Solving Bayesian Inverse Problems via Variational Autoencoders

Hwan Goh, Sheroze Sherriffdeen, Jon Wittmer, Tan Bui-Thanh

The Oden Institute for Computational Engineering and Sciences
The University of Texas at Austin

May 23, 2021

Overview

1. Inverse Problems Introduction

2. Flexible, Adaptive Framework for Rapid Data-Driven Uncertainty Quantification

3. Results

4. Conclusion

Introduction - Inverse Problems

- Suppose we have a physical system with *states* y and *parameters* u.
- Whilst forward problems challenge us to find the observation data y_{obs} given the parameters u, inverse problems challenge us to find the parameters u given the observeration data y_{obs}.
- An example is the heat equation with the state being temperature and parameters being the heat conductivity

$$-\nabla \cdot u \nabla y = 0 \qquad \text{in } \Omega \tag{1a}$$

$$-u(\nabla y \cdot \hat{\mathbf{n}}) = \mathrm{Bi}\, y \qquad \text{on } \Omega^{\text{ext}} \setminus \Omega^{\text{root}} \tag{1b}$$

$$-u(\nabla y \cdot \hat{\mathbf{n}}) = -1 \qquad \text{on } \Omega^{\text{root}} \tag{1c}$$

where u denotes the thermal heat conductivity, Bi is the Biot number, Ω is the physical domain and Ω^{root} is the bottom edge of the domain, Ω^{ext} is the exterior edges of the domain.

Steady-State Heat Equation Parameters and State

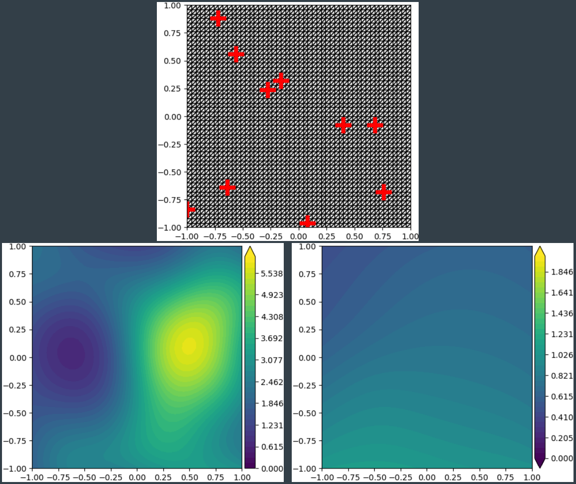


Figure: Top: mesh and sensor distribution. Bottom left: parameter distribution. Bottom right: state distribution.

Solution Process

- Often, we solve for the parameter by minimizing a functional

$$\min_{\mathbf{u}} \|\mathbf{y}_{\text{obs}} - \mathcal{F}(\mathbf{u})\|_2^2 + \mathcal{R}(\mathbf{u}) \tag{2}$$

where $\mathcal{F}$ is the parameter-to-observable (PtO) map, $\mathcal{R}$ is some regularization functional and $\mathbf{u}$ is the parameter-of-interest (PoI).

- What if we try and learn a parameterized inverse problem solver Ψ through optimizing

$$\min_{\boldsymbol{W}} \frac{1}{M} \sum_{m=1}^{M} \left\| \mathbf{u}^{(m)} - \Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right) \right\|_2^2 + \mathcal{R}(\boldsymbol{W}). \tag{3}$$

using a dataset of parameter and observation pairs $\left\{\left(\mathbf{u}^{(m)}, \mathbf{y}_{\text{obs}}^{(m)}\right)\right\}_{m=1}^{M}$.

Proposed Regularization

Instead of regularizing the weights of the network directly, one possible approach is to regularize the output of the network.

$$\min_{\boldsymbol{W}} \frac{1}{M} \sum_{m=1}^{M} \left\| \mathbf{u}^{(m)} - \Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right) \right\|_2^2 + \left\| \mathbf{y}_{\text{obs}}^{(m)} - \mathcal{F}\left(\Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right)\right) \right\|_2^2. \tag{4}$$

Proposed Regularization

Instead of regularizing the weights of the network directly, one possible approach is to regularize the output of the network.

$$\min_{\boldsymbol{W}} \frac{1}{M} \sum_{m=1}^{M} \left\| \mathbf{u}^{(m)} - \Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right) \right\|_2^2 + \left\| \mathbf{y}_{\text{obs}}^{(m)} - \mathcal{F}\left(\Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right)\right) \right\|_2^2. \tag{4}$$

$$\min_{\boldsymbol{W}} \frac{1}{M} \sum_{m=1}^{M} \left\| \mathbf{u}^{(m)} - \Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right) \right\|_2^2 + \left\| \mathcal{M}\left(\mathbf{y}_{\text{obs}}^{(m)} - \mathcal{F}\left(\Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right)\right)\right) \right\|_2^2 \tag{5}$$

where $\mathcal{M}$ is some noise regularization operator.

Proposed Regularization

Instead of regularizing the weights of the network directly, one possible approach is to regularize the output of the network.

$$\min_{\boldsymbol{W}} \frac{1}{M} \sum_{m=1}^{M} \left\| \mathbf{u}^{(m)} - \Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right) \right\|_2^2 + \left\| \mathbf{y}_{\text{obs}}^{(m)} - \mathcal{F}\left(\Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right)\right) \right\|_2^2. \tag{4}$$

$$\min_{\boldsymbol{W}} \frac{1}{M} \sum_{m=1}^{M} \left\| \mathbf{u}^{(m)} - \Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right) \right\|_2^2 + \left\| \mathcal{M}\left(\mathbf{y}_{\text{obs}}^{(m)} - \mathcal{F}\left(\Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right)\right)\right) \right\|_2^2 \tag{5}$$

where $\mathcal{M}$ is some noise regularization operator.

$$\begin{aligned} \min_{\boldsymbol{W}} \frac{1}{M} \sum_{m=1}^{M} \left\| \mathbf{u}^{(m)} - \Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right) \right\|_2^2 + \left\| \mathcal{M}\left(\mathbf{y}_{\text{obs}}^{(m)} - \mathcal{F}\left(\Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right)\right)\right) \right\|_2^2 && \text{(6a)} \\ + \left\| \mathcal{P}\left(\Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right)\right) \right\|_2^2 && \text{(6b)} \end{aligned}$$

where $\mathcal{P}$ is a map encoding prior information.

Motivation for Uncertainty Quantification

- Training a neural network through the optimization problem (6) yields a learned inverse problems solver that outputs a point estimate of our PoI.
- As it is, this deterministic solver is unable to provide information about the accuracy of the estimate. It would be more ideal to have a probabilistic interpretation of our learned solver that facilitates uncertainty quantification.
- With this in mind, we are motivated to view inverse problems under the framework of Bayesian statisics. In this setting, we instead work towards a solver for Bayesian inverse problems which, in turn, allows us to formally establish the regularization terms in (6).

Bayesian Inverse Problems

- Under the statistical framework, the PoI of an inverse problem is considered to be a random variable instead of an unknown value.
- Consequently, the solution of the statistical inverse problem is a probability distribution instead of a single estimated value.
- The statistical framework attempts to remove the ill-posedness of inverse problems by restating the inverse problem as a well-posed extension in a larger space of probability distributions

Bayesian Inverse Problems

- With the assumption that our data is corrupted by additive noise, we consider the following observational model

$$Y = \mathcal{F}(U) + E \tag{7}$$

- 'given the observed data $\mathbf{y}_{\text{obs}}$, what is the distribution of the PoI U responsible for our measurement?'. Therefore, the conditional density $p_{U|Y}(\mathbf{u}|Y = \mathbf{y}_{\text{obs}})$ is the solution to the statistical parameter estimation problem under the Bayesian framework.

