Learning optimal filters using variational inference

Eviatar Bach

24 October 2024

University of Reading

Inverse Problems and Data Assimilation: A Machine Learning Approach

Eviatar Bach [♯♮], Ricardo Baptista [♯], Daniel Sanz-Alonso [♭], Andrew Stuart [♯]

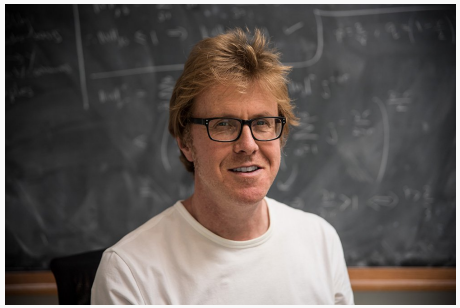
https://www.arxiv.org/abs/2410.10523

E. Luk, E. Bach, R. Baptista, and A. Stuart (2024). ***Learning Optimal Filters Using Variational Inference.*** arXiv: 2406.18066[cs,math]

Collaborators

Enoch Luk (Caltech)

Ricardo Baptista (Caltech)



Andrew Stuart (Caltech)

Data assimilation

Consider the dynamical system

$$x_{j+1}^{\dagger} = \Psi(x_j^{\dagger}) + \xi_j^{\dagger}, \tag{1a}$$

$$x_0^{\dagger} \sim \mathcal{N}(m_0, C_0), \quad \xi_j^{\dagger} \sim \mathcal{N}(0, Q) \text{ i.i.d.} \tag{1b}$$

Data assimilation

Consider the dynamical system

$$x_{j+1}^{\dagger} = \Psi(x_j^{\dagger}) + \xi_j^{\dagger}, \tag{1a}$$

$$x_0^{\dagger} \sim \mathcal{N}(m_0, C_0), \quad \xi_j^{\dagger} \sim \mathcal{N}(0, Q) \text{ i.i.d.} \tag{1b}$$

The observations are given by

$$y_{j+1}^{\dagger} = h(x_{j+1}^{\dagger}) + \eta_{j+1}^{\dagger}, \tag{2a}$$

$$\eta_j^{\dagger} \sim \mathcal{N}(0, R) \quad \text{i.i.d.} \tag{2b}$$

Data assimilation

Consider the dynamical system

$$x_{j+1}^\dagger = \Psi(x_j^\dagger) + \xi_j^\dagger, \tag{1a}$$

$$x_0^\dagger \sim \mathcal{N}(m_0, C_0), \quad \xi_j^\dagger \sim \mathcal{N}(0, Q) \text{ i.i.d.} \tag{1b}$$

The observations are given by

$$y_{j+1}^\dagger = h(x_{j+1}^\dagger) + \eta_{j+1}^\dagger, \tag{2a}$$

$$\eta_j^\dagger \sim \mathcal{N}(0, R) \quad \text{i.i.d.} \tag{2b}$$

We define, for a fixed integer J,

$$Y_j^\dagger = \{y_1^\dagger, \ldots, y_j^\dagger\}.$$

Data assimilation

Consider the dynamical system

$$x_{j+1}^\dagger = \Psi(x_j^\dagger) + \xi_j^\dagger, \tag{1a}$$

$$x_0^\dagger \sim \mathcal{N}(m_0, C_0), \quad \xi_j^\dagger \sim \mathcal{N}(0, Q) \text{ i.i.d.} \tag{1b}$$

The observations are given by

$$y_{j+1}^\dagger = h(x_{j+1}^\dagger) + \eta_{j+1}^\dagger, \tag{2a}$$

$$\eta_j^\dagger \sim \mathcal{N}(0, R) \quad \text{i.i.d.} \tag{2b}$$

We define, for a fixed integer J,

$$Y_j^\dagger = \{y_1^\dagger, \ldots, y_j^\dagger\}.$$

The problem of *filtering* is to obtain $\mathbb{P}(x_j^\dagger | Y_j^\dagger)$ for any $j \leq J$.

Solution to the filtering problem

The solution to the filtering problem has a forecast–assimilation structure: $\hat{\pi}_{j+1} = \mathbb{P}(x_{j+1}^{\dagger} | Y_j^{\dagger})$ and decompose the update $\pi_j \mapsto \pi_{j+1}$ into the following two steps:

Solution to the filtering problem

The solution to the filtering problem has a forecast–assimilation structure: $\hat{\pi}_{j+1} = \mathbb{P}(x_{j+1}^{\dagger} | Y_j^{\dagger})$ and decompose the update $\pi_j \mapsto \pi_{j+1}$ into the following two steps:

$$\begin{aligned} &\textbf{Prediction Step:} && \hat{\pi}_{j+1} = \mathsf{P}\pi_j, \\ &\textbf{Analysis Step:} && \pi_{j+1} = \mathsf{A}(\hat{\pi}_{j+1}; y_{j+1}^{\dagger}), \end{aligned} \tag{3}$$

where

$$\nu_\pi(u, y) \propto \exp\left(-\frac{1}{2}\left\|y - h(u)\right\|_R^2\right) \pi(u)$$

and

$$\mathsf{A}(\pi, y^{\dagger})(u) = \frac{\nu_\pi(u, y^{\dagger})}{\int_{\mathbb{R}^d} \nu_\pi(u, y^{\dagger})\, du}. \tag{4}$$

Solution to the filtering problem

The solution to the filtering problem has a forecast–assimilation structure: $\hat{\pi}_{j+1} = \mathbb{P}(x_{j+1}^\dagger | Y_j^\dagger)$ and decompose the update $\pi_j \mapsto \pi_{j+1}$ into the following two steps:

$$\begin{aligned} &\textbf{Prediction Step:} && \hat{\pi}_{j+1} = \mathrm{P}\pi_j, \\ &\textbf{Analysis Step:} && \pi_{j+1} = \mathrm{A}(\hat{\pi}_{j+1}; y_{j+1}^\dagger), \end{aligned} \tag{3}$$

where

$$\nu_\pi(u, y) \propto \exp\left(-\frac{1}{2}\|y - h(u)\|_R^2\right) \pi(u)$$

and

$$\mathrm{A}(\pi, y^\dagger)(u) = \frac{\nu_\pi(u, y^\dagger)}{\int_{\mathbb{R}^d} \nu_\pi(u, y^\dagger)\, du}. \tag{4}$$

The analysis step thus multiplies by the likelihood $\mathbb{P}(y_{j+1}^\dagger | x_{j+1})$ and normalizes, corresponding to an application of Bayes' Theorem with the prior $\hat{\pi}_{j+1}$.

Introduction

The aim of probabilistic DA is to estimate A, the analysis step.

Introduction

The aim of probabilistic DA is to estimate A, the analysis step.

Existing filters used in practice are suboptimal and involve numerous tuning parameters.

Introduction

The aim of probabilistic DA is to estimate A, the analysis step.

Existing filters used in practice are suboptimal and involve numerous tuning parameters.

Here we consider learning the analysis step of the DA process to get improved filters:

Introduction

The aim of probabilistic DA is to estimate A, the analysis step.

Existing filters used in practice are suboptimal and involve numerous tuning parameters.

Here we consider learning the analysis step of the DA process to get improved filters:

- Learn tuning parameters in existing filtering methods.

