Caltech

EnKF

Kovachki & Stuart

Invese Problems

Ensemble Kalman Inversion

Numerics

Ensemble Kalman Inversion: A Derivative-Free Technique For Machine Learning Tasks

Nikola B. Kovachki[1]   Andrew M. Stuart[1]

[1]Computational and Mathematical Sciences
California Institute of Technology

Applied Inverse Porblems
July 8-12th, 2019

Challenges

- Issues with gradient-based methods:
  - Suffer when parallelized
  - Hard to do Bayesian inference
  - Vanishing/Exploding gradients
  - Non-differentiable loss/model

Caltech

Table of Contents

Table of Contents

Supervised Learning

De Vito, Rosasco, Caponnetto, De Giovannini, and Odone 2005. (JMLR)

- **Data**: $\{(x_j, y_j)\}_{j=1}^N$ with $x_j \in \mathcal{X}$, $y_j \in \mathcal{Y}$.

Supervised Learning

De Vito, Rosasco, Caponnetto, De Giovannini, and Odone 2005. (JMLR)

- **Data**: $\{(x_j, y_j)\}_{j=1}^{N}$ with $x_j \in \mathcal{X}$, $y_j \in \mathcal{Y}$.
- **Find**: $\mathcal{G}(u|\cdot) : \mathcal{X} \to \mathcal{Y}$ for parameter $u \in \mathcal{U}$ consistent with the data.

Supervised Learning

De Vito, Rosasco, Caponnetto, De Giovannini, and Odone 2005. (JMLR)

- **Data**: $\{(x_j, y_j)\}_{j=1}^N$ with $x_j \in \mathcal{X}$, $y_j \in \mathcal{Y}$.
- **Find**: $\mathcal{G}(u|\cdot) : \mathcal{X} \to \mathcal{Y}$ for parameter $u \in \mathcal{U}$ consistent with the data.
- **Concatenate**:

$$\mathsf{y} = \mathsf{G}(u|\mathsf{x}) + \eta$$

where $\mathsf{G}(\cdot|\mathsf{x}) : \mathcal{U} \to \mathcal{Y}^N$ and η is model or data error.

Supervised Learning

De Vito, Rosasco, Caponnetto, De Giovannini, and Odone 2005. (JMLR)

- **Data**: $\{(x_j, y_j)\}_{j=1}^N$ with $x_j \in \mathcal{X}$, $y_j \in \mathcal{Y}$.
- **Find**: $\mathcal{G}(u|\cdot) : \mathcal{X} \to \mathcal{Y}$ for parameter $u \in \mathcal{U}$ consistent with the data.
- **Concatenate**:

$$\mathsf{y} = \mathsf{G}(u|\mathsf{x}) + \eta$$

where $\mathsf{G}(\cdot|\mathsf{x}) : \mathcal{U} \to \mathcal{Y}^N$ and η is model or data error.

- **Losses**: $\Phi(u; \mathsf{x}, \mathsf{y})$

$$\frac{1}{2}\|\mathsf{y} - \mathsf{G}(u|\mathsf{x})\|^2_{\mathcal{Y}^N} + R(u) \quad \text{or} \quad -\sum_{j=1}^N \langle y_j, \log \mathcal{G}(u|, x_j)\rangle_{\mathcal{Y}} + R(u)$$

Supervised Learning

De Vito, Rosasco, Caponnetto, De Giovannini, and Odone 2005. (JMLR)

- **Data**: $\{(x_j, y_j)\}_{j=1}^N$ with $x_j \in \mathcal{X}$, $y_j \in \mathcal{Y}$.
- **Find**: $\mathcal{G}(u|\cdot) : \mathcal{X} \to \mathcal{Y}$ for parameter $u \in \mathcal{U}$ consistent with the data.
- **Concatenate**:

$$\mathrm{y} = \mathrm{G}(u|\mathrm{x}) + \eta$$

where $\mathrm{G}(\cdot|\mathrm{x}) : \mathcal{U} \to \mathcal{Y}^N$ and η is model or data error.

- **Losses**: $\Phi(u; \mathrm{x}, \mathrm{y})$

$$\frac{1}{2}\|\mathrm{y} - \mathrm{G}(u|\mathrm{x})\|_{\mathcal{Y}^N}^2 + R(u) \quad \text{or} \quad -\sum_{j=1}^{N} \langle y_j, \log \mathcal{G}(u|, x_j)\rangle_{\mathcal{Y}} + R(u)$$

- **Standard Solution (SGD)**:

$$\dot{u} = -\nabla_u \Phi(u; \mathrm{x}, \mathrm{y}); \quad u(0) = u_0$$
$$u^* = u(T)$$

Online Supervised Learning

- **Data**: As before possibly with $N = \infty$.

