $\boldsymbol{f}_k = \boldsymbol{f}_k(\mathrm{X}, \boldsymbol{\theta})$, $\boldsymbol{m} = [\boldsymbol{m}_1, \ldots, \boldsymbol{m}_K]$, and $\boldsymbol{\Lambda} = [\boldsymbol{\Lambda}_1, \ldots, \boldsymbol{\Lambda}_K]$, then mean vector and precision matrix of $p(\boldsymbol{\theta}|\mathrm{Y}, \mathrm{X}, \boldsymbol{\varphi}, \boldsymbol{\gamma})$ are given by

$$
\begin{aligned}
\boldsymbol{\eta}_{\boldsymbol{\theta}} &= \arg\max_{\boldsymbol{\theta}} \ln \prod_k \mathcal{N}(\boldsymbol{f}_k(\mathrm{X}, \boldsymbol{\theta})|\boldsymbol{m}_k, \boldsymbol{\Lambda}_k^{-1}) \\
&= \arg\max_{\boldsymbol{\theta}} \sum_k \ln \mathcal{N}(\boldsymbol{f}_k(\mathrm{X}, \boldsymbol{\theta})|\boldsymbol{m}_k, \boldsymbol{\Lambda}_k^{-1}) \\
&= \arg\min_{\boldsymbol{\theta}} \frac{1}{2} \sum_k (\boldsymbol{f}_k - \boldsymbol{m}_k)^T \boldsymbol{\Lambda}_k (\boldsymbol{f}_k - \boldsymbol{m}_k) \\
&= \arg\min_{\boldsymbol{\theta}} cost_{\boldsymbol{\theta}}(\mathrm{X}, \boldsymbol{\theta}, \boldsymbol{m}, \boldsymbol{\Lambda}) \qquad (18) \\
\boldsymbol{\Xi}_{\boldsymbol{\theta}}^{-1} &= \nabla\nabla cost_{\boldsymbol{\theta}}|_{\boldsymbol{\theta}=\boldsymbol{\eta}_{\boldsymbol{\theta}}} \qquad (19)
\end{aligned}
$$

Inference algorithm

- Initialize using Gaussian process regression
- Repeat until convergence or maximum iteration
  - Update $\boldsymbol{\theta}$ while keeping the others fixed
  - For $u = 1, \ldots, K$, update $\boldsymbol{x}_u$ while keeping the others fixed
- Calculate precision matrices

Derivation for the gradients and Hessians

Recall that $cost_{\boldsymbol{x}_u}$ for state u is given by

$$cost_{\boldsymbol{x}_u} = \frac{1}{2}[(\boldsymbol{x}_u - \boldsymbol{\mu}_u(\boldsymbol{y}_u))^T \boldsymbol{\Sigma}_u^{-1}(\boldsymbol{x}_u - \boldsymbol{\mu}_u(\boldsymbol{y}_u)) + \sum_k (\boldsymbol{f}_k - \boldsymbol{m}_k)^T \boldsymbol{\Lambda}_k (\boldsymbol{f}_k - \boldsymbol{m}_k)]$$

Using matrix derivative and the fact that $\boldsymbol{\Sigma}_u^{-1}$ is symmetric, we have

$$\nabla_{\boldsymbol{x}_u} \frac{1}{2}(\boldsymbol{x}_u - \boldsymbol{\mu}_u(\boldsymbol{y}_u))^T \boldsymbol{\Sigma}_u^{-1}(\boldsymbol{x}_u - \boldsymbol{\mu}_u(\boldsymbol{y}_u)) = \boldsymbol{\Sigma}_u^{-1}\boldsymbol{x}_u \tag{20}$$

$$\nabla\nabla_{\boldsymbol{x}_u} \frac{1}{2}(\boldsymbol{x}_u - \boldsymbol{\mu}_u(\boldsymbol{y}_u))^T \boldsymbol{\Sigma}_u^{-1}(\boldsymbol{x}_u - \boldsymbol{\mu}_u(\boldsymbol{y}_u)) = \boldsymbol{\Sigma}_u^{-1} \tag{21}$$