Introduction

The aim of probabilistic DA is to estimate A, the analysis step.

Existing filters used in practice are suboptimal and involve numerous tuning parameters.

Here we consider learning the analysis step of the DA process to get improved filters:

- Learn tuning parameters in existing filtering methods.
- Learn new filtering algorithms closer to optimality.

Motivation

Other work has considered learning analysis steps for the purpose of *state estimation*, trained using RMSE.

Motivation

Other work has considered learning analysis steps for the purpose of *state estimation*, trained using RMSE.

Here, we want to learn a probabilistic filter, so RMSE will not do. We need a cost function that is minimized at, and only at, the filtering distribution.

Variational inference

The *Kullback–Leibler (KL) divergence* measures "distance" between probability distributions q and π:

$$\mathrm{D}_{\mathrm{KL}}(q \| \pi) = \int \log\left(\frac{q(u)}{\pi(u)}\right) q(u) du. \tag{5}$$

Variational inference

The *Kullback–Leibler (KL) divergence* measures "distance" between probability distributions q and π:

$$\mathrm{D}_{\mathrm{KL}}(q\|\pi) = \int \log\left(\frac{q(u)}{\pi(u)}\right) q(u)du. \tag{5}$$

Fact: $\mathrm{D}_{\mathrm{KL}}(q\|\pi) \geq 0$ and is minimized if and only if $q = \pi$.

Variational inference

Theorem (Variational formulation of Bayes' Theorem)

$$\mathsf{J}(q) = \mathsf{D}_{\mathsf{KL}}(q||\mathbb{P}(u)) - \mathbb{E}^q[\log \mathbb{P}(y|u)], \tag{6a}$$

$$\mathbb{P}(u|y) = \operatorname*{arg\,min}_{q \in \mathcal{P}} \mathsf{J}(q), \tag{6b}$$

where $\mathcal{P}$ is the set of all probability distributions on $\mathbb{R}^d$.

Variational inference

Theorem (Variational formulation of Bayes' Theorem)

$$\mathsf{J}(q) = \mathsf{D}_{\mathsf{KL}}(q||\mathbb{P}(u)) - \mathbb{E}^q[\log \mathbb{P}(y|u)], \tag{6a}$$

$$\mathbb{P}(u|y) = \operatorname*{arg\,min}_{q \in \mathcal{P}} \mathsf{J}(q), \tag{6b}$$

where $\mathcal{P}$ is the set of all probability distributions on $\mathbb{R}^d$.

Notice that the objective is a tradeoff between closeness to the prior (first term) and closeness to observations (second term).

Variational inference

Theorem (Variational formulation of Bayes' Theorem)

$$\mathsf{J}(q) = \mathsf{D}_{\mathsf{KL}}(q||\mathbb{P}(u)) - \mathbb{E}^q[\log \mathbb{P}(y|u)], \tag{6a}$$

$$\mathbb{P}(u|y) = \operatorname*{arg\,min}_{q \in \mathcal{P}} \mathsf{J}(q), \tag{6b}$$

where $\mathcal{P}$ is the set of all probability distributions on $\mathbb{R}^d$.

Notice that the objective is a tradeoff between closeness to the prior (first term) and closeness to observations (second term).

With the variational formulation of Bayes' Theorem, we turn the inference problem into an optimization ("variational") problem.

Mathematical framework

Recall that

$$\mathsf{A}(\pi, y^\dagger)(u) = \frac{\nu_\pi(u, y^\dagger)}{\int_{\mathbb{R}^d} \nu_\pi(u, y^\dagger)\, du}, \quad \pi_{j+1} = \mathsf{A}(\hat{\pi}_{j+1}, y^\dagger_{j+1}).$$

Mathematical framework

Recall that

$$\mathsf{A}(\pi, y^\dagger)(u) = \frac{\nu_\pi(u, y^\dagger)}{\int_{\mathbb{R}^d} \nu_\pi(u, y^\dagger)\, du}, \quad \pi_{j+1} = \mathsf{A}(\hat{\pi}_{j+1}, y^\dagger_{j+1}).$$

Variational inference can be used to transform the analysis step to an optimization problem

$$\mathsf{J}_{j+1}(q) = \mathsf{D}_{\mathsf{KL}}(q \| \hat{\pi}_{j+1}) - \mathbb{E}^{x_{j+1} \sim q}[\log \mathbb{P}(y^\dagger_{j+1} | x_{j+1})], \tag{7a}$$

$$\pi_{j+1} = \arg\min_q \mathsf{J}_{j+1}(q). \tag{7b}$$

Mathematical framework

Recall that

$$\mathsf{A}(\pi, y^\dagger)(u) = \frac{\nu_\pi(u, y^\dagger)}{\int_{\mathbb{R}^d} \nu_\pi(u, y^\dagger)\, du}, \quad \pi_{j+1} = \mathsf{A}(\hat{\pi}_{j+1}, y^\dagger_{j+1}).$$

Variational inference can be used to transform the analysis step to an optimization problem

$$\mathsf{J}_{j+1}(q) = \mathsf{D}_{\mathsf{KL}}(q \| \hat{\pi}_{j+1}) - \mathbb{E}^{x_{j+1}\sim q}[\log \mathbb{P}(y^\dagger_{j+1}|x_{j+1})], \tag{7a}$$

$$\pi_{j+1} = \operatorname*{arg\,min}_q \mathsf{J}_{j+1}(q). \tag{7b}$$

Suppose we introduce parameters θ into an approximate analysis map:

$$q_{j+1}(\theta) = \mathsf{A}_\theta(\mathsf{P}q_j(\theta); y^\dagger_{j+1}), \quad q_0 = \pi_0, \tag{8}$$

e.g., tuning parameters in the EnKF analysis step or neural network parameters.

Method

We then consider the optimization problem

$$\mathsf{J}_{j+1}(\theta) = \mathsf{D}_{\mathsf{KL}}(q_{j+1}(\theta) \| \mathsf{P} q_j) - \mathbb{E}^{x_{j+1} \sim q_{j+1}(\theta)} [\log \mathbb{P}(y_{j+1}^{\dagger} | x_{j+1})]. \quad (9)$$

Method

We then consider the optimization problem

$$\mathsf{J}_{j+1}(\theta) = \mathsf{D}_{\mathsf{KL}}(q_{j+1}(\theta) \| \mathsf{P} q_j) - \mathbb{E}^{x_{j+1} \sim q_{j+1}(\theta)}[\log \mathbb{P}(y^{\dagger}_{j+1} | x_{j+1})]. \quad (9)$$

Only the θ which recovers the true filtering distribution will minimize the cost function!

Method

We then consider the optimization problem

$$\mathrm{J}_{j+1}(\theta) = \mathrm{D}_{\mathrm{KL}}(q_{j+1}(\theta) \| \mathrm{P} q_j) - \mathbb{E}^{x_{j+1} \sim q_{j+1}(\theta)}[\log \mathbb{P}(y_{j+1}^{\dagger} | x_{j+1})]. \quad (9)$$

Only the θ which recovers the true filtering distribution will minimize the cost function!