Online Supervised Learning

- **Data**: As before possibly with $N = \infty$.
- **Dynamic**: For $j = 0, 1, 2, \ldots$

$$
\begin{aligned}
u_{j+1} &= u_j \\
y_{j+1} &= \mathcal{G}(u_{j+1} | x_{j+1}) + \eta_{j+1}
\end{aligned}
$$

Online Supervised Learning

- **Data**: As before possibly with $N = \infty$.
- **Dynamic**: For $j = 0, 1, 2, \ldots$

$$\begin{aligned} u_{j+1} &= u_j \\ y_{j+1} &= \mathcal{G}(u_{j+1} | x_{j+1}) + \eta_{j+1} \end{aligned}$$

- **Find**: u_j given $Y_j = \{y_k\}_{k=1}^{j}$ and update sequentially.

Online Supervised Learning

- **Data**: As before possibly with $N = \infty$.
- **Dynamic**: For $j = 0, 1, 2, \ldots$

$$
\begin{aligned}
u_{j+1} &= u_j \\
y_{j+1} &= \mathcal{G}(u_{j+1}|x_{j+1}) + \eta_{j+1}
\end{aligned}
$$

- **Find**: u_j given $Y_j = \{y_k\}_{k=1}^{j}$ and update sequentially.
- **Loss**: $\Phi(u; x, y)$

$$
\frac{1}{2}\|y - \mathcal{G}(u|x)\|_{\mathcal{Y}}^2 + R(u) \quad \text{or} \quad -\langle y, \log \mathcal{G}(u|x)\rangle_{\mathcal{Y}} + R(u)
$$

Online Supervised Learning

- **Data**: As before possibly with $N = \infty$.
- **Dynamic**: For $j = 0, 1, 2, \dots$

$$\begin{aligned} u_{j+1} &= u_j \\ y_{j+1} &= \mathcal{G}(u_{j+1}|x_{j+1}) + \eta_{j+1} \end{aligned}$$

- **Find**: u_j given $Y_j = \{y_k\}_{k=1}^{j}$ and update sequentially.
- **Loss**: $\Phi(u; x, y)$

$$\frac{1}{2}\|y - \mathcal{G}(u|x)\|_{\mathcal{Y}}^2 + R(u) \quad \text{or} \quad -\langle y, \log \mathcal{G}(u|x)\rangle_{\mathcal{Y}} + R(u)$$

- **Standard Solution (OGD)**:

$$\begin{aligned} \dot{u} &= -\nabla_u \Phi(u; x_{j+1}, y_{j+1}); \quad u(0) = u_j \\ u_{j+1} &= u(T_j) \end{aligned}$$

Table of Contents

Continous-time EnKF for Inversion

Kantas, Beskos, Jasra, 2014. (JUQ)

Iglesias, Law and Stuart, 2013. (IP)

- **Inverse Problem**:

$$y = \mathrm{G}(u) + \eta \qquad \eta \sim \mathcal{N}(0, \Gamma)$$
$$u \sim \mu_0(u)$$

Continous-time EnKF for Inversion

Kantas, Beskos, Jasra, 2014. (JUQ)

Iglesias, Law and Stuart, 2013. (IP)

- **Inverse Problem**:

$$y = G(u) + \eta \qquad \eta \sim \mathcal{N}(0, \Gamma)$$
$$u \sim \mu_0(u)$$

- **Sequential Monte Carlo (SMC)**:

$$\mu_{n+1}(du) \propto \exp(-hn\Phi(u; y))\mu_0(du)$$

Continous-time EnKF for Inversion

Kantas, Beskos, Jasra, 2014. (JUQ)

Iglesias, Law and Stuart, 2013. (IP)

- **Inverse Problem**:

$$y = G(u) + \eta \qquad \eta \sim \mathcal{N}(0, \Gamma)$$
$$u \sim \mu_0(u)$$

- **Sequential Monte Carlo (SMC)**:

$$\mu_{n+1}(du) \propto \exp(-hn\Phi(u; y))\mu_0(du)$$

- **Approximate SMC (EnKF)**:

$$u_{n+1}^{(j)} = u_n^{(j)} + C^{uw}(u_n)(C^{ww}(u_n) + \Gamma)^{-1}(y - G(u_n^{(j)}))$$