Posterior Distribution

- To approximate this conditional density, we utilize Bayes' Theorem to form a model of $p_{U|Y}\left(\mathbf{u}|Y=\mathbf{y}_{\text{obs}}\right)$ called the posterior distribution which we denote as p_{post}:

$$p_{\text{post}}\left(\mathbf{u}|\mathbf{y}_{\text{obs}}\right) \propto p_{\text{lkhd}}\left(\mathbf{y}_{\text{obs}}|\mathbf{u}\right) p_{\text{pr}}\left(\mathbf{u}\right). \tag{8}$$

  where p_{lkhd} is the likelihood model and p_{pr} is the prior model.
- This challenges us with the completion of three tasks:
  1. construct the likelihood model p_{lkhd} that expresses the interrelation between the data and the unknown,
  2. using prior information we may possess about the unknown $\mathbf{u}$, construct a prior probability density p_{pr} that expresses this information,
  3. develop methods which extract meaningful information from the posterior probability density p_{post}.

Posterior Distribution

To address these three tasks, two assumptions are often made:

1. The first assumption supposes that the noise E is mutually independent with respect to our parameter of interest U. Then, using our observation model (7) and marginalization of the noise E, we obtain the following likelihood model:

$$p_{\mathrm{lkhd}} = p_E\left(\mathbf{y}_{\mathrm{obs}} - \mathcal{F}\left(\mathbf{u}\right)\right). \tag{9}$$

2. The second assumption supposes that all are random variables are Gaussian. That is, $\mathcal{N}(\boldsymbol{\mu}_E, \boldsymbol{\Gamma}_E)$ and $\mathcal{N}(\boldsymbol{\mu}_{\mathrm{pr}}, \boldsymbol{\Gamma}_{\mathrm{pr}})$. With this, our posterior model becomes

$$p_{\mathrm{post}}(\mathbf{u}|\mathbf{y}_{\mathrm{obs}}) \propto p_E(\mathbf{y}_{\mathrm{obs}} - \mathcal{F}(\mathbf{u}))p_{\mathrm{pr}}(\mathbf{u}) \tag{10a}$$

$$= \exp\left(-\frac{1}{2}\left(\left\|\mathbf{y}_{\mathrm{obs}} - \mathcal{F}(\mathbf{u}) - \boldsymbol{\mu}_E\right\|^2_{\boldsymbol{\Gamma}_E^{-1}} + \left\|\mathbf{u} - \boldsymbol{\mu}_{\mathrm{pr}}\right\|^2_{\boldsymbol{\Gamma}_{\mathrm{pr}}^{-1}}\right)\right) \tag{10b}$$

Posterior Distribution

Recall:

$$\min_{\boldsymbol{W}} \frac{1}{M} \sum_{m=1}^{M} \left\| \mathbf{u}^{(m)} - \Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right) \right\|_2^2 + \left\| \mathcal{M}\left(\mathbf{y}_{\text{obs}}^{(m)} - \mathcal{F}\left(\Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right)\right)\right) \right\|_2^2 \tag{11a}$$

$$+ \left\| \mathcal{P}\left(\Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W}\right)\right) \right\|_2^2 \tag{11b}$$

The functionals

$$- \left\| \mathbf{y}_{\text{obs}} - \mathcal{F}(\mathbf{u}) - \boldsymbol{\mu}_E \right\|_{\boldsymbol{\Gamma}_E^{-1}}^2 + \left\| \mathbf{u} - \boldsymbol{\mu}_{\text{pr}} \right\|_{\boldsymbol{\Gamma}_{\text{pr}}^{-1}}^2 \tag{12}$$

look like a good candidate for the regularization terms. We now formalize the inclusion of these terms.

Notion of Distance

- Let $p(\mathbf{u}|\mathbf{y})$ denote the target posterior density we wish to estimate and let $q_{\boldsymbol{\phi}}(\mathbf{u}|\mathbf{y})$ denote our model of the target density parameterized by $\boldsymbol{\phi}$.
- To optimize for the parameters $\boldsymbol{\phi}$, we require some notion of distance between our model posterior and target posterior.
- In our work, we elect to use the following family of Jensen-Shannon divergences (JSD) [Nielsen(2010)]:

$$\mathrm{JS}_{\alpha}(q||p) = \alpha \mathrm{KL}(q||(1-\alpha)q + \alpha p) + (1-\alpha)\mathrm{KL}(p||(1-\alpha)q + \alpha p) \tag{13}$$

JSD Family

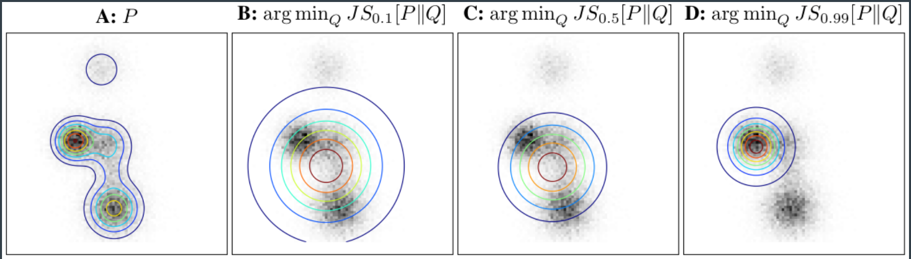


Figure: Illustrating the behaviour of the JS divergence family for the case of under model underspecification for the range of values of α. This figure was obtained from [Huszár(2015)] A: the target data. For images B, C and D, an unimodal Gaussian is used to approximate the target data where the contours show level sets of the approximating distribution q. B: $\alpha = 0.1$. C: $\alpha = 0.5$. D: $\alpha = 0.9$.

Main Theorem

Theorem

Let $\alpha \in (0,1)$. *Then*

$$\frac{1}{\alpha}\mathrm{JS}_\alpha(q_{\boldsymbol{\phi}}(\boldsymbol{u}|\boldsymbol{y})||p(\boldsymbol{u}|\boldsymbol{y})) = -\,\mathbb{E}_{\boldsymbol{u}\sim q_{\boldsymbol{\phi}}}\left[\log\left(\alpha + \frac{(1-\alpha)q_{\boldsymbol{\phi}}(\boldsymbol{u}|\boldsymbol{y})}{p(\boldsymbol{u}|\boldsymbol{y})}\right)\right] \tag{14a}$$

$$+\log(p(\boldsymbol{y})) \tag{14b}$$

$$-\,L_{\mathrm{JS}}(\boldsymbol{\phi},\boldsymbol{y}) \tag{14c}$$

where

$$L_{\mathrm{JS}}(\boldsymbol{\phi},\boldsymbol{y}) = \frac{1-\alpha}{\alpha}\mathbb{E}_{\boldsymbol{u}\sim p(\boldsymbol{u}|\boldsymbol{y})}\left[\log\left(\alpha + \frac{(1-\alpha)q_{\boldsymbol{\phi}}(\boldsymbol{u}|\boldsymbol{y})}{p(\boldsymbol{u}|\boldsymbol{y})}\right)\right] + \mathbb{E}_{\boldsymbol{u}\sim q_{\boldsymbol{\phi}}}\left[\log\left(\frac{p(\boldsymbol{y},\boldsymbol{u})}{q_{\boldsymbol{\phi}}(\boldsymbol{u}|\boldsymbol{y})}\right)\right]. \tag{15}$$