Then, using automatic differentiation and an approximate filter parameterized by θ, the sum $\sum_{j=1}^{J} \mathrm{J}_j(\theta)$ can be minimized to find the best filter parameters over a trajectory (offline method).

Method

We then consider the optimization problem

$$\mathrm{J}_{j+1}(\theta) = \mathrm{D}_{\mathrm{KL}}(q_{j+1}(\theta) \| \mathrm{P} q_j) - \mathbb{E}^{x_{j+1} \sim q_{j+1}(\theta)}[\log \mathbb{P}(y_{j+1}^{\dagger} | x_{j+1})]. \quad (9)$$

Only the θ which recovers the true filtering distribution will minimize the cost function!

Then, using automatic differentiation and an approximate filter parameterized by θ, the sum $\sum_{j=1}^{J} \mathrm{J}_j(\theta)$ can be minimized to find the best filter parameters over a trajectory (offline method).

Alternatively, $\mathrm{J}_j(\theta)$ can be minimized at each j (online method).

Method

We then consider the optimization problem

$$\mathsf{J}_{j+1}(\theta) = \mathsf{D}_{\mathsf{KL}}(q_{j+1}(\theta) \| \mathsf{P} q_j) - \mathbb{E}^{x_{j+1} \sim q_{j+1}(\theta)}[\log \mathbb{P}(y_{j+1}^{\dagger} | x_{j+1})]. \quad (9)$$

Only the θ which recovers the true filtering distribution will minimize the cost function!

Then, using automatic differentiation and an approximate filter parameterized by θ, the sum $\sum_{j=1}^{J} \mathsf{J}_j(\theta)$ can be minimized to find the best filter parameters over a trajectory (offline method).

Alternatively, $\mathsf{J}_j(\theta)$ can be minimized at each j (online method).

jax was used for all the experiments that follow.

Gain learning

We first learn a fixed gain K in a 3DVar-like method:

$$x_{j+1} = \widehat{x}_{j+1} + K(y^{\dagger}_{j+1} + \eta_{j+1} - H\widehat{x}_{j+1}), \tag{10}$$

for a linear dynamical system.

Gain learning

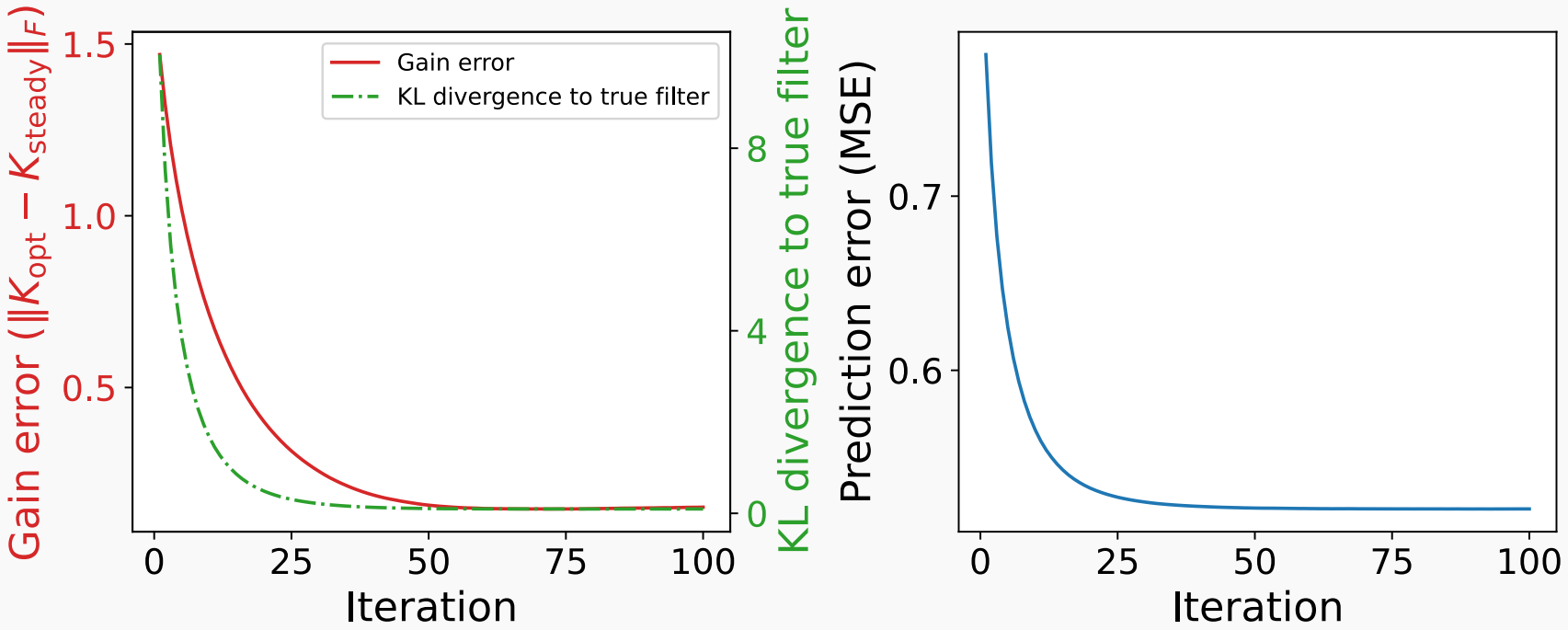
We first learn a fixed gain K in a 3DVar-like method:

$$x_{j+1} = \widehat{x}_{j+1} + K(y^{\dagger}_{j+1} + \eta_{j+1} - H\widehat{x}_{j+1}), \tag{10}$$

for a linear dynamical system.

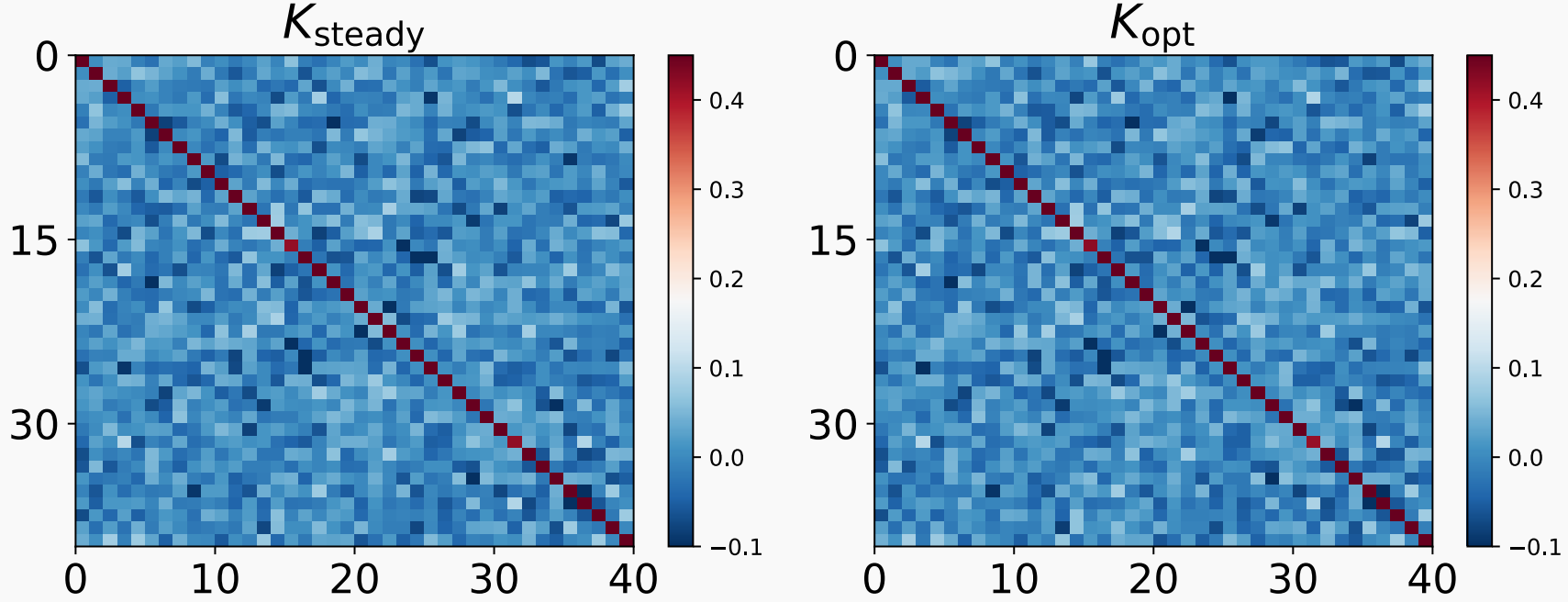
In this case the analytical solution as $J \to \infty$ is the steady-state Kalman gain.