Continous-time EnKF for Inversion

Kantas, Beskos, Jasra, 2014. (JUQ)

Iglesias, Law and Stuart, 2013. (IP)

- **Inverse Problem**:

$$y = G(u) + \eta \qquad \eta \sim \mathcal{N}(0, \Gamma)$$
$$u \sim \mu_0(u)$$

- **Sequential Monte Carlo (SMC)**:

$$\mu_{n+1}(du) \propto \exp(-hn\Phi(u; y))\mu_0(du)$$

- **Approximate SMC (EnKF)**:

$$u_{n+1}^{(j)} = u_n^{(j)} + C^{uw}(u_n)(C^{ww}(u_n) + \Gamma)^{-1}(y - G(u_n^{(j)}))$$

- **Continous-time**: $\Gamma \mapsto \frac{1}{h}\Gamma$, $h \to 0$

$$\dot{u}^{(j)} = -\frac{1}{J}\sum_{k=1}^{J}\langle G(u^{(k)}) - \bar{G}, G(u^{(j)}) - y\rangle_\Gamma \; u^{(k)}$$

Approximate Natural Gradient Decent

Amari, 1998. (NC)

Ollivier, 2017. (CoRR); Ollivier, 2019. (CoRR)

- **Linear**: $\mathrm{G}(\cdot) = A\cdot$

$$\dot{u}^{(j)} = -C(u)\nabla_u \Phi(u^{(j)}, \mathrm{y})$$

where

$$C(u) = \frac{1}{J}\sum_{j=1}^{J}(u^{(j)} - \bar{u}) \otimes (u^{(j)} - \bar{u}); \quad \Phi(u, \mathrm{y}) = \frac{1}{2}\|\mathrm{y} - Au\|_{\Gamma}^{2}$$

Approximate Natural Gradient Decent

Amari, 1998. (NC)

Ollivier, 2017. (CoRR); Ollivier, 2019. (CoRR)

- **Linear**: $\mathrm{G}(\cdot) = A\cdot$

$$\dot{u}^{(j)} = -C(u)\nabla_u \Phi(u^{(j)}, \mathrm{y})$$

where

$$C(u) = \frac{1}{J}\sum_{j=1}^{J}(u^{(j)} - \bar{u}) \otimes (u^{(j)} - \bar{u}); \quad \Phi(u, \mathrm{y}) = \frac{1}{2}\|\mathrm{y} - Au\|_{\Gamma}^2$$

- **Natural Gradient Decent**:

$$\dot{u} = -F^{-1}(u)\nabla_u \Phi(u, \mathrm{y})$$

Approximate Natural Gradient Decent

Amari, 1998. (NC)

Ollivier, 2017. (CoRR); Ollivier, 2019. (CoRR)

- **Linear**: $\mathrm{G}(\cdot) = A\cdot$

$$\dot{u}^{(j)} = -C(u)\nabla_u \Phi(u^{(j)}, \mathrm{y})$$

where

$$C(u) = \frac{1}{J}\sum_{j=1}^{J}(u^{(j)} - \bar{u}) \otimes (u^{(j)} - \bar{u}); \quad \Phi(u, \mathrm{y}) = \frac{1}{2}\|\mathrm{y} - Au\|_{\Gamma}^{2}$$

- **Natural Gradient Decent**:

$$\dot{u} = -F^{-1}(u)\nabla_u \Phi(u, \mathrm{y})$$

- **Cramér-Rao**:

$$C(u) \succeq F^{-1}(u)$$

Approximate Natural Gradient Decent

- **Non-linear**: $\mathsf{G}(\cdot)$

$$\mathsf{G}(u^{(k)}) = \mathsf{G}(u^{(j)} + u^{(k)} - u^{(j)}) \approx \mathsf{G}(u^{(j)}) + D\mathsf{G}(u^{(j)})(u^{(k)} - u^{(j)})$$
$$\mathsf{G}(u^{(l)}) = \mathsf{G}(u^{(j)} + u^{(l)} - u^{(j)}) \approx \mathsf{G}(u^{(j)}) + D\mathsf{G}(u^{(j)})(u^{(l)} - u^{(j)})$$