Derivation for the gradients and Hessians

Using *chain rule* and the fact that $\mathbf{\Lambda}_k$ is symmetric, we have

$$
\begin{aligned}
&\nabla_{\boldsymbol{x}_u} \frac{1}{2} (\boldsymbol{f}_k - \boldsymbol{m}_k)^T \mathbf{\Lambda}_k (\boldsymbol{f}_k - \boldsymbol{m}_k) \\
&= \begin{bmatrix} \frac{\partial (\boldsymbol{f}_k)_1}{\partial x_u(t_1)} & \cdots & \frac{\partial (\boldsymbol{f}_k)_N}{\partial x_u(t_1)} \\ \vdots & \ddots & \vdots \\ \frac{\partial (\boldsymbol{f}_k)_1}{\partial x_u(t_N)} & \cdots & \frac{\partial (\boldsymbol{f}_k)_N}{\partial x_u(t_N)} \end{bmatrix} \mathbf{\Lambda}_k (\boldsymbol{f}_k - \boldsymbol{m}_k) \\
&\quad - \begin{bmatrix} \frac{\partial (\boldsymbol{m}_k)_1}{\partial x_u(t_1)} & \cdots & \frac{\partial (\boldsymbol{m}_k)_N}{\partial x_u(t_1)} \\ \vdots & \ddots & \vdots \\ \frac{\partial (\boldsymbol{m}_k)_1}{\partial x_u(t_N)} & \cdots & \frac{\partial (\boldsymbol{m}_k)_N}{\partial x_u(t_N)} \end{bmatrix} \mathbf{\Lambda}_k (\boldsymbol{f}_k - \boldsymbol{m}_k)
\end{aligned} \tag{22}
$$

Derivation for the gradients and Hessians

The (i, j)-th entry of the Hessian is given by

$$\begin{aligned}
&\frac{\partial^2 \frac{1}{2}(\boldsymbol{f}_k - \boldsymbol{m}_k)^T \boldsymbol{\Lambda}_k (\boldsymbol{f}_k - \boldsymbol{m}_k)}{\partial x_u(t_i) \partial x_u(t_j)} \\
&= \begin{bmatrix} \frac{\partial^2 (\boldsymbol{f}_k - \boldsymbol{m}_k)_1}{\partial x_u(t_i) \partial x_u(t_j)} & \cdots & \frac{\partial^2 (\boldsymbol{f}_k - \boldsymbol{m}_k)_N}{\partial x_u(t_i) \partial x_u(t_j)} \end{bmatrix} \boldsymbol{\Lambda}_k (\boldsymbol{f}_k - \boldsymbol{m}_k) \\
&\quad + \begin{bmatrix} \frac{\partial (\boldsymbol{f}_k - \boldsymbol{m}_k)_1}{\partial x_u(t_j)} & \cdots & \frac{\partial (\boldsymbol{f}_k - \boldsymbol{m}_k)_N}{\partial x_u(t_j)} \end{bmatrix} \boldsymbol{\Lambda}_k \begin{bmatrix} \frac{\partial (\boldsymbol{f}_k - \boldsymbol{m}_k)_1}{\partial x_u(t_i)} \\ \vdots \\ \frac{\partial (\boldsymbol{f}_k - \boldsymbol{m}_k)_N}{\partial x_u(t_i)} \end{bmatrix}
\end{aligned} \tag{23}$$

Positivity constraint

Let

$$\boldsymbol{\theta} = [\theta_1, \ldots, \theta_M]^\top = [e^{\widetilde{\theta}_1}, \ldots, e^{\widetilde{\theta}_M}]^\top \tag{24}$$

Since the exponential function is monotonic, we can first find

$$\begin{aligned} \widetilde{\boldsymbol{\theta}}^* &= \arg\min_{\widetilde{\boldsymbol{\theta}}} cost_{\boldsymbol{\theta}}(\mathrm{X}, e^{\widetilde{\boldsymbol{\theta}}}, \boldsymbol{m}, \boldsymbol{\Lambda}) \\ &= \arg\min_{\widetilde{\boldsymbol{\theta}}} \sum_k \ln \mathcal{N}(\boldsymbol{f}_k(\boldsymbol{X}, e^{\widetilde{\boldsymbol{\theta}}}) | \boldsymbol{m}_k, \boldsymbol{\Lambda}_k^{-1})) \end{aligned} \tag{25}$$

and then obtain $\boldsymbol{\theta}^*$ as

$$\boldsymbol{\theta}^* = [e^{\widetilde{\theta}_1^*}, \ldots, e^{\widetilde{\theta}_M^*}]^\top \tag{26}$$

Since $e^r > 0$ for any $r \in \mathbb{R}$, we essentially restrict $\boldsymbol{\theta}^*$ to positive values.