Gain learning loss



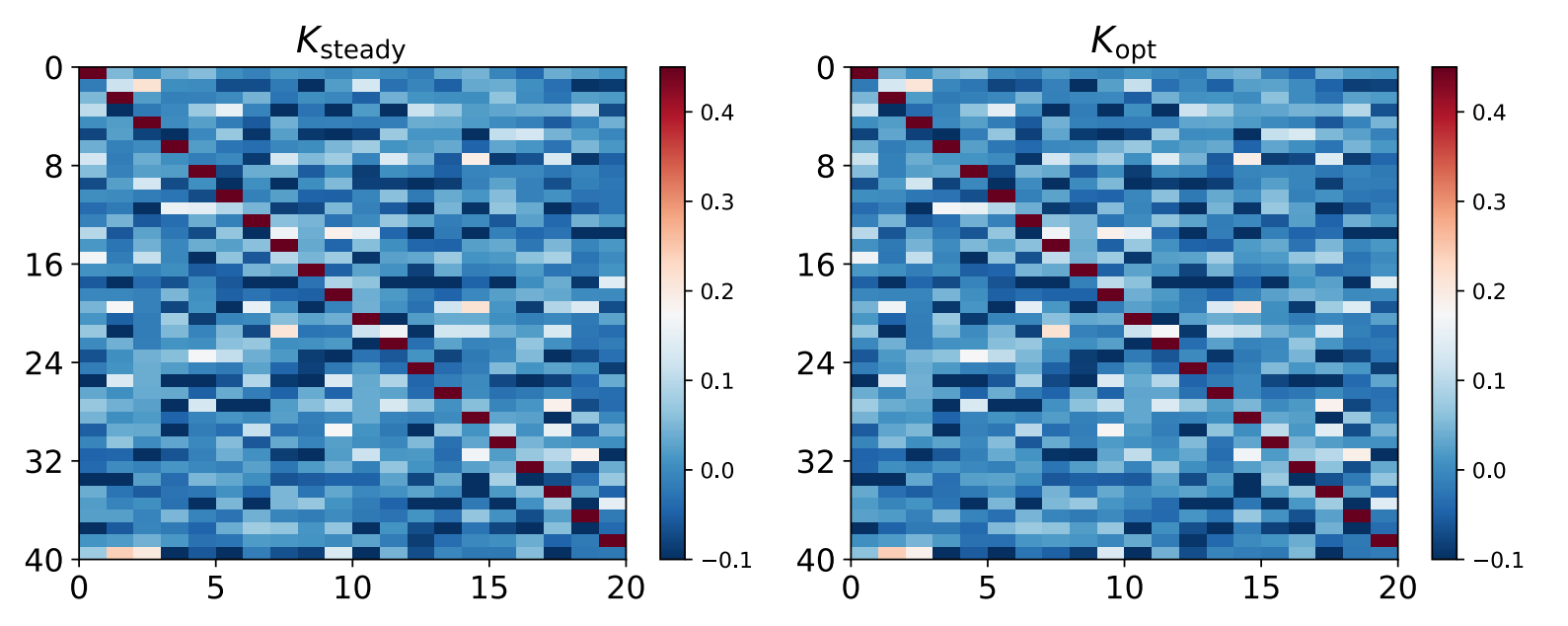
Errors as a function of gradient descent iterations. The Frobenius norm of the difference between the learned gain and the steady-state gain (red solid line). The KL divergence between the learned filtering distribution and the Kalman filter distribution (green dashed line). The mean-square prediction error in the filter mean compared to the true trajectory (blue solid line).

Gain learning matrices



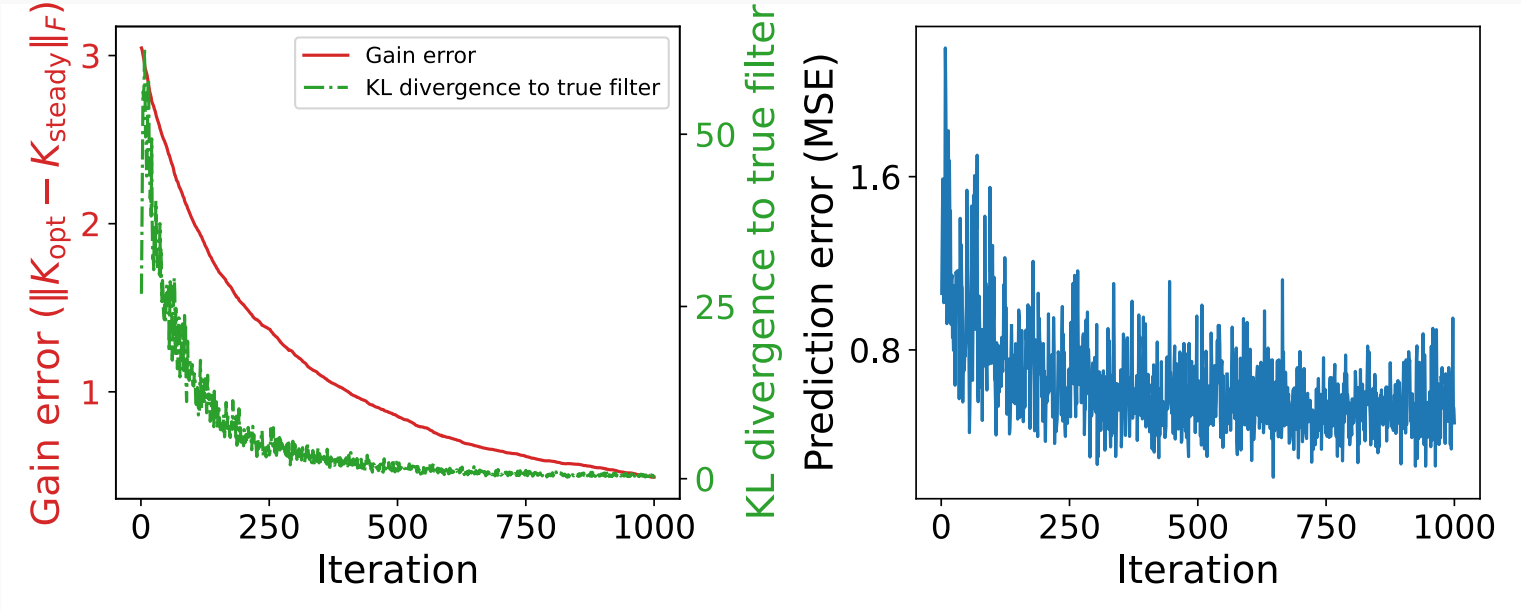
A comparison between the steady-state Kalman gain (left) and the learned gain (right).