Insightful Corollary

Corollary

Let $\alpha \in (0,1)$ *and consider again (14). Equation (14) is bounded above such that:*

$$
\begin{aligned}
\frac{1}{\alpha}\mathrm{JS}_{\alpha}(q_{\boldsymbol{\phi}}(\boldsymbol{u}|\boldsymbol{y})||p(\boldsymbol{u}|\boldsymbol{y})) \leq & - \mathrm{KL}(q_{\boldsymbol{\phi}}(\boldsymbol{u}|\boldsymbol{y})||p(\boldsymbol{u}|\boldsymbol{y})) && \text{(16a)}\\
& + \log(p(\boldsymbol{y})) - \log(1-\alpha) - \frac{(1-\alpha)\log(1-\alpha)}{\alpha} && \text{(16b)}\\
& + \frac{1-\alpha}{\alpha}\mathrm{KL}(p(\boldsymbol{u}|\boldsymbol{y})||q_{\boldsymbol{\phi}}(\boldsymbol{u}|\boldsymbol{y})) && \text{(16c)}\\
& - \mathbb{E}_{\boldsymbol{u}\sim q_{\boldsymbol{\phi}}}\left[\log\left(p(\boldsymbol{y}|\boldsymbol{u})\right] + \mathrm{KL}\left(q_{\boldsymbol{\phi}}(\boldsymbol{u}|\boldsymbol{y})|p(\boldsymbol{u})\right)\right. . && \text{(16d)}
\end{aligned}
$$

In particular, we have that

$$
\begin{aligned}
-L_{\mathrm{JS}}(\boldsymbol{\phi},\boldsymbol{y}) \leq & -\frac{(1-\alpha)\log(1-\alpha)}{\alpha} + \frac{1-\alpha}{\alpha}\mathrm{KL}(p(\boldsymbol{u}|\boldsymbol{y})||q_{\boldsymbol{\phi}}(\boldsymbol{u}|\boldsymbol{y})) && \text{(17a)}\\
& - \mathbb{E}_{\boldsymbol{u}\sim q_{\boldsymbol{\phi}}}\left[\log\left(p(\boldsymbol{y}|\boldsymbol{u})\right)\right] + \mathrm{KL}\left(q_{\boldsymbol{\phi}}(\boldsymbol{u}|\boldsymbol{y})||p(\boldsymbol{u})\right) . && \text{(17b)}
\end{aligned}
$$

Key Point

The significance of Corollary 2 is that minimization of

$$\frac{1-\alpha}{\alpha}\mathrm{KL}(p(\mathbf{u}|\mathbf{y})||q_{\boldsymbol{\phi}}(\mathbf{u}|\mathbf{y})) - \mathbb{E}_{\mathbf{u}\sim q_{\boldsymbol{\phi}}}\left[\log\left(p(\mathbf{y}|\mathbf{u})\right)\right] + \mathrm{KL}\left(q_{\boldsymbol{\phi}}(\mathbf{u}|\mathbf{y})||p(\mathbf{u})\right) \tag{18}$$

with respect to $\boldsymbol{\phi}$ minimizes

$$\frac{1}{\alpha}\mathrm{JS}_{\alpha}(q_{\boldsymbol{\phi}}(\mathbf{u}|\mathbf{y})||p(\mathbf{u}|\mathbf{y})) + \mathrm{KL}(q_{\boldsymbol{\phi}}(\mathbf{u}|\mathbf{y})||p(\mathbf{u}|\mathbf{y})) \tag{19}$$

which is exactly the task of variational inference.

Flexibility of our Framework

Note that for

$$\frac{1}{\alpha}\mathrm{JS}_{\alpha}(q_{\phi}(\mathbf{u}|\mathbf{y})||p(\mathbf{u}|\mathbf{y})) + \mathrm{KL}(q_{\phi}(\mathbf{u}|\mathbf{y})||p(\mathbf{u}|\mathbf{y})) \tag{20}$$

- Recalling

  $$\mathrm{JS}_{\alpha}(q||p) = \alpha\mathrm{KL}(q||(1-\alpha)q + \alpha p) + (1-\alpha)\mathrm{KL}(p||(1-\alpha)q + \alpha p) \tag{21}$$

  It is clear that if $\alpha = 1$ then we recover the usual zero-avoiding $\mathrm{KL}(q||p)$. Indeed, as $\alpha \to 1$, then the first term in (18) tends to 0 which recovers the negative of the ELBO.
- In (20), the presence of the KLD term ensures that our model posterior will inherently be zero-avoiding. However, the $\frac{1}{\alpha}$ scaling factor ensures that as $\alpha \to 0$, the consequently zero-forcing JSD dominates the KLD.

Flexibility of our Framework

- Therefore, our UQ-VAE framework essentially retains the full flexibility of the JSD family.
- With only an adjustment of a single scalar value, our framework allows the selection of the notion of distance used by the optimization routine to direct the model posterior towards the target posterior.
- This, in turn, translates to control of the balance of data-fitting and regularization used in the training procedure.

Regularized Optimization Framework

The minimization target:

$$\frac{1-\alpha}{\alpha}\mathrm{KL}\left(p(\mathbf{u}|\mathbf{y})||q_{\phi}(\mathbf{u}|\mathbf{y})\right) - \mathbb{E}_{\mathbf{u}\sim q_{\phi}}\left[\log\left(p(\mathbf{y}|\mathbf{u})\right)\right] + \mathrm{KL}\left(q_{\phi}(\mathbf{u}|\mathbf{y})||p(\mathbf{u})\right) \tag{22}$$

correspond to the three desired terms:

$$\min_{\boldsymbol{W}} \frac{1}{M}\sum_{m=1}^{M}\left\|\mathbf{u}^{(m)} - \Psi\left(\mathbf{y}_{\mathrm{obs}}^{(m)}, \boldsymbol{W}\right)\right\|_2^2 + \left\|\mathcal{M}\left(\mathbf{y}_{\mathrm{obs}}^{(m)} - \mathcal{F}\left(\Psi\left(\mathbf{y}_{\mathrm{obs}}^{(m)}, \boldsymbol{W}\right)\right)\right)\right\|_2^2 \tag{23a}$$

$$+\left\|\mathcal{P}\left(\Psi\left(\mathbf{y}_{\mathrm{obs}}^{(m)}, \boldsymbol{W}\right)\right)\right\|_2^2. \tag{23b}$$

Regularized Optimization Framework

First, notice that

$$
\begin{aligned}
\mathrm{KL}\left(p(\mathbf{u}|\mathbf{y})||q_{\boldsymbol{\phi}}(\mathbf{u}|\mathbf{y})\right) &= \mathbb{E}_{\mathbf{u}\sim p(\mathbf{u}|\mathbf{y})}\left[\log\left(\frac{p(\mathbf{u}|\mathbf{y})}{q_{\boldsymbol{\phi}}(\mathbf{u}|\mathbf{y})}\right)\right] && \text{(24a)}\\
&= \cancel{\mathbb{E}_{\mathbf{u}\sim p(\mathbf{u}|\mathbf{y})}\left[\log\left(p(\mathbf{u}|\mathbf{y})\right)\right]} - \mathbb{E}_{\mathbf{u}\sim p(\mathbf{u}|\mathbf{y})}\left[\log\left(q_{\boldsymbol{\phi}}(\mathbf{u}|\mathbf{y})\right)\right] && \text{(24b)}
\end{aligned}
$$

where the first term can be omitted when optimizing with respect to $\boldsymbol{\phi}$ since it does not have any dependence on $\boldsymbol{\phi}$.