Positivity constraint

The positivity constraint on states can be achieved by transforming $cost_{\boldsymbol{x}_u}$ to $cost_{\widetilde{\boldsymbol{x}}_u}$, where we define for $u = 1, \ldots, K$

$$\boldsymbol{x}_u = [x_u(t_1), \ldots, x_u(t_N)]^\top = [e^{\widetilde{x}_u(t_1)}, \ldots, e^{\widetilde{x}_u(t_N)}]^\top \tag{27}$$

Using chain rule, we have for $u = 1, \ldots, K$ and $n = 1, \ldots, N$ the following:

$$\begin{aligned} \frac{d\widetilde{x}_u(t_n)}{dt} &= \frac{d \ln x_u(t_n)}{dt} \\ &= \frac{1}{x_u(t_n)} \frac{dx_u(t_n)}{dt} \\ &= \frac{f_u(e^{\widetilde{\boldsymbol{x}}(t_n)}, \boldsymbol{\theta})}{e^{\widetilde{x}(t_n)}} \end{aligned} \tag{28}$$

Positivity constraint

Caveats

- The covariance matrix for the new variables cannot be transformed back to the original variables easily.
- Probabilistic interpretability is lost since proper change of random variables requires the evaluation of the Jacobian determinant, which is computationally expensive.

Extension to Random Dynamical Systems

Random ordinary differential equations (RODEs)

Given a complete probability space $(\boldsymbol{\Omega}, \mathcal{F}, \mathbb{P})$, let

$(\boldsymbol{\zeta}_t)_{t\in[0,T]} \in \mathbb{R}^W$ be a stochastic process with continuous sample paths,

$\boldsymbol{f} : \mathbb{R}^K \times \mathbb{R}^W \mapsto \mathbb{R}^K$ be a continuous function.

For all $\boldsymbol{\omega} \in \boldsymbol{\Omega}$, a K-dimensional RODE defined as

$$\frac{d\boldsymbol{x}(t)}{dt} = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{\zeta}_t(\boldsymbol{\omega})) \tag{29}$$

is a *non-autonomous* ODE system

$$\dot{\boldsymbol{x}}(t) = \frac{d\boldsymbol{x}(t)}{dt} = \boldsymbol{F}_{\boldsymbol{\omega}}(\boldsymbol{x}, t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{\omega}_t) \tag{30}$$

RODEs

An example of a scalar RODE with additive noise is given by

$$\frac{dx(t)}{dt} = -x + \cos\left(W_t(\omega)\right) \tag{31}$$

where W_t is a one-dimensional Wiener process.

To ensure the existence of a unique solution for the initial value problem on the finite time interval $[0, T]$, we assume that $\boldsymbol{f}$ is infinitely differentiable in $\boldsymbol{x}$, and hence, it is locally *Lipschitz* in $\boldsymbol{x}$.

RODEs

Since ζ_t is usually *Hölder continuous* in t, the vector fields of F_ω are continuous but not differentiable in t for every $\omega \in \Omega$.

- Numerical schemes, e.g. the *Runge-Kutta* method, fail to achieve high order of convergence.
- The solution paths of the RODEs are once differentiable, which implies that the gradient matching model can be ideally applied.

Solve many RODEs sample paths deterministically in parallel to obtain an *ensemble* solution.

- The Laplace mean-field approximation is computationally very efficient.

Doss-Sussmann/Imkeller-Schmalfuss correspondence

Any finite dimensional SDE system with additive noise can be transformed to an equivalent RODE system and vice versa as follows:

$$d\boldsymbol{x}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{\theta})dt + d\boldsymbol{W}_t \rightleftarrows \frac{d\boldsymbol{z}(t)}{dt} = \boldsymbol{f}(\boldsymbol{z}(t) + \boldsymbol{O}_t, \boldsymbol{\theta}) + \boldsymbol{O}_t \tag{32}$$

where $\boldsymbol{z}(t) = \boldsymbol{x}(t) - \boldsymbol{O}_t$ and $\boldsymbol{O}_t$ is a stationary stochastic *Ornstein-Uhlenbeck* process defined as

$$d\boldsymbol{O}_t = -\boldsymbol{O}_t dt + d\boldsymbol{W}_t \tag{33}$$

Using the scalar RODE in (Eq. 31) as an example,

$$d\begin{pmatrix} x_t \\ y_t \end{pmatrix} = \begin{pmatrix} -x_t + \cos(y_t) \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 1 \end{pmatrix} dW_t \rightleftarrows \frac{dx(t)}{dt} = -x + \cos(W_t(\omega)) \tag{34}$$

Inference algorithm

1: Transform the SDEs into RODEs.
2: Generate N_{paths} RODEs sample paths using each time an independently generated Ornstein-Uhlenbeck process sample path.
3: **for** $i = 1, \ldots, N_{paths}$ **do**
4: Estimate using Laplace mean-field approximation
5: **end for**
6: Average the estimation results from all the sample paths.