50% observations



A comparison between the steady-state Kalman gain (left) and the learned gain (right) for the partially observed case.

Online learning



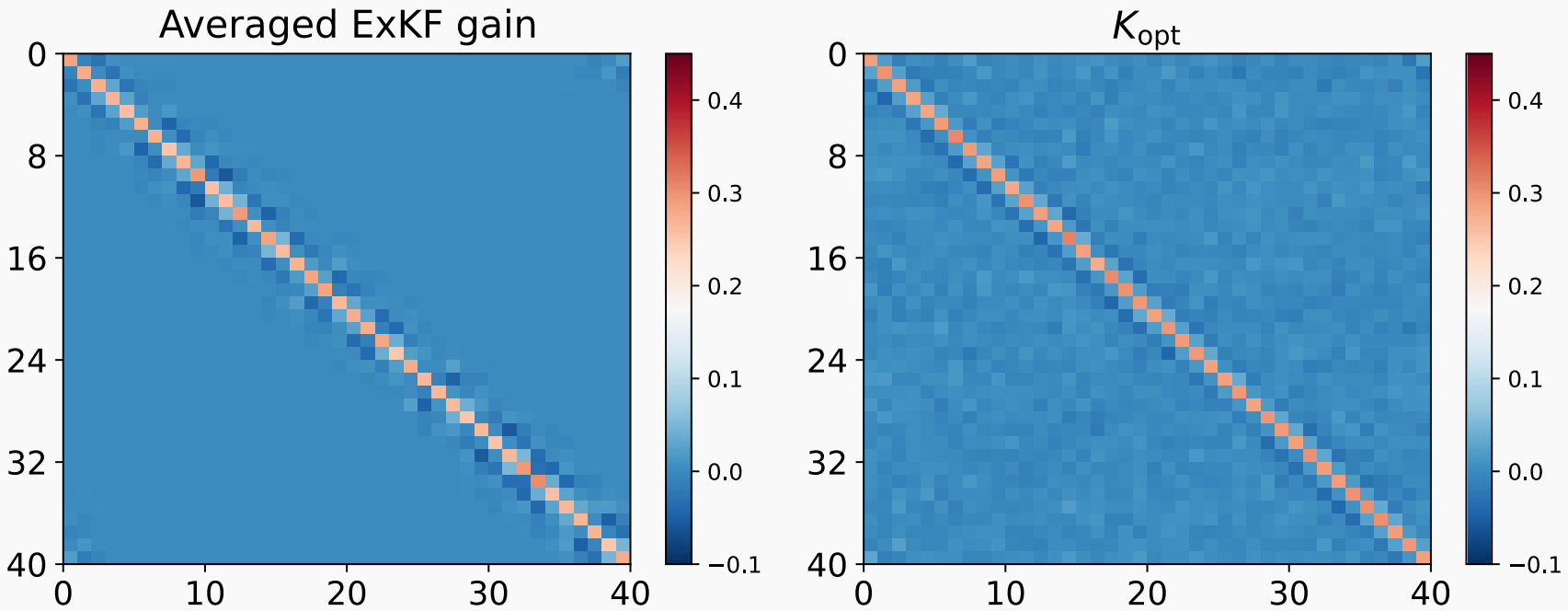
Errors as a function of time step for learning the gain with the online method. The Frobenius norm of the difference between the learned gain and the steady-state gain (red solid line). The KL divergence between the learned filtering distribution and the Kalman filter distribution (green dashed line). The mean-square prediction error in the filter mean compared to the true trajectory (blue solid line).

Gain learning

We now learn a gain matrix for a chaotic nonlinear dynamical system, Lorenz ’96.

Gain learning

We now learn a gain matrix for a chaotic nonlinear dynamical system, Lorenz '96.



Gains for the Lorenz '96 system. On the left is the extended Kalman filter (ExKF) gain, averaged over 50 iterations. On the right is the learned gain.

Learning inflation and localization

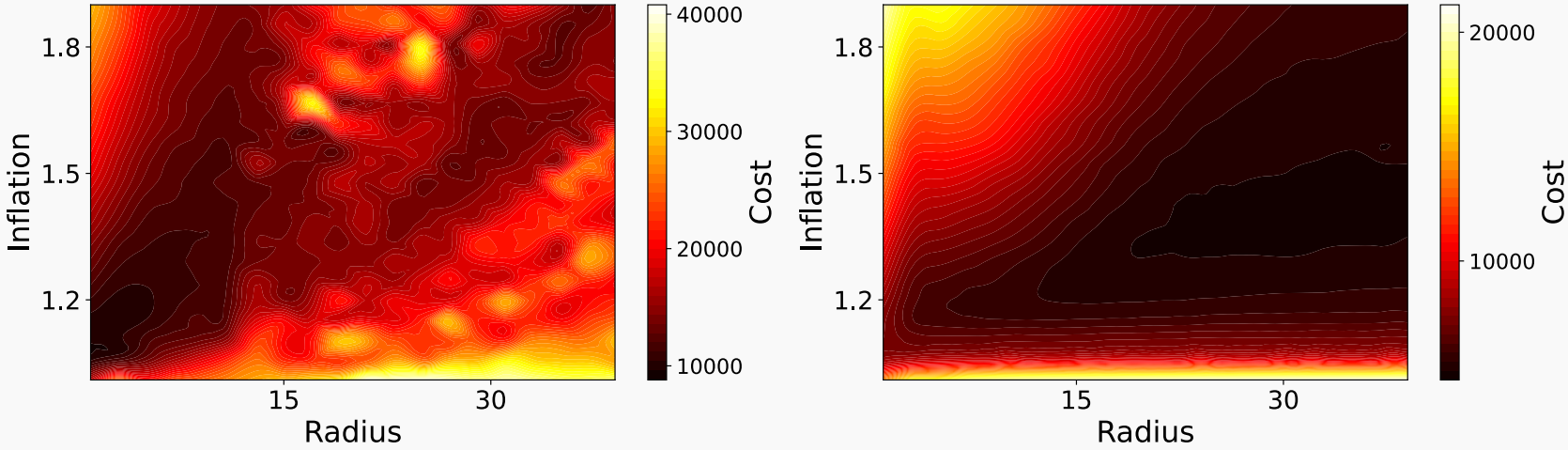
Due to sampling error, empirical covariance matrices need to be inflated and localized to be used in EnKFs, involving tuning parameters.