Regularized Optimization Framework

With dataset $\left\{\left(\mathbf{u}^{(m)}, \mathbf{y}_{\text{obs}}^{(m)}\right)\right\}_{m=1}^{M}$, we form a Monte-Carlo approximation

$$\mathbb{E}_{\mathbf{u}\sim p\left(\mathbf{u}\middle|\mathbf{y}_{\text{obs}}^{(m)}\right)}\left[-\log\left(q_\phi\left(\mathbf{u}\middle|\mathbf{y}_{\text{obs}}^{(m)}\right)\right)\right] \approx -\log\left(q_\phi\left(\mathbf{u}^{(m)}\middle|\mathbf{y}_{\text{obs}}^{(m)}\right)\right)$$

and assume a Gaussian model: $q_\phi\left(\mathbf{u}\middle|\mathbf{y}_{\text{obs}}^{(m)}\right) = \mathcal{N}\left(\mathbf{u}\middle|\boldsymbol{\mu}_{\text{post}}^{(m)}, \boldsymbol{\Gamma}_{\text{post}}^{(m)}\right)$ to obtain

$$\frac{D}{2}\log(2\pi) + \frac{1}{2}\log\left|\boldsymbol{\Gamma}_{\text{post}}^{(m)}\right| + \frac{1}{2}\left\|\boldsymbol{\mu}_{\text{post}}^{(m)} - \mathbf{u}^{(m)}\right\|_{\boldsymbol{\Gamma}_{\text{post}}^{(m)-1}}^{2}.$$

Regularized Optimization Framework

For the remaining terms:

$$-\mathbb{E}_{\mathbf{u}\sim q_{\boldsymbol{\phi}}}\left[\log\left(p(\mathbf{y}|\mathbf{u})\right]=\left\|\mathbf{y}_{\text{obs}}^{(m)}-\mathcal{F}\left(\mathbf{u}(\boldsymbol{W})\right)-\boldsymbol{\mu}_E\right\|_{\boldsymbol{\Gamma}_E^{-1}}^2 \qquad (25)$$

and

$$\mathrm{KL}\left(q_{\boldsymbol{\phi}}(\mathbf{u}|\mathbf{y})|p(\mathbf{u})\right)=\mathrm{tr}\left(\boldsymbol{\Gamma}_{\text{pr}}^{-1}\boldsymbol{\Gamma}_{\text{post}}^{(m)}\right)+\left\|\boldsymbol{\mu}_{\text{post}}^{(m)}-\boldsymbol{\mu}_{\text{pr}}\right\|_{\boldsymbol{\Gamma}_{\text{pr}}^{-1}}^2+\log\frac{\left|\boldsymbol{\Gamma}_{\text{pr}}\right|}{\left|\boldsymbol{\Gamma}_{\text{post}}^{(m)}\right|} \qquad (26)$$

Regularized Optimization Framework

With dataset $\left\{\left(\mathbf{u}^{(m)}, \mathbf{y}_{\text{obs}}^{(m)}\right)\right\}_{m=1}^{M}$ we have the following optimization problem:

$$\min_{\boldsymbol{W}} \frac{1}{M} \sum_{m=1}^{M} \frac{1-\alpha}{\alpha} \left(\log \left| \boldsymbol{\Gamma}_{\text{post}}^{(m)} \right| + \left\| \boldsymbol{\mu}_{\text{post}}^{(m)} - \mathbf{u}^{(m)} \right\|_{\boldsymbol{\Gamma}_{\text{post}}^{(m)-1}}^{2} \right) \tag{27a}$$

$$+ \left\| \mathbf{y}_{\text{obs}}^{(m)} - \mathcal{F}\left(\mathbf{u}_{\text{draw}}^{(m)}(\boldsymbol{W}) \right) - \boldsymbol{\mu}_{E} \right\|_{\boldsymbol{\Gamma}_{E}^{-1}}^{2} \tag{27b}$$

$$+ \operatorname{tr}\left(\boldsymbol{\Gamma}_{\text{pr}}^{-1} \boldsymbol{\Gamma}_{\text{post}}^{(m)} \right) + \left\| \boldsymbol{\mu}_{\text{post}}^{(m)} - \boldsymbol{\mu}_{\text{pr}} \right\|_{\boldsymbol{\Gamma}_{\text{pr}}^{-1}}^{2} + \log \frac{\left| \boldsymbol{\Gamma}_{\text{pr}} \right|}{\left| \boldsymbol{\Gamma}_{\text{post}}^{(m)} \right|} \tag{27c}$$

$$\text{where} \quad \left(\boldsymbol{\mu}_{\text{post}}^{(m)}, \boldsymbol{\Gamma}_{\text{post}}^{\frac{1}{2}(m)} \right) = \Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W} \right), \tag{27d}$$

$$\mathbf{u}_{\text{draw}}^{(m)}(\boldsymbol{W}) = \boldsymbol{\mu}_{\text{post}}^{(m)} + \boldsymbol{\Gamma}_{\text{post}}^{\frac{1}{2}(m)} \boldsymbol{\epsilon}^{(m)}, \tag{27e}$$

$$\boldsymbol{\epsilon}^{(m)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_D) \tag{27f}$$

Schematic

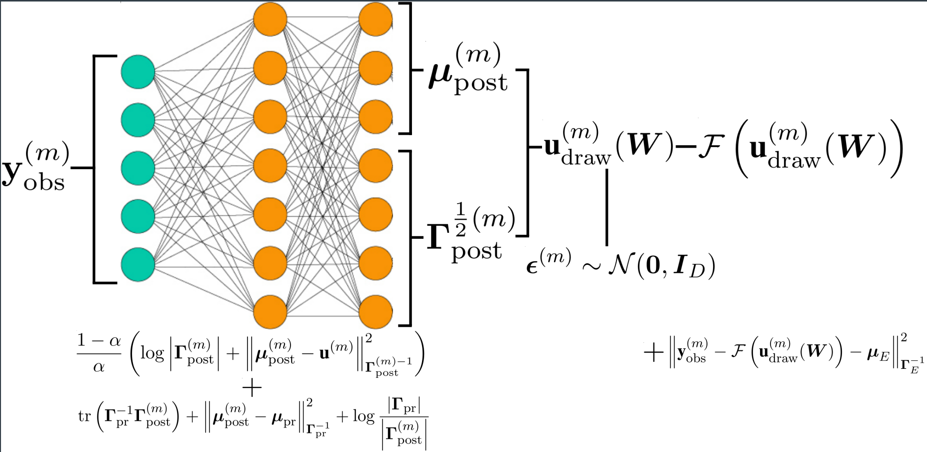


Figure: Schematic of the UQ-VAE framework.