Experiments

Lotka-Volterra model

Two non-linear ODEs to model the interaction between prey and predator.

$$\begin{aligned} \dot{x}(t) &= \alpha x(t) - \beta x(t) y(t) \\ \dot{y}(t) &= \delta x(t) y(t) - \gamma y(t) \end{aligned} \tag{35}$$

where

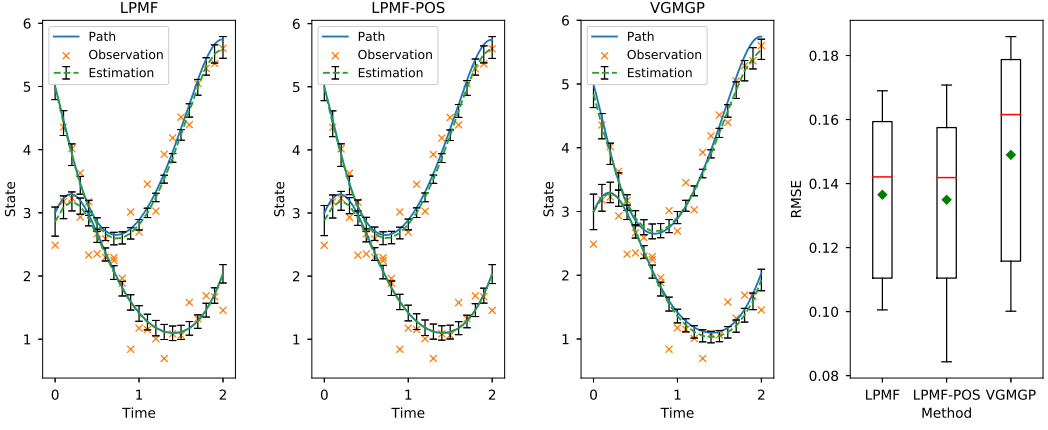
$x(t), y(t) \in \mathbb{R}_{\geqslant 0}$ are the populations of prey and predator at time t.

$\alpha, \beta, \delta, \gamma \in \mathbb{R}^{+}$ are the parameters controlling the dynamics.

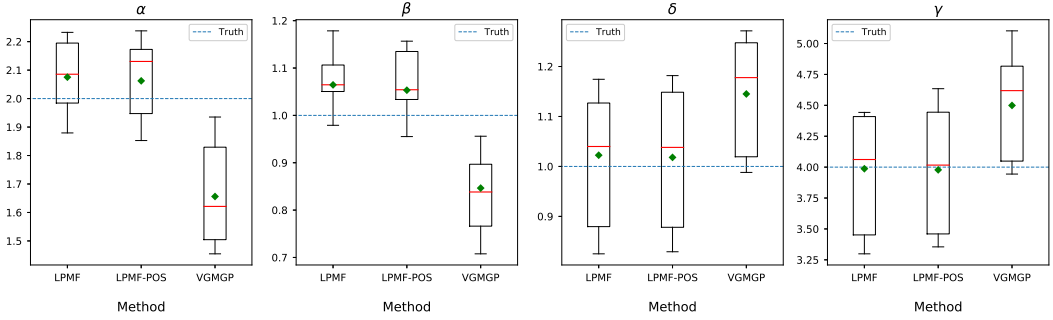
Experimental setup

K	K_{obs}	t_0	t_T	δt	$\alpha, \beta, \delta, \gamma$	σ_k^2	$freq_{obs}$
2	2	0	2	0.01	2, 1, 1, 4	0.1	10

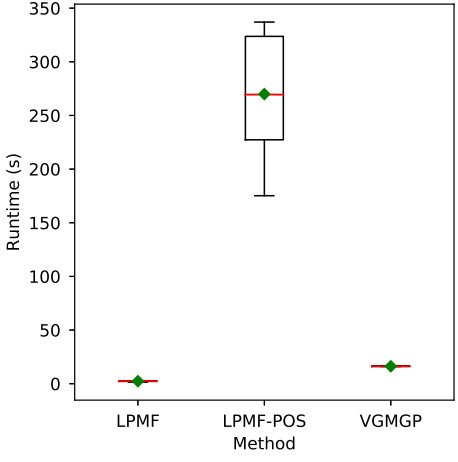
State estimation



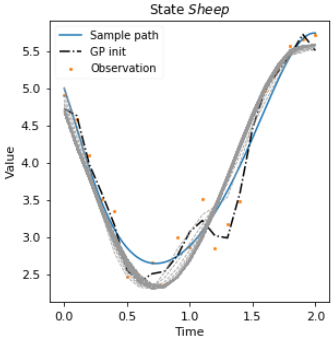
Parameter estimation

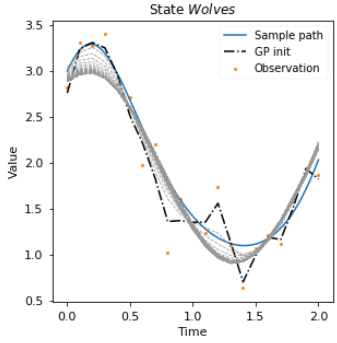


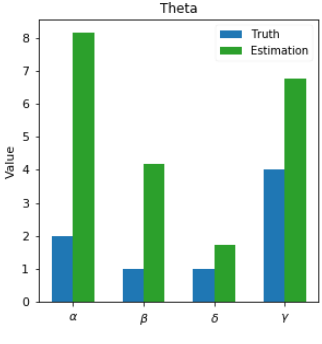
Runtime performance



Positivity constraint on states







Protein signalling transduction pathway

A signal transduction cascade model to describe the dynamics among protein species.