Learning inflation and localization

Due to sampling error, empirical covariance matrices need to be inflated and localized to be used in EnKFs, involving tuning parameters.

These parameters are sometimes tuned to minimize RMSE in the EnKF mean. Here, we instead measure distance to the true filter (uncertainty quantification, not only accuracy).

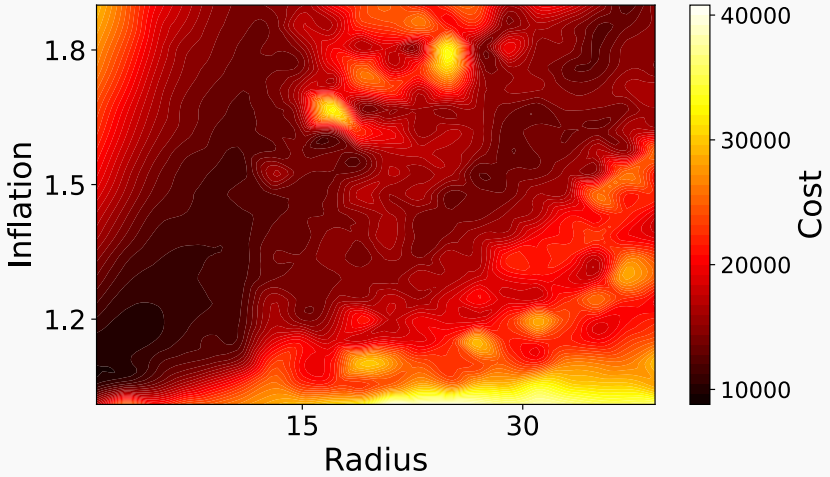
Learning inflation and localization (Lorenz '96)



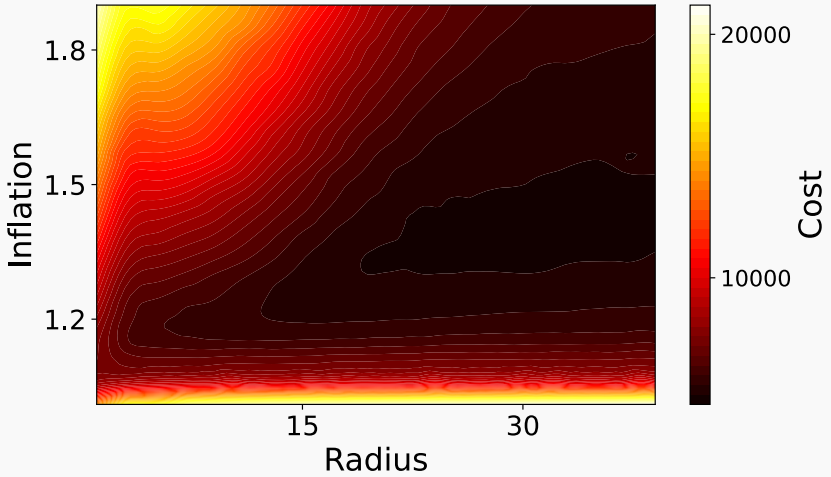
Ensemble size 5 Ensemble size 20

Contours for the cost function as a function of localization radius and inflation.

Learning inflation and localization (Lorenz '96)



Ensemble size 5



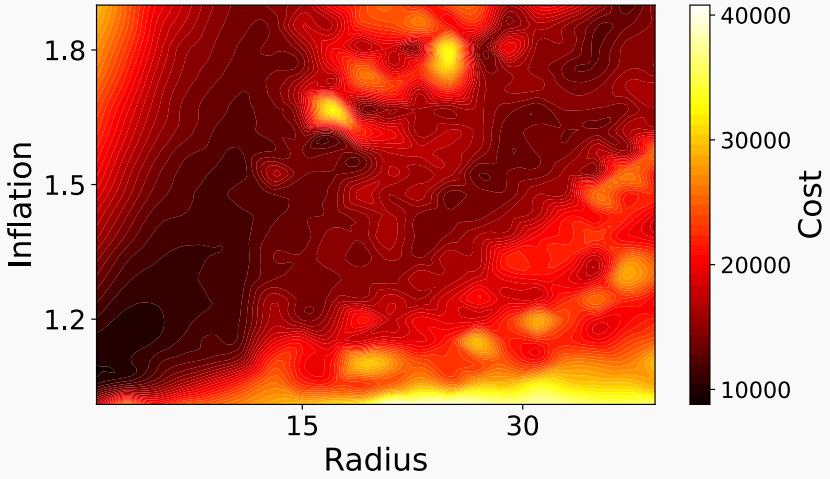
Ensemble size 20

Contours for the cost function as a function of localization radius and inflation.

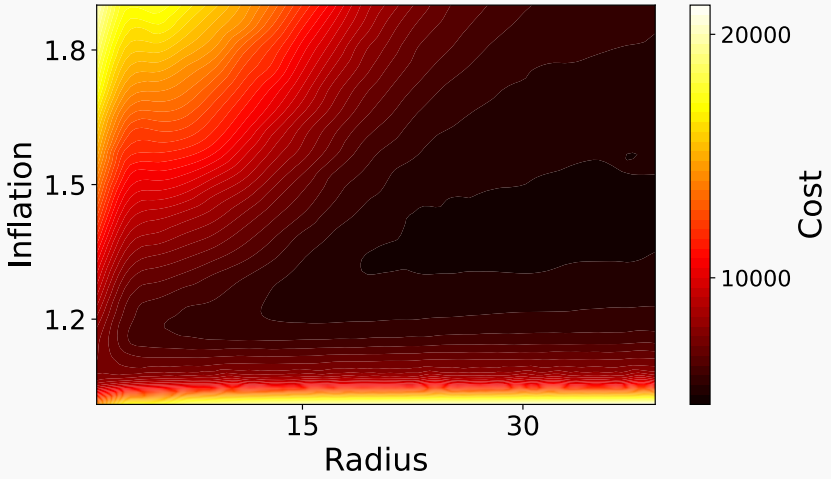
Results:

- Larger localization radius $\rightarrow$ larger optimal inflation.

Learning inflation and localization (Lorenz '96)



Ensemble size 5



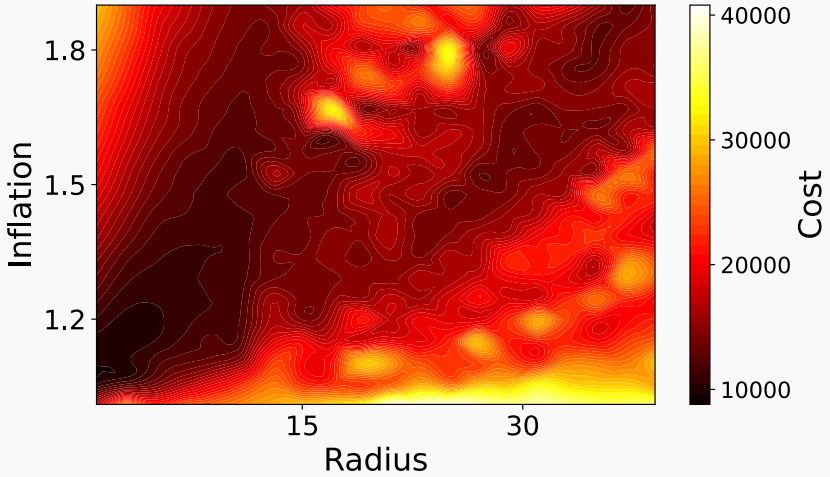
Ensemble size 20

Contours for the cost function as a function of localization radius and inflation.