Inherent Adaptive Optimization

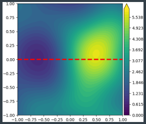
Looking closely at

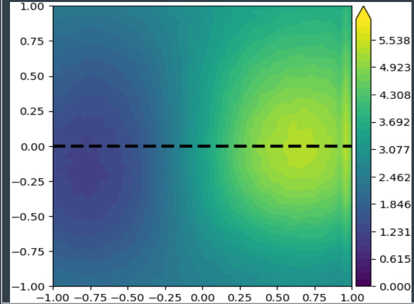
$$\frac{1-\alpha}{\alpha}\left(\log\left|\boldsymbol{\Gamma}_{\text{post}}^{(m)}\right| + \left\|\boldsymbol{\mu}_{\text{post}}^{(m)} - \mathbf{u}^{(m)}\right\|^2_{\boldsymbol{\Gamma}_{\text{post}}^{(m)-1}}\right) \tag{28}$$

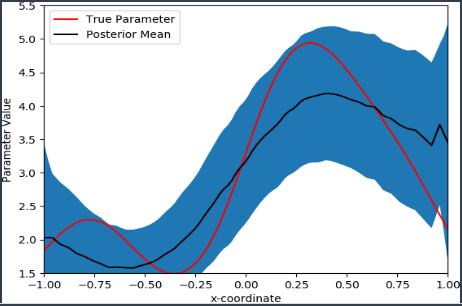
we can see that the presence of the matrix $\boldsymbol{\Gamma}_{\text{post}}^{(m)-1}$ in the weighted norm of (28) acts as an adaptive penalty for the data-misfit term. With this in mind, we make the following observations about the two terms in (28):

- Since $\lim_{\alpha\to 0}\frac{1-\alpha}{\alpha} = \infty$, then a choice of $\alpha \approx 0$ emphasizes the $\log\left|\boldsymbol{\Gamma}_{\text{post}}^{(m)}\right|$ term. This causes a preference for a small posterior variance which, in turn, creates a large penalization of the data-misfit term by the inverse of the posterior covariance.
- In contrast, a choice of $\alpha \approx 0$ relieves the requirement of a small posterior variance to promote the influence of the PoI data on the optimization problem.

Inherent Adaptive Optimization







Analytical Result

Consider a Gaussian prior model $\mathcal{N}\left(\boldsymbol{\mu}_{\mathrm{pr}}, \boldsymbol{\Gamma}_{\mathrm{pr}}\right)$ and Gaussian noise model $\mathcal{N}\left(\boldsymbol{\mu}_E, \boldsymbol{\Gamma}_E\right)$. Suppose the target posterior $p(\mathbf{u}|\mathbf{y}_{\mathrm{obs}}) \sim \mathcal{N}\left(\boldsymbol{\mu}_{\mathrm{true}}, \boldsymbol{\Gamma}_{\mathrm{true}}\right)$ is such that

$$\boldsymbol{\Gamma}_{\mathrm{true}} = \left(\boldsymbol{F}^{\mathrm{T}} \boldsymbol{\Gamma}_E^{-1} \boldsymbol{F} + \boldsymbol{\Gamma}_{\mathrm{pr}}^{-1}\right)^{-1} \tag{29a}$$

$$\boldsymbol{\mu}_{\mathrm{true}} = \boldsymbol{\Gamma}_{\mathrm{true}} \left(\boldsymbol{F}^{\mathrm{T}} \boldsymbol{\Gamma}_E^{-1} \left(\mathbf{y}_{\mathrm{obs}} - \boldsymbol{\mu}_E\right) + \boldsymbol{\Gamma}_{\mathrm{pr}}^{-1} \boldsymbol{\mu}_{\mathrm{pr}}\right) \tag{29b}$$

where $\boldsymbol{F}$ is a matrix.

Analytical Result

Theorem

Suppose the model posterior $q_{\boldsymbol{\phi}}(\boldsymbol{u}|\boldsymbol{y}_{\text{obs}}) \sim \mathcal{N}\left(\boldsymbol{\mu}_{\text{post}}, \boldsymbol{\Gamma}_{\text{post}}\right)$ *is such that*

$$\boldsymbol{\mu}_{\text{post}} = \boldsymbol{W}_{\boldsymbol{\mu}}\boldsymbol{y}_{\text{obs}} + \boldsymbol{b}_{\boldsymbol{\mu}} \tag{30a}$$

$$\boldsymbol{\Gamma}_{\text{post}}^{\frac{1}{2}} = \boldsymbol{L} \odot \boldsymbol{L}_{\mathbb{1}} + \operatorname{diag}(\boldsymbol{\sigma}) \tag{30b}$$

$$\log(\boldsymbol{\sigma}) = \boldsymbol{W}_{\boldsymbol{\sigma}}\boldsymbol{y}_{\text{obs}} + \boldsymbol{b}_{\boldsymbol{\sigma}} \tag{30c}$$

$$\operatorname{vec}(\boldsymbol{L}) = \boldsymbol{W}_{\boldsymbol{L}}\boldsymbol{y}_{\text{obs}} + \boldsymbol{b}_{\boldsymbol{L}} \tag{30d}$$

where $\boldsymbol{L}_{\mathbb{1}}$ *is a lower triangular matrix of ones with zeros on the diagonal. Let* $\alpha = \frac{1}{2}$. *Then the optimization problem*

$$\min_{\boldsymbol{W}_{\boldsymbol{\mu}},\boldsymbol{b}_{\boldsymbol{\mu}},\boldsymbol{W}_{\boldsymbol{\sigma}},\boldsymbol{b}_{\boldsymbol{\sigma}},\boldsymbol{W}_{\boldsymbol{L}},\boldsymbol{b}_{\boldsymbol{L}}} \frac{1-\alpha}{\alpha}\left(\log|\boldsymbol{\Gamma}_{\text{post}}| + \operatorname{tr}\left(\boldsymbol{\Gamma}_{\text{post}}^{-1}\boldsymbol{\Gamma}_{\text{true}}\right) + \left\|\boldsymbol{\mu}_{\text{post}} - \boldsymbol{\mu}_{\text{true}}\right\|_{\boldsymbol{\Gamma}_{\text{post}}^{-1}}^2\right) \tag{31a}$$

$$+ \operatorname{tr}\left(\boldsymbol{\Gamma}_{E}^{-1}\boldsymbol{F}\boldsymbol{\Gamma}_{\text{post}}\boldsymbol{F}^{\mathrm{T}}\right) + \left\|\boldsymbol{y}_{\text{obs}} - \boldsymbol{F}\boldsymbol{\mu}_{\text{post}} - \boldsymbol{\mu}_{E}\right\|_{\boldsymbol{\Gamma}_{E}^{-1}}^2 \tag{31b}$$

$$+ \operatorname{tr}\left(\boldsymbol{\Gamma}_{\text{pr}}^{-1}\boldsymbol{\Gamma}_{\text{post}}\right) + \left\|\boldsymbol{\mu}_{\text{post}} - \boldsymbol{\mu}_{\text{pr}}\right\|_{\boldsymbol{\Gamma}_{\text{pr}}^{-1}}^2 + \log\frac{|\boldsymbol{\Gamma}_{\text{pr}}|}{|\boldsymbol{\Gamma}_{\text{post}}|} \tag{31c}$$

achieves its minimum if and only if $\boldsymbol{W}_{\boldsymbol{\mu}}, \boldsymbol{b}_{\boldsymbol{\mu}}, \boldsymbol{W}_{\boldsymbol{\sigma}}, \boldsymbol{b}_{\boldsymbol{\sigma}}, \boldsymbol{W}_{\boldsymbol{L}}, \boldsymbol{b}_{\boldsymbol{L}}$ *are such that* $\boldsymbol{\mu}_{\text{post}} = \boldsymbol{\mu}_{\text{true}}$ *and* $\boldsymbol{\Gamma}_{\text{post}} = \boldsymbol{\Gamma}_{\text{true}}$.