$$
\begin{aligned}
\dot{S} &= -k_1 \times S - k_2 \times S \times R + k_3 \times RS \\
\dot{dS} &= k_1 \times S \\
\dot{R} &= -k_2 \times S \times R + k_3 \times RS + V \times \frac{Rpp}{K_m + Rpp} \\
\dot{RS} &= k_2 \times S \times R - k_3 \times RS - k_4 \times RS \\
\dot{Rpp} &= k_4 \times RS - V \times \frac{Rpp}{K_m + Rpp}
\end{aligned}
\tag{36}
$$

where

$S, dS, R, RS, Rpp \in \mathbb{R}_{\geqslant 0}$ are the concentrations of proteins.

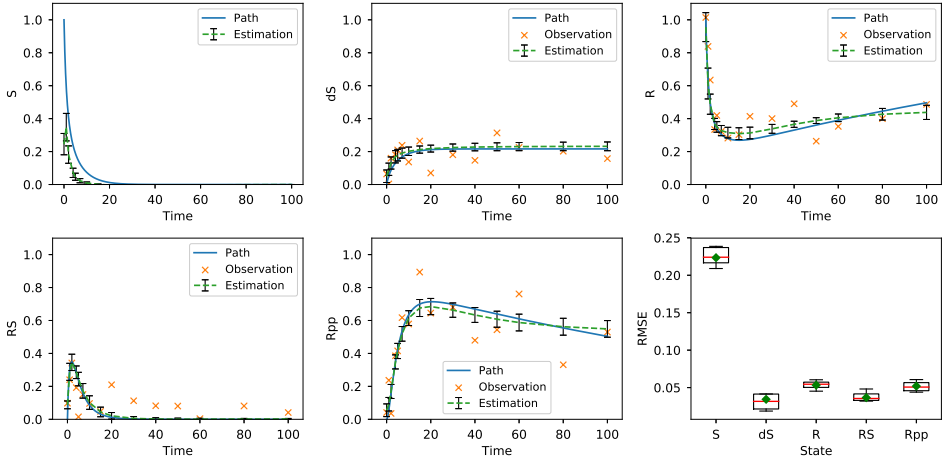
$k_1, k_2, k_3, k_4, K_m, V \in \mathbb{R}^+$ are kentic parameters

Experimental setup

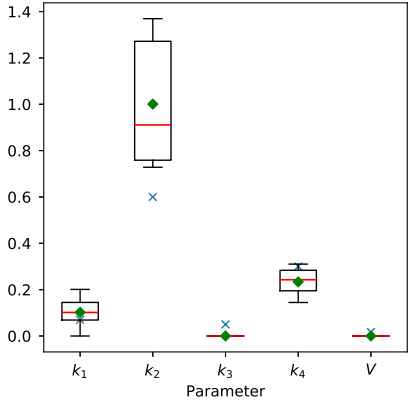
K	K_{obs}	t_0	t_T	δt	$k_1, k_2, k_3, k_4, V, Km$	σ_k^2
5	4	0	100	0.05	0.07, 0.6, 0.05, 0.3, 0.017, 3	0.01

The observations, in total 15, are collected at time points 0, 1, 2, 4, 5, 7, 10, 15, 20, 30, 40, 50, 60, 80 and 100.

State estimation



Parameter estimation



Lorenz 96 model

A minimalistic weather forecast model. A K-dimensional deterministic Lorenz 96 dynamical system is defined for $k = 1, \ldots, K$, state-wise as follows:

$$\dot{x}_k(t) = (x_{k+1}(t) - x_{k-2}(t))x_{k-1}(t) - x_k(t) + F \tag{37}$$

where

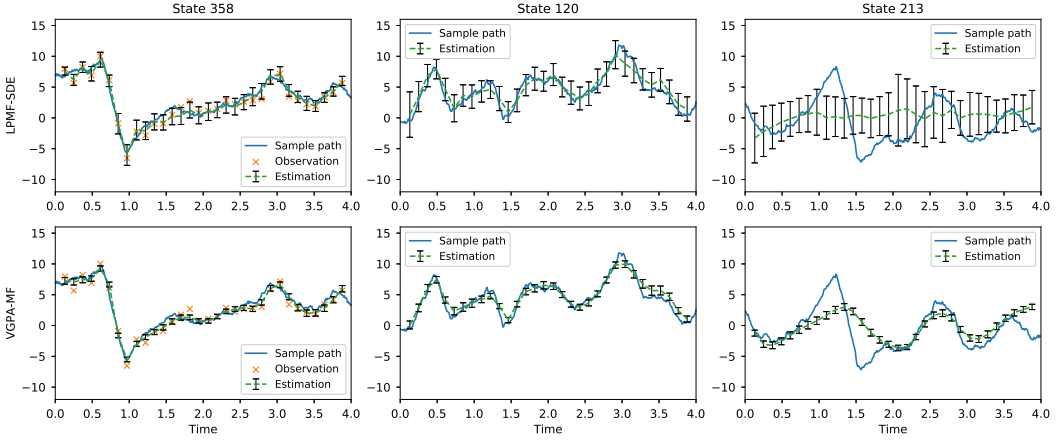
$$x_{-1}(t) = x_{K-1}(t)$$
$$x_0(t) = x_K(t)$$
$$x_{K+1}(t) = x_1(t)$$

$F \in \mathbb{R}$ controls the behavior of the system (38)