Approximate Natural Gradient Decent

- **Non-linear**: $\mathsf{G}(\cdot)$

$$\mathsf{G}(u^{(k)}) = \mathsf{G}(u^{(j)} + u^{(k)} - u^{(j)}) \approx \mathsf{G}(u^{(j)}) + D\mathsf{G}(u^{(j)})(u^{(k)} - u^{(j)})$$
$$\mathsf{G}(u^{(l)}) = \mathsf{G}(u^{(j)} + u^{(l)} - u^{(j)}) \approx \mathsf{G}(u^{(j)}) + D\mathsf{G}(u^{(j)})(u^{(l)} - u^{(j)})$$

- **Re-write**:

$$\begin{aligned}\dot{u}^{(j)} &= -\frac{1}{J^2}\sum_{k=1}^{J}\sum_{l=1}^{J}\langle \mathsf{G}(u^{(k)}) - \mathsf{G}(u^{(l)}), \mathsf{G}(u^{(j)}) - \mathsf{y}\rangle_\Gamma (u^{(k)} - \bar{u}) \\ &\approx -\frac{1}{J}\sum_{k=1}^{J}\langle D\mathsf{G}^*(u^{(j)})(\mathsf{G}(u^{(j)}) - \mathsf{y}), u^{(k)} - \bar{u}\rangle_\Gamma (u^{(k)} - \bar{u}) \\ &= -C(u)\nabla_u \Phi(u^{(j)}, \mathsf{y})\end{aligned}$$

Long-time Linear Behavior

Schillings and Stuart 2017. (SINUM)

Theorem (CS, AMS)

Suppse $\mathrm{G}(\cdot) = A\cdot$ *and that* y *is the image of a truth* $u^\dagger$ *under* A. *Define* $r^{(j)}(t) = u^{(j)}(t) - u^\dagger$ *then (under some assumptions)*

$$Ar^{(j)}(t) = Ar_{\|}^{(j)}(t) + Ar_{\perp}^{(j)}(t)$$

with $Ar_{\|}^{(j)} \in span\{A(u^{(j)}(0) - \bar{u}(0))\}$ *and* $Ar_{\perp}^{(j)} \in span\{A(u^{(j)}(0) - \bar{u}(0))\}^{\perp}$. *Furthermore*

$$Ar_{\|}^{(j)}(t) = \mathcal{O}\left(\frac{1}{t}\right)$$

$$Ar_{\perp}^{(j)}(t) = Ar_{\perp}^{(1)}(0) \quad \forall t \geq 0.$$

Arbitrary Loss

- **Consider**:

$$\begin{aligned}\dot{u}^{(j)} &= -C^{\mathrm{uw}}(u)\Gamma^{-1}(\mathrm{G}(u^{(j)}) - y)\\ &= -C^{\mathrm{uw}}(u)\nabla_{\mathrm{z}}\Psi(\mathrm{G}(u^{(j)}), \mathrm{y})\end{aligned}$$

where

$$C^{\mathrm{uw}}(u) = \frac{1}{J}\sum_{j=1}^{J}(u^{(j)} - \bar{u}) \otimes (\mathrm{G}(u^{(j)}) - \bar{\mathrm{G}}); \quad \Psi(\mathrm{z}, \mathrm{y}) = \frac{1}{2}\|\mathrm{y} - \mathrm{z}\|_{\Gamma}^{2}$$

Caltech

Arbitrary Loss

- **Consider**:

$$\begin{aligned}\dot{u}^{(j)} &= -C^{\mathrm{uw}}(u)\Gamma^{-1}(\mathrm{G}(u^{(j)}) - y)\\ &= -C^{\mathrm{uw}}(u)\nabla_{\mathrm{z}}\Psi(\mathrm{G}(u^{(j)}), \mathrm{y})\end{aligned}$$

where

$$C^{\mathrm{uw}}(u) = \frac{1}{J}\sum_{j=1}^{J}(u^{(j)} - \bar{u}) \otimes (\mathrm{G}(u^{(j)}) - \bar{\mathrm{G}}); \quad \Psi(\mathrm{z}, \mathrm{y}) = \frac{1}{2}\|\mathrm{y} - \mathrm{z}\|_{\Gamma}^{2}$$

- **New Method**: Replace $\frac{1}{2}\|\mathrm{y} - \mathrm{z}\|_{\Gamma}^{2}$ with arbitrary loss.