Learning the PtO

With dataset $\left\{\left(\mathbf{u}^{(m)}, \mathbf{y}_{\text{obs}}^{(m)}\right)\right\}_{m=1}^{M}$ we have the following optimization problem:

$$\min_{\boldsymbol{W}, \boldsymbol{W}_{\text{d}}} \frac{1}{M} \sum_{m=1}^{M} \frac{1-\alpha}{\alpha} \left(\log \left| \boldsymbol{\Gamma}_{\text{post}}^{(m)} \right| + \left\| \boldsymbol{\mu}_{\text{post}}^{(m)} - \mathbf{u}^{(m)} \right\|_{\boldsymbol{\Gamma}_{\text{post}}^{(m)-1}}^{2} \right) \tag{32a}$$

$$+ \left\| \mathbf{y}_{\text{obs}}^{(m)} - \Psi_{\text{d}} \left(\mathbf{u}_{\text{draw}}^{(m)}(\boldsymbol{W}), \boldsymbol{W}_{\text{d}} \right) - \boldsymbol{\mu}_{E} \right\|_{\boldsymbol{\Gamma}_{E}^{-1}}^{2} \tag{32b}$$

$$+ \operatorname{tr}\left(\boldsymbol{\Gamma}_{\text{pr}}^{-1} \boldsymbol{\Gamma}_{\text{post}}^{(m)} \right) + \left\| \boldsymbol{\mu}_{\text{post}}^{(m)} - \boldsymbol{\mu}_{\text{pr}} \right\|_{\boldsymbol{\Gamma}_{\text{pr}}^{-1}}^{2} + \log \frac{\left| \boldsymbol{\Gamma}_{\text{pr}} \right|}{\left| \boldsymbol{\Gamma}_{\text{post}}^{(m)} \right|} \tag{32c}$$

$$\text{where} \quad \left(\boldsymbol{\mu}_{\text{post}}^{(m)}, \boldsymbol{\Gamma}_{\text{post}}^{\frac{1}{2}(m)} \right) = \Psi\left(\mathbf{y}_{\text{obs}}^{(m)}, \boldsymbol{W} \right), \tag{32d}$$

$$\mathbf{u}_{\text{draw}}^{(m)}(\boldsymbol{W}) = \boldsymbol{\mu}_{\text{post}}^{(m)} + \boldsymbol{\Gamma}_{\text{post}}^{\frac{1}{2}(m)} \boldsymbol{\epsilon}^{(m)}, \tag{32e}$$

$$\boldsymbol{\epsilon}^{(m)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{D}) \tag{32f}$$

Schematic

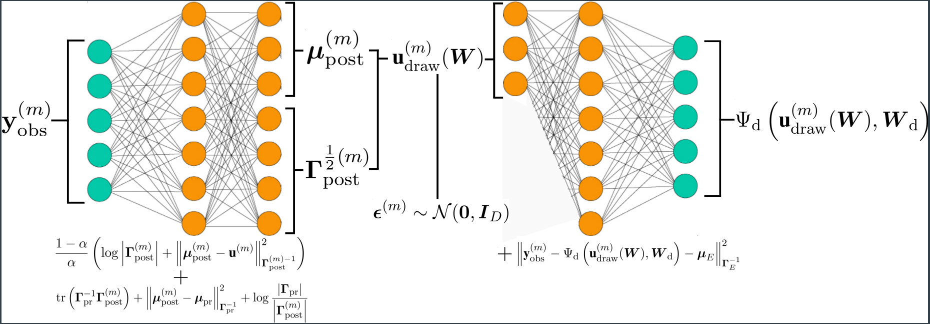


Figure: Schematic of the UQ-VAE framework where the PtO is learned.

Computational Cost

Problem dimension is 2601 degrees of freedom.
Training Cost:

- Modelled PtO map $\mathcal{F}$ (CPU):
  4.5 seconds per batch on a dual-socket node with two Intel Xeon E5-2690 CPUs for a total of 24 cores.
  Approximately one day for 400 epochs of batch-size 100 when $M = 5000$
- Learned PtO map Ψ_{d} (GPU):
  0.35 seconds per batch on a NVidia 1080-TI GPU.
  Approximately 10 minutes for 400 epochs of batch-size 100 when $M = 5000$

Inference Cost:

- In total, it takes on average 110 seconds to form the Laplace approximation.
- Forming the model posterior by propagation through a trained neural network takes, on average, 0.04 seconds; more than 2750 times faster.

Results: 0% Noise, $M = 50$

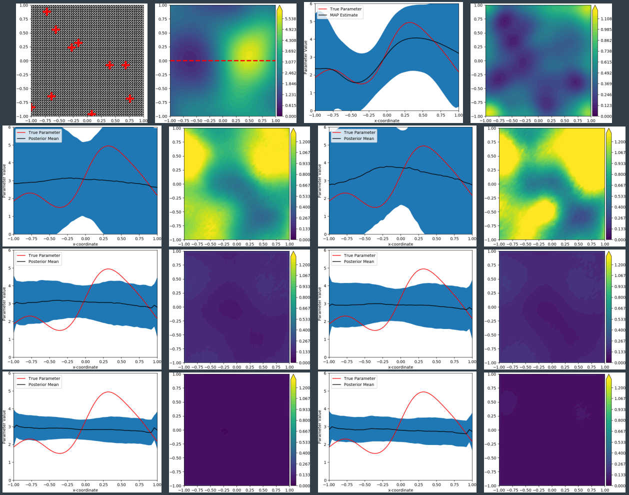


Figure: Top row left to right: mesh with sensors denoted with a red cross, true PoI, cross-sectional uncertainty estimate and pointwise posterior variance from Laplace approximation. Second to fourth rows: $\alpha = 0.00001, 0.1, 0.5$. First and third columns: cross-sectional uncertainty estimates. Second and fourth columns: approximate pointwise posterior variance. First and second columns: modelled PtO map. Third and fourth columns: learned PtO map.

Results: 0% Noise, $M = 500$

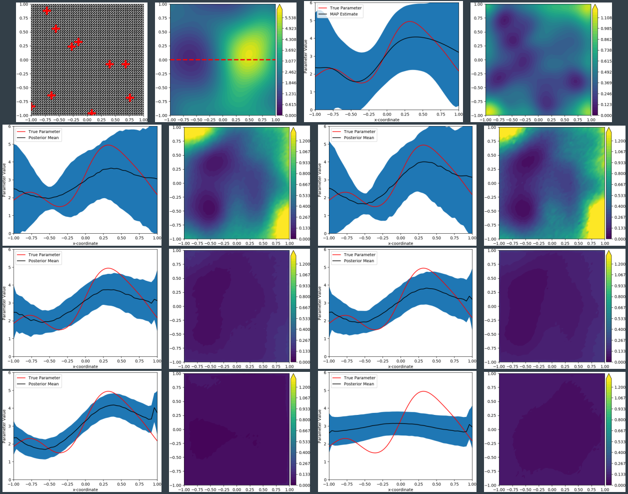


Figure: Top row left to right: mesh with sensors denoted with a red cross, true PoI, cross-sectional uncertainty estimate and pointwise posterior variance from Laplace approximation. Second to fourth rows: $\alpha = 0.00001, 0.1, 0.5$. First and third columns: cross-sectional uncertainty estimates. Second and fourth columns: approximate pointwise posterior variance. First and second columns: modelled PtO map. Third and fourth columns: learned PtO map.

Results: 0% Noise, $M = 5000$

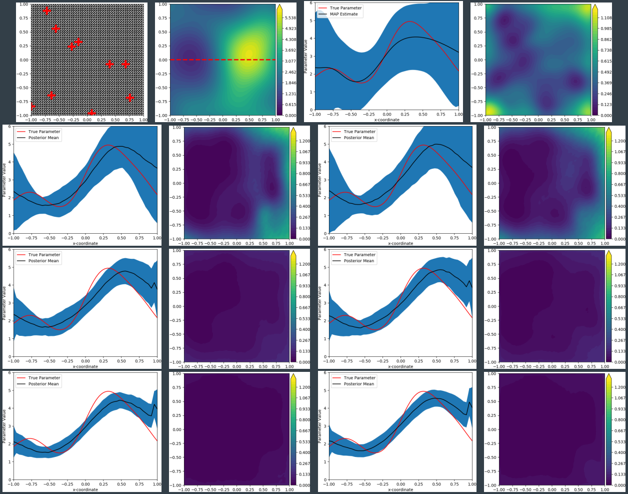


Figure: Top row left to right: mesh with sensors denoted with a red cross, true PoI, cross-sectional uncertainty estimate and pointwise posterior variance from Laplace approximation. Second to fourth rows: $\alpha = 0.00001, 0.1, 0.5$. First and third columns: cross-sectional uncertainty estimates. Second and fourth columns: approximate pointwise posterior variance. First and second columns: modelled PtO map. Third and fourth columns: learned PtO map.