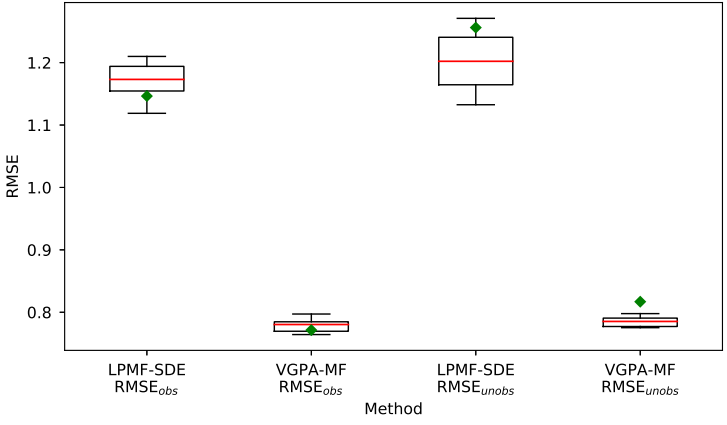
Experimental setup

K	K_{obs}	t_0	t_T	δt	F	σ_k^2	ρ_k^2	$freq_{obs}$
500	325 (65%)	0	4	0.01	8	1	4	8

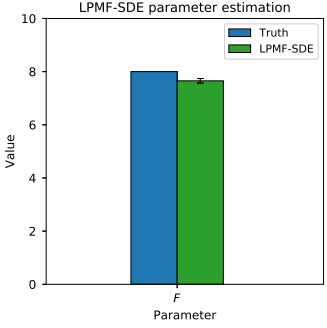
State estimation

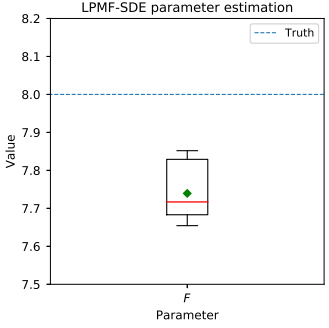


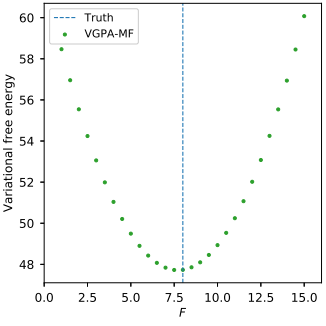
State estimation



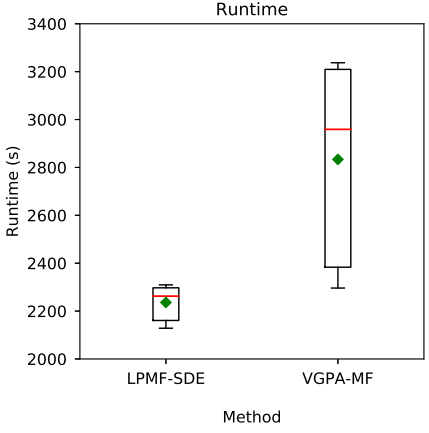
Parameter estimation

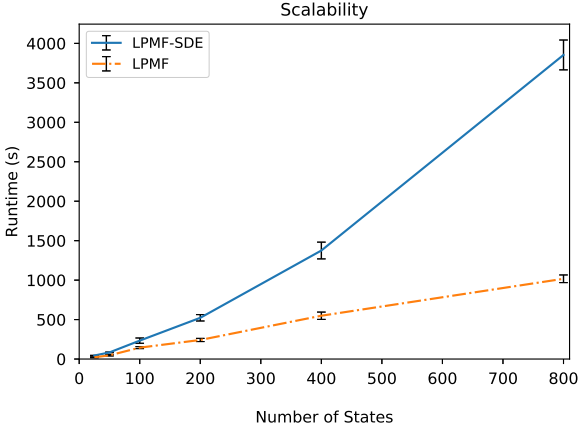






Runtime performance





Lorenz 63 model

Low dimensional mathematical model for thermal convection in the atmosphere. The vector field of the deterministic Lorenz 63 is defined as follows:

$$\boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{\theta}) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{z}(t) \end{bmatrix} = \begin{bmatrix} \sigma(y(t) - x(t)) \\ x(t)(\rho - z(t)) - y(t) \\ x(t)y(t) - \beta z(t) \end{bmatrix} \tag{39}$$

where

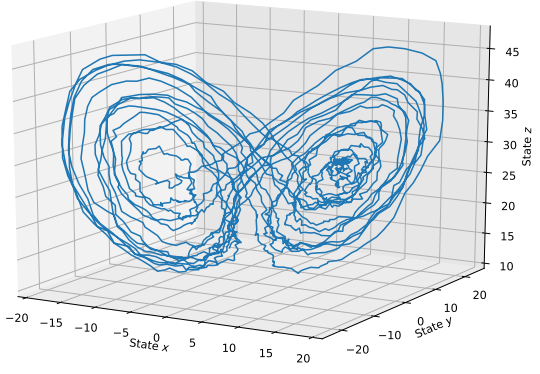
$\boldsymbol{x}(t) = [x(t), y(t), z(t)]^\top \in \mathbb{R}^3$ is the state vector.

$\boldsymbol{\theta} = [\sigma, \rho, \beta]^\top \in \mathbb{R}^3$ is the parameter vector

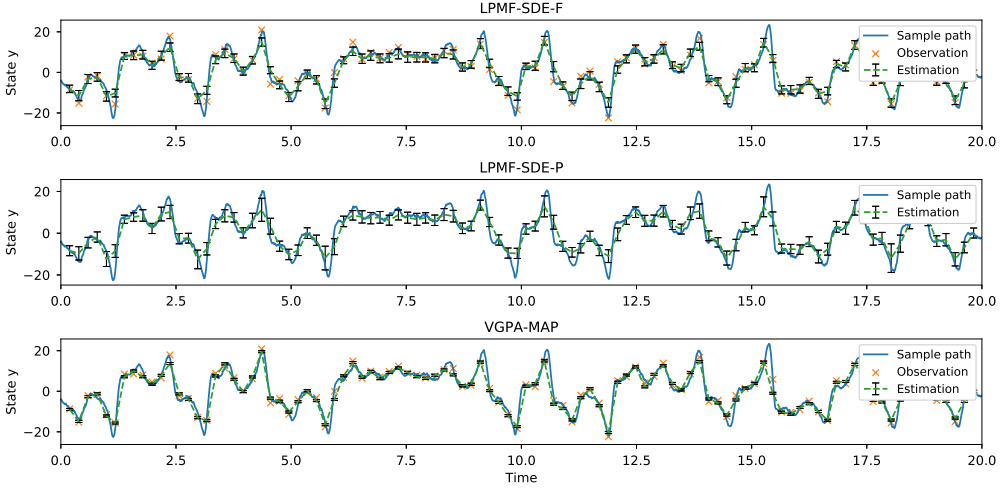
Experimental setup

K	K_{obs}	t_0	t_T	δt	σ, ρ, β	σ_k^2	ρ_k^2	$freq_{obs}$
3	3 or 2	0	20	0.01	$10, 28, \frac{8}{3}$	2	10	5

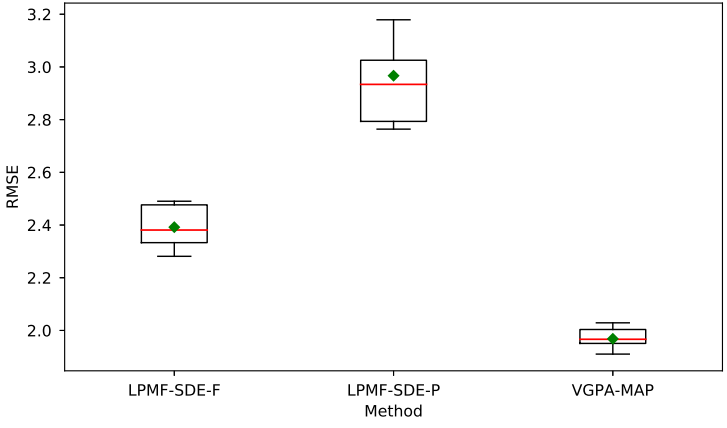
Sample path


45
40
35
30
25
20
15
10
State z
−20
−15
−10
−5
0
5
10
15
20
State x
20
10
0
−10
−20
State y