Arbitrary Loss

- **Consider**:

$$\begin{aligned}\dot{u}^{(j)} &= -C^{\mathrm{uw}}(u)\Gamma^{-1}(\mathsf{G}(u^{(j)}) - y) \\ &= -C^{\mathrm{uw}}(u)\nabla_{\mathsf{z}}\Psi(\mathsf{G}(u^{(j)}), \mathsf{y})\end{aligned}$$

where

$$C^{\mathrm{uw}}(u) = \frac{1}{J}\sum_{j=1}^{J}(u^{(j)} - \bar{u}) \otimes (\mathsf{G}(u^{(j)}) - \bar{\mathsf{G}}); \quad \Psi(\mathsf{z}, \mathsf{y}) = \frac{1}{2}\|\mathsf{y} - \mathsf{z}\|_{\Gamma}^{2}$$

- **New Method**: Replace $\frac{1}{2}\|\mathsf{y} - \mathsf{z}\|_{\Gamma}^{2}$ with arbitrary loss.
- **Cross Entropy**:

$$\Psi(z, y) = -\langle \mathsf{y}, \log \mathsf{z}\rangle_{\mathcal{Y}^N}, \quad \nabla_z \Psi(\mathsf{G}(u^{(j)}), \mathsf{y}) = -\frac{\mathsf{y}}{\mathsf{G}(u^{(j)})}$$

Momentum

Su, Boyd, Candes 2015. (JMLR)

- **Nesterov EKI**:

$$\ddot{u}^{(j)} + \frac{3}{t}\dot{u}^{(j)} = -C^{\mathrm{uw}}(u)\nabla_{\mathrm{z}}\Psi(\mathrm{G}(u^{(j)}), \mathrm{y})$$

Momentum

Su, Boyd, Candes 2015. (JMLR)

- **Nesterov EKI**:

$$\ddot{u}^{(j)} + \frac{3}{t}\dot{u}^{(j)} = -C^{\mathrm{uw}}(u)\nabla_{\mathrm{z}}\Psi(\mathrm{G}(u^{(j)}), \mathrm{y})$$

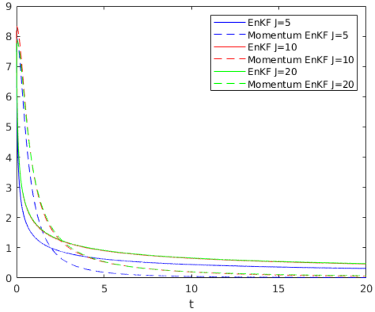


Figure: Linear problem: ℓ_2 error.

Discrete Scheme

- **Concatenate**: $u = [u^{(1)}, \dots, u^{(J)}]$ and define $D(u) \in \mathbb{R}^{J \times J}$ by

$$D^{(jk)}(u) = \frac{1}{J}\langle \mathsf{G}(u^{(k)}) - \bar{\mathsf{G}}, \nabla_{\mathsf{z}}\Psi(\mathsf{G}(u^{(j)}), \mathsf{y})\rangle$$

Discrete Scheme

- **Concatenate**: $u = [u^{(1)}, \dots, u^{(J)}]$ and define $D(u) \in \mathbb{R}^{J\times J}$ by

$$D^{(jk)}(u) = \frac{1}{J}\langle \mathrm{G}(u^{(k)}) - \bar{\mathrm{G}}, \nabla_{\mathrm{z}}\Psi(\mathrm{G}(u^{(j)}), \mathrm{y})\rangle$$

- **Discretize**:

$$u_{n+1} = u_n - h_n D(u_n) u_n$$

or

$$u_{n+1} = v_n - h_n D(v_n) v_n$$
$$v_{n+1} = u_{n+1} + \frac{n}{n+3}(u_{n+1} - u_n)$$

with $v_0 = u_0$ and adaptive time step: $h_n = \frac{h}{\|D(u_n)\| + \epsilon}$

Initialization, Noise, and Predictions

Glorot and Bengio. (JMLR)

Garbuno-Inigo, Hoffmann, Li and Stuart 2019. (CoRR)

- **Initial Ensemble**:

$$u_0^{(1)}, \dots, u_0^{(J)} \sim \mu_0(u)$$

Initialization, Noise, and Predictions

Glorot and Bengio. (JMLR)

Garbuno-Inigo, Hoffmann, Li and Stuart 2019. (CoRR)

- **Initial Ensemble**:

$$u_0^{(1)}, \dots, u_0^{(J)} \sim \mu_0(u)$$

- **Mini-batch data** (at each step):

$$\mathrm{x}_n = \{x_{i_l^{(n)}}\}_{l=1}^m \quad \mathrm{y}_n = \{y_{i_l^{(n)}}\}_{l=1}^m$$

where $\{i_1^{(n)}, \dots, i_m^{(n)}\} \subseteq \{1, \dots, N\}$.