Results: 1% Noise, $M = 50$

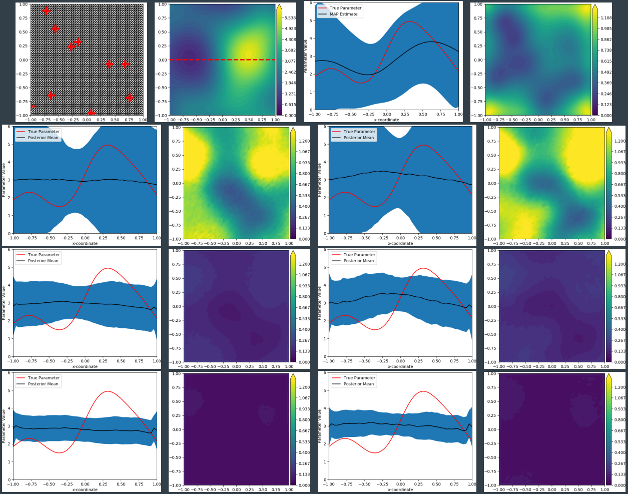


Figure: Top row left to right: mesh with sensors denoted with a red cross, true PoI, cross-sectional uncertainty estimate and pointwise posterior variance from Laplace approximation. Second to fourth rows: $\alpha = 0.00001, 0.1, 0.5$. First and third columns: cross-sectional uncertainty estimates. Second and fourth columns: approximate pointwise posterior variance. First and second columns: modelled PtO map. Third and fourth columns: learned PtO map.

Results: 1% Noise, $M = 500$

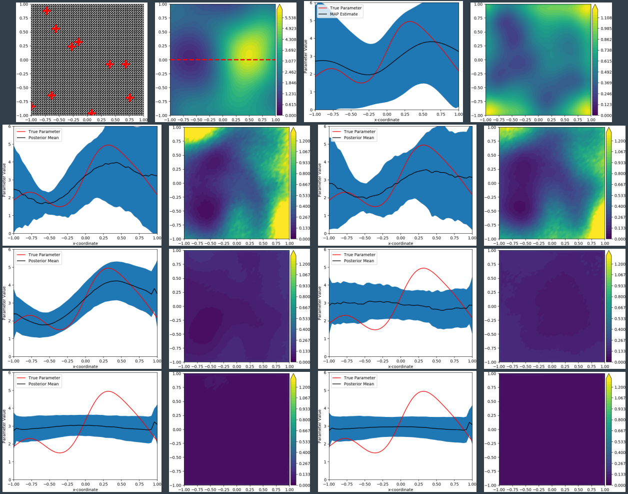


Figure: Top row left to right: mesh with sensors denoted with a red cross, true PoI, cross-sectional uncertainty estimate and pointwise posterior variance from Laplace approximation. Second to fourth rows: $\alpha = 0.00001, 0.1, 0.5$. First and third columns: cross-sectional uncertainty estimates. Second and fourth columns: approximate pointwise posterior variance. First and second columns: modelled PtO map. Third and fourth columns: learned PtO map.

Results: 1% Noise, $M = 5000$

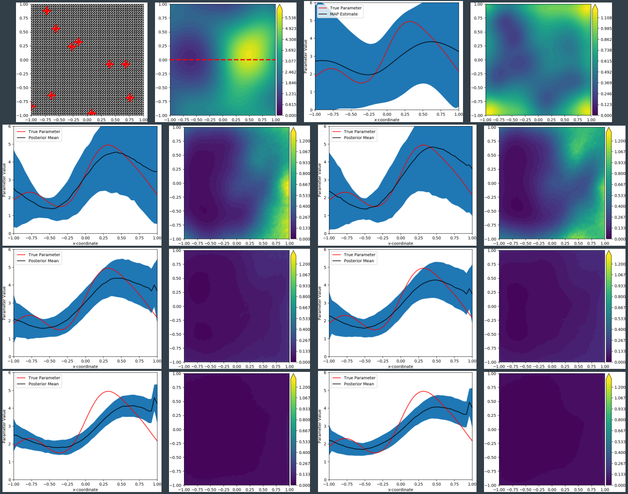


Figure: Top row left to right: mesh with sensors denoted with a red cross, true PoI, cross-sectional uncertainty estimate and pointwise posterior variance from Laplace approximation. Second to fourth rows: $\alpha = 0.00001, 0.1, 0.5$. First and third columns: cross-sectional uncertainty estimates. Second and fourth columns: approximate pointwise posterior variance. First and second columns: modelled PtO map. Third and fourth columns: learned PtO map.

Results: 5% Noise, $M = 50$

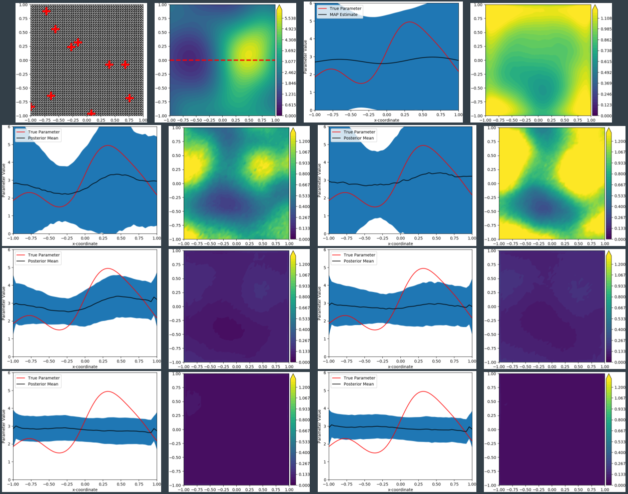


Figure: Top row left to right: mesh with sensors denoted with a red cross, true PoI, cross-sectional uncertainty estimate and pointwise posterior variance from Laplace approximation. Second to fourth rows: $\alpha = 0.00001, 0.1, 0.5$. First and third columns: cross-sectional uncertainty estimates. Second and fourth columns: approximate pointwise posterior variance. First and second columns: modelled PtO map. Third and fourth columns: learned PtO map.

Results: 5% Noise, $M = 500$

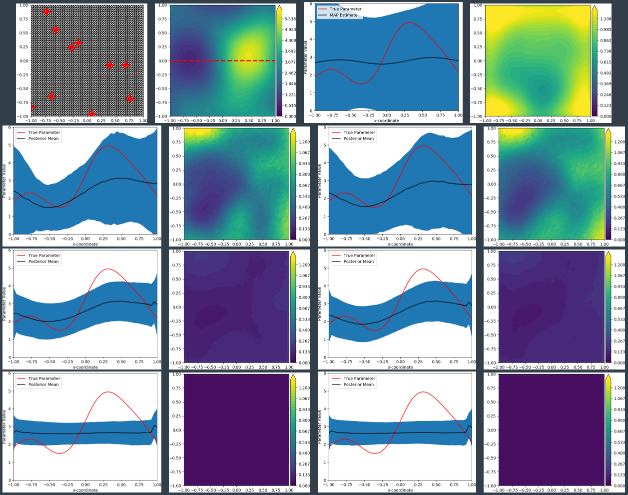


Figure: Top row left to right: mesh with sensors denoted with a red cross, true PoI, cross-sectional uncertainty estimate and pointwise posterior variance from Laplace approximation. Second to fourth rows: $\alpha = 0.00001, 0.1, 0.5$. First and third columns: cross-sectional uncertainty estimates. Second and fourth columns: approximate pointwise posterior variance. First and second columns: modelled PtO map. Third and fourth columns: learned PtO map.