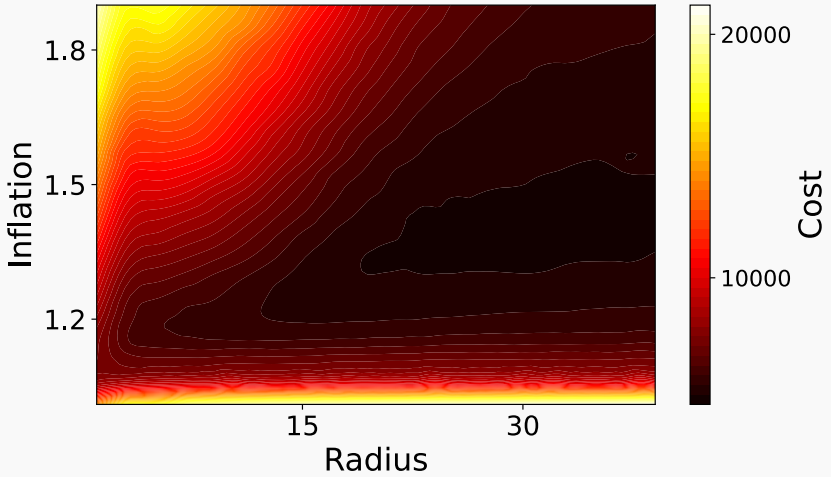
Results:

- Larger localization radius $\rightarrow$ larger optimal inflation.
- Optimal inflation always greater than 1

Learning inflation and localization (Lorenz '96)



Ensemble size 5



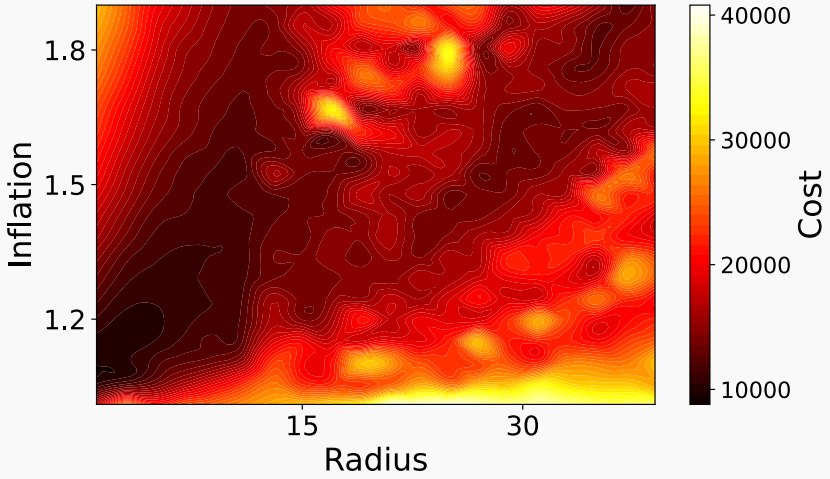
Ensemble size 20

Contours for the cost function as a function of localization radius and inflation.

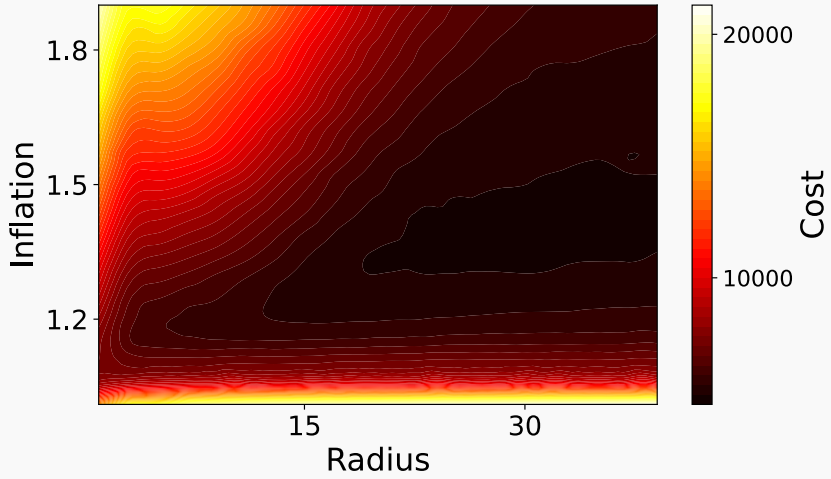
Results:

- Larger localization radius $\rightarrow$ larger optimal inflation.
- Optimal inflation always greater than 1
- Larger ensemble size $\rightarrow$ less severe localization.

Learning inflation and localization (Lorenz '96)



Ensemble size 5



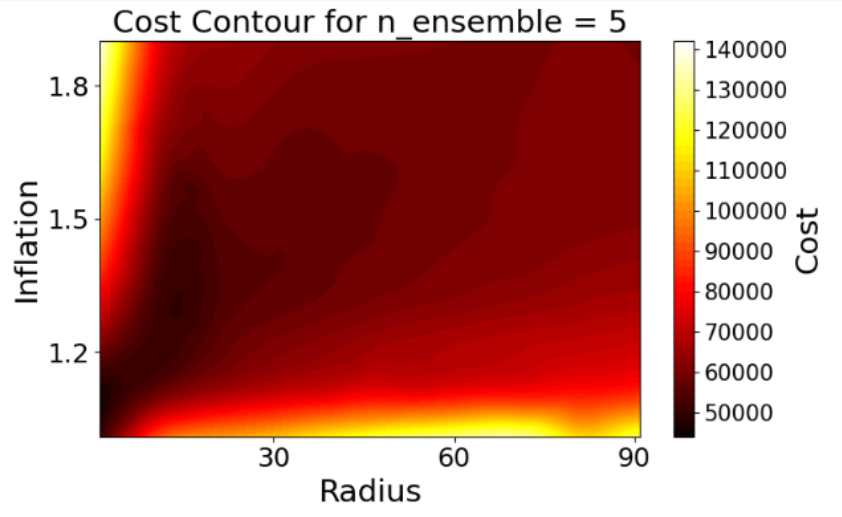
Ensemble size 20

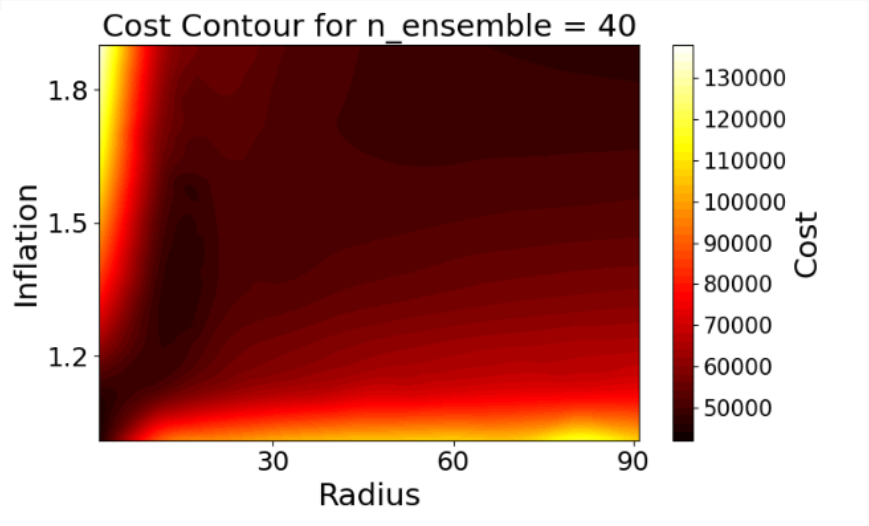
Contours for the cost function as a function of localization radius and inflation.