Initialization, Noise, and Predictions

Glorot and Bengio. (JMLR)

Garbuno-Inigo, Hoffmann, Li and Stuart 2019. (CoRR)

- **Initial Ensemble**:

$$u_0^{(1)}, \dots, u_0^{(J)} \sim \mu_0(u)$$

- **Mini-batch data** (at each step):

$$\mathrm{x}_n = \{x_{i_l^{(n)}}\}_{l=1}^m \quad \mathrm{y}_n = \{y_{i_l^{(n)}}\}_{l=1}^m$$

where $\{i_1^{(n)}, \dots, i_m^{(n)}\} \subseteq \{1, \dots, N\}$.

- **Particle Spread**:

$$u_{n+1}^{(j)} = \bar{u}_n + \xi_{n+1}^{(j)} \qquad \xi_{n+1}^{(j)} \sim \gamma(n)\mu_0(u)$$

expanding ensnemble

Initialization, Noise, and Predictions

Glorot and Bengio. (JMLR)

Garbuno-Inigo, Hoffmann, Li and Stuart 2019. (CoRR)

- **Initial Ensemble**:

$$u_0^{(1)}, \dots, u_0^{(J)} \sim \mu_0(u)$$

- **Mini-batch data** (at each step):

$$\mathrm{x}_n = \{x_{i_l^{(n)}}\}_{l=1}^m \quad \mathrm{y}_n = \{y_{i_l^{(n)}}\}_{l=1}^m$$

  where $\{i_1^{(n)}, \dots, i_m^{(n)}\} \subseteq \{1, \dots, N\}$.

- **Particle Spread**:

$$u_{n+1}^{(j)} = \bar{u}_n + \xi_{n+1}^{(j)} \qquad \xi_{n+1}^{(j)} \sim \gamma(n)\mu_0(u)$$

  expanding ensnemble

- **Predict**:

$$\bar{u}_{n+1} = \frac{1}{J}\sum_{j=1}^{J} u_{n+1}^{(j)}$$

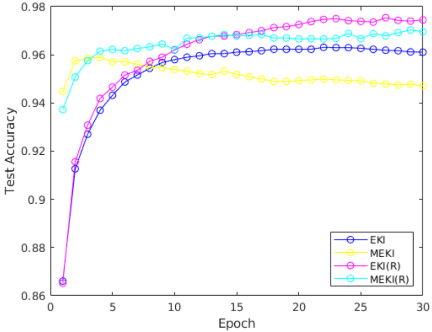
Table of Contents

Dense Networks (MNIST)

		DNN 1		DNN 2		DNN 3		DNN 4
SGD		0.9199		0.9735		0.9798		0.9818
MSGD		0.9257		**0.9807**		**0.9830**		**0.9840**
EKI	$\bar{u}$	0.9092	$\bar{u}$	0.9398	$\bar{u}$	0.9424	$\bar{u}$	0.9404
	$u^{(j^*)}$	0.9114	$u^{(j^*)}$	0.9416	$u^{(j^*)}$	0.9432	$u^{(j^*)}$	0.9418
MEKI	$\bar{u}$	0.9094	$\bar{u}$	0.9320		n/a		n/a
	$u^{(j^*)}$	0.9107	$u^{(j^*)}$	0.9332				
EKI(R)	$\bar{u}$	0.9252	$\bar{u}$	0.9721	$\bar{u}$	0.9738	$\bar{u}$	0.9741
	$u^{(j^*)}$	**0.9260**	$u^{(j^*)}$	0.9695	$u^{(j^*)}$	0.9716	$u^{(j^*)}$	0.9691
MEKI(R)	$\bar{u}$	0.9142	$\bar{u}$	0.9509		n/a		n/a
	$u^{(j^*)}$	0.9162	$u^{(j^*)}$	0.9511				

Figure: Final test accuracies of six training methods on four dense neural networks, solving the MNIST classification problem. Each bold number is the maximum across the column. For each EKI method we report the accuracy of the mean particle $\bar{u}$ and of the best performing particle in the ensemble $u^{(j^*)}$. The ensemble size is $J = 2000$.