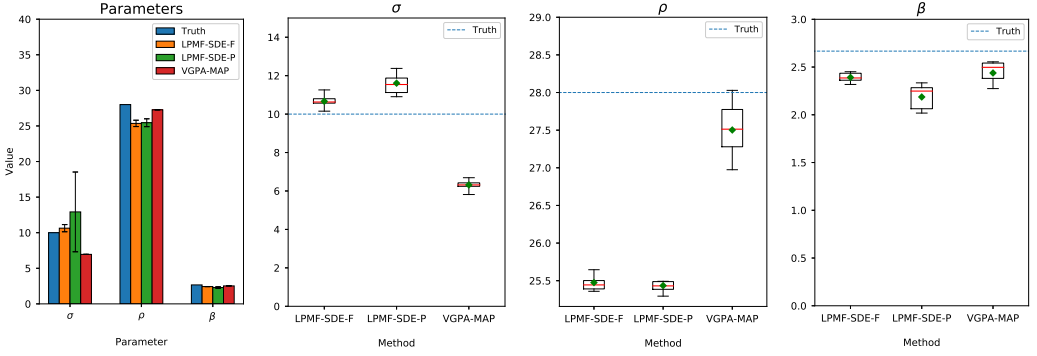
State estimation



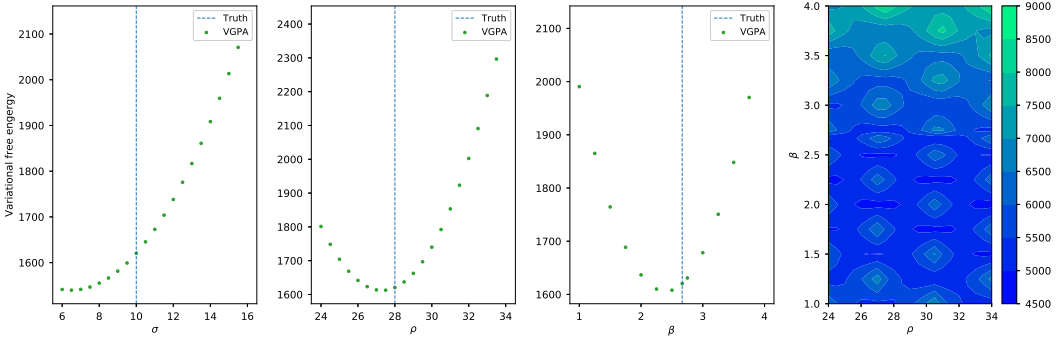
State estimation



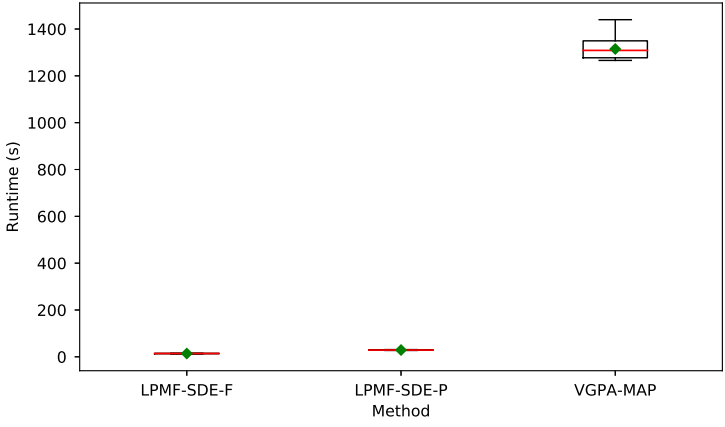
Parameter estimation



Parameter estimation



Runtime performance



Conclusion

Contributions

- Implemented a brand new Python inference library.
- Proposed the Laplace mean-field approximation for ODEs.
- Derived analytical solution for the gradients and Hessians.
- Improved the flexibility of the ODE inference algorithm by
  - relaxing the structural assumption on the dynamical systems.
  - adding support for positivity constraints on states and parameters.
- Developed an efficient parallel inference pipeline for SDEs.

Future work

- Add support to infer the diffusion noise.
- Add support to select hyperparameters.
- Improve the numerical optimization strategy.
- Add GPU implementation.

Q & A