Results: 5% Noise, $M = 5000$

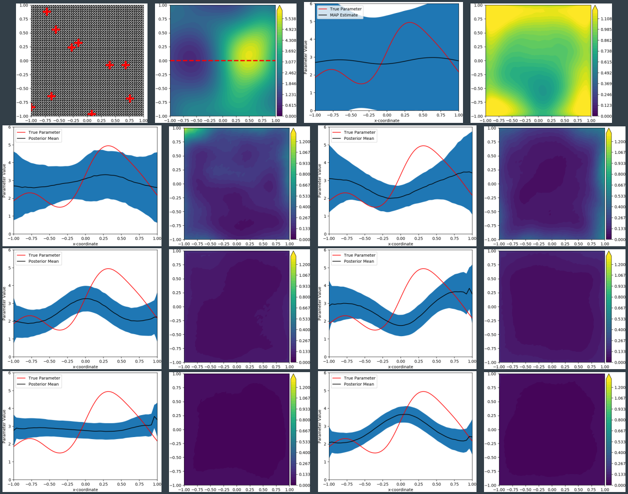


Figure: Top row left to right: mesh with sensors denoted with a red cross, true PoI, cross-sectional uncertainty estimate and pointwise posterior variance from Laplace approximation. Second to fourth rows: $\alpha = 0.00001, 0.1, 0.5$. First and third columns: cross-sectional uncertainty estimates. Second and fourth columns: approximate pointwise posterior variance. First and second columns: modelled PtO map. Third and fourth columns: learned PtO map.

Key Observations

- Selecting α small (zero-forcing KLD) yields larger uncertainty estimates
- Smaller datasets yield larger uncertainty estimates representing the lack of information
- Larger noise regularization yields larger uncertainty estimates
- Our framework yields feasible estimates for 0% and 1% noise but struggles with 5% noise when a large dataset is used

Conclusion

- This framework is derived from a solid mathematical foundation and possesses a complex, dynamic interplay of many factors from variational inference as well as regularization.
- Despite this complexity, the results show that the framework is robust and, aside from the usual tunable parameters associated with neural network architecture, requires relatively few design decisions.
- Our results also show that the estimates from our framework exhibits behaviour similar to that of more traditional methods.
- Our results also show that the estimates from our framework is responsive to the training dataset size. Larger datasets represent more information which is communicated with smaller uncertainty estimates.
- However, our preliminary investigation utilizes somewhat crude approximations. We believe that the results presented in this paper can be improved with the inclusion of more sophisticated statistical machinery.

Data-Driven Double-Edged Sword

- The utilization of datasets alleviates the burden on accurate prior and physics modelling.
- Poorly constructed or highly corrupted datasets could completely sabotage the inversion process regardless of any accuracy achieved by the prior and physics models.

Retains the Same Strategy of Traditional Methods

- Recall

$$p_{\text{post}}(\mathbf{u}|\mathbf{y}_{\text{obs}}) \propto p_E(\mathbf{y}_{\text{obs}} - \mathcal{F}(\mathbf{u}))p_{\text{pr}}(\mathbf{u}) \tag{33a}$$

$$= \exp\left(-\frac{1}{2}(\|\boldsymbol{\Gamma}_E^{-1}(\mathbf{y}_{\text{obs}} - \mathcal{F}(\mathbf{u}) - \boldsymbol{\mu}_E)\|^2_{\mathbb{R}^O} + \|\boldsymbol{\Gamma}_{\text{pr}}^{-1}(\mathbf{u} - \boldsymbol{\mu}_{\text{pr}})\|^2_{\mathbb{R}^D})\right). \tag{33b}$$

Where $\mathbf{u} = [u_1, u_2, \dots, u_D]^{\mathrm{T}}$ where

$$u_d = \sum_{k=1}^{D} u_d \varphi_k(\mathbf{x}_d). \tag{34}$$

- In contrast, one can view a neural network as an expansion over a non-linear basis. Indeed, consider a two-layer neural network to output the model posterior mean $\boldsymbol{\mu}_{\text{post}} = [\mu_1, \mu_2, \dots, \mu_D]^{\mathrm{T}}$:

$$\mu_d\left(\mathbf{y}_{\text{obs}}, \boldsymbol{W}^{\langle 1\rangle}, \boldsymbol{W}^{\langle 2\rangle}\right) = \sum_{k=1}^{K} \boldsymbol{W}^{\langle 2\rangle}_{dk}\, h\left(\sum_{o=1}^{O} \boldsymbol{W}^{\langle 1\rangle}_{ko}\, y_o\right) \tag{35}$$

where $\mathbf{y}_{\text{obs}} = [y_1, y_2, \dots, y_O]^{\mathrm{T}}$, $\boldsymbol{W}^{\langle l\rangle}$ is the lth layer of weights and h is an activation function.

- With this view, we have the non-linear basis function $\varphi_k = h\left(\sum_{o=1}^{O} \boldsymbol{W}^{\langle 1\rangle}_{ko}\, y_o\right)$ which is dependent on the optimization target $\boldsymbol{W}^{\langle 1\rangle}$.

Retains the Same Strategy of Traditional Methods

Here are a few points to consider:

- Whilst the inference procedure for a traditional method involves an optimization problem, the inference procedure of UQ-VAE involves a propagation through a trained neural network. Indeed, optimization is only required in UQ-VAE for training.
- Notice that the non-linear basis function $\varphi_k = h\left(\sum_{o=1}^{O} \boldsymbol{W}_{ko}^{\langle 1 \rangle} y_o\right)$ includes a dependence on the observation data $\mathbf{y}_{\text{obs}}$. It is this characteristic that encapsulates the idea that the optimization problem is used to train a solver.
- Whilst the optimization target for the traditional method appears only in the expansion coefficients, the optimization target for UQ-VAE appears in the basis function as well as its expansion coefficient. Therefore, intuitively, the optimization process for UQ-VAE is more ill-posed.

Generality Allows Jumping on the Band-wagon

- What we offer here is a mathematical framework and not a neural network architecture.
- Benefit of this is we are left with alot of room to sprinkle deep learning magic (ResNets, CNNs, Transformers, Domain specific-architectures)

Frank Nielsen.
A family of statistical symmetric divergences based on jensen's inequality.
arXiv preprint arXiv:1009.4004, 2010.

Ferenc Huszár.
How (not) to train your generative model: Scheduled sampling, likelihood, adversary?
arXiv preprint arXiv:1511.05101, 2015.

Hwan Goh, Sheroze Sheriffdeen, Jonathan Wittmer, and Tan Bui-Thanh.
Solving bayesian inverse problems via variational autoencoders.
arXiv preprint arXiv:1912.04212, 2020.

Hwan Goh.
uq-vae, 2020.
https://github.com/hwangoh/uq-vae.

Diederik P Kingma and Max Welling.
Auto-encoding variational bayes.
arXiv preprint arXiv:1312.6114, 2013.