Dense Networks (MNIST)



	First	Final
EKI	0.8661	0.9611
MEKI	**0.9447**	0.9471
EKI(R)	0.8652	**0.9745**
MEKI(R)	0.9373	0.9696

Figure: Comparison of the test accuracies of four EKI methods on DNN 2 with ensemble size $J = 6000$.

Caltech

Ensemble Size Scaling

EnKF

Kovachki & Stuart

Invese Problems

Ensemble Kalman Inversion

Numerics

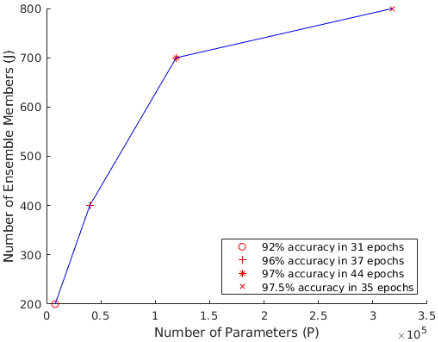


Figure: MNIST

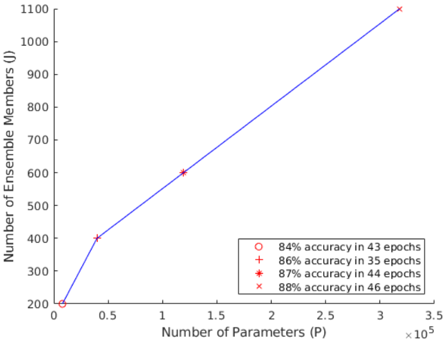


Figure: FashionMNIST

Convolutional Networks

Romero, Ballas, Kahou, Chassang, Gatta, Bengio, 2015. (ICML)

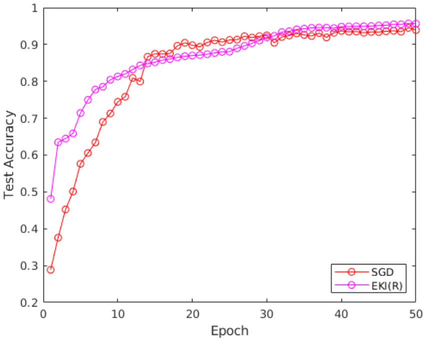
CNN-MNIST	CNN-1	CNN-2	CNN-3
Conv 16x3x3	Conv 16x3x3	Conv 16x3x3	Conv 16x3x3
Conv 16x3x3	Conv 16x3x3	Conv 16x3x3	Conv 16x3x3
		Conv 16x3x3	Conv 32x3x3
			Conv 32x3x3
			Conv 32x3x3
MaxPool 4x4 ($s = 2$)	MaxPool 2x2	MaxPool 2x2	MaxPool 2x2
Conv 16x3x3	Conv 16x3x3	Conv 32x3x3	Conv 48x3x3
Conv 16x3x3	Conv 16x3x3	Conv 32x3x3	Conv 48x3x3
		Conv 32x3x3	Conv 48x3x3
			Conv 48x3x3
			Conv 48x3x3
MaxPool 4x4 ($s = 2$)	MaxPool 2x2	MaxPool 2x2	MaxPool 2x2
Conv 12x3x3	Conv 32x3x3	Conv 48x3x3	Conv 64x3x3
Conv 12x3x3	Conv 32x3x3	Conv 48x3x3	Conv 64x3x3
		Conv 64x3x3	Conv 96x3x3
			Conv 96x3x3
			Conv 96x3x3
MaxPool 2x2	MaxPool 8x8	MaxPool 8x8	MaxPool 8x8
FC-10	FC-500	FC-500	FC-500
	FC-10	FC-10	FC-10

Convolutional Networks

	CNN-MNIST	CNN-1	CNN-2	CNN-3
SGD	0.9878	n/a	n/a	n/a
MSGD	0.9880	0.9150	0.9324	**0.9414**
EKI(R)	**0.9912**	**0.9249**	**0.9353**	0.9299

Figure: Comparison of the test accuracies of SGD and EKI(R) on four convolutional neural networks. SGD(M) refers to momentum SGD. CNN-MNIST is trained on the MNIST data set, while CNN-(1,2,3) are trained on the SVHN data set.

Recurrent Network (Classification)



	First	Final
SGD	0.2825	0.9391
EKI(R)	**0.4810**	**0.9566**

Figure: Two-layer RNN on MNIST