Results:

- Larger localization radius $\rightarrow$ larger optimal inflation.
- Optimal inflation always greater than 1
- Larger ensemble size $\rightarrow$ less severe localization.
- EnKF gets closer to the true filter with the larger ensemble size.

Learning inflation and localization (Kuramoto–Sivashinsky)

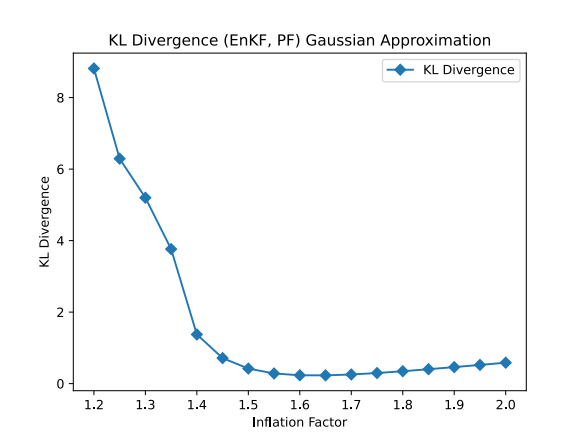




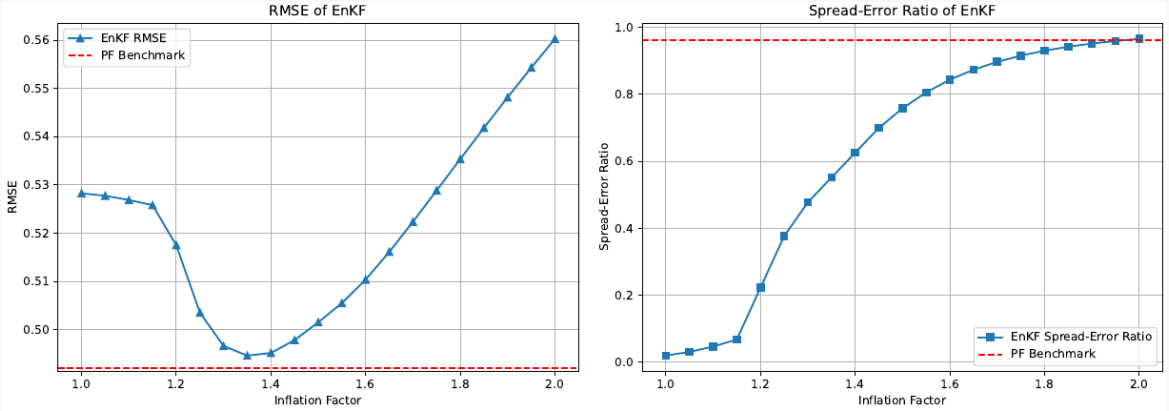
Ensemble size 5 Ensemble size 40

Contours for the cost function as a function of localization radius and inflation.

Comparison to other cost functions (Lorenz '63)



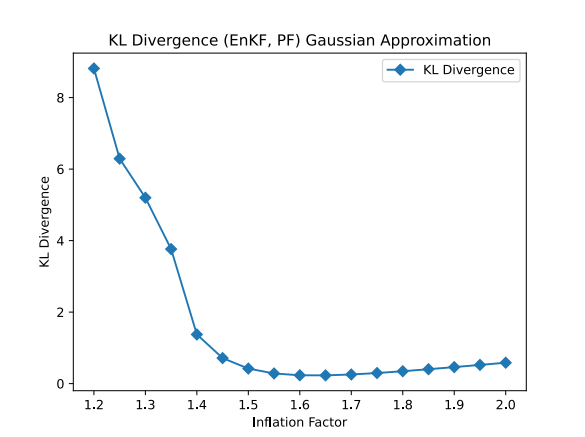
KL to true filter



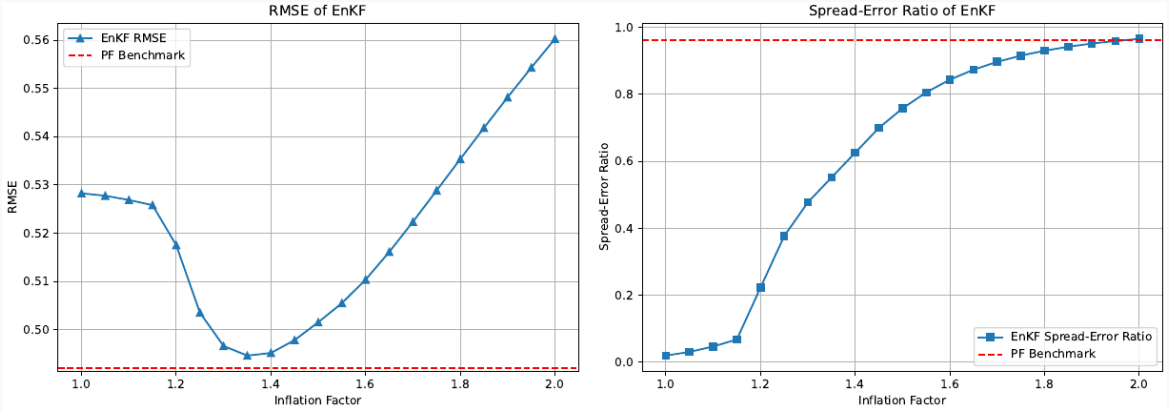
RMSE

Spread–error ratio

Comparison to other cost functions (Lorenz '63)



KL to true filter



RMSE

Spread–error ratio

Variational inference cost function gives different results than RMSE or spread–error!

Conclusions and future work

Variational inference can be used to learn probabilistic filters for DA and set hyperparameters in a Bayesian way.

Conclusions and future work

Variational inference can be used to learn probabilistic filters for DA and set hyperparameters in a Bayesian way.

We show examples applying this method to learning gains for filtering linear and nonlinear dynamical systems, as well as inflation and localization for an EnKF.

Conclusions and future work

Variational inference can be used to learn probabilistic filters for DA and set hyperparameters in a Bayesian way.

We show examples applying this method to learning gains for filtering linear and nonlinear dynamical systems, as well as inflation and localization for an EnKF.

Future work: learning more general analysis steps parameterized by neural networks and scaling to higher-dimensional problems.

Questions?

e.bach@reading.ac.uk

eviatarbach.com

References i

Luk, E., E. Bach, R. Baptista, and A. Stuart (2024). ***Learning Optimal Filters Using Variational Inference.*** arXiv: 2406.18066[cs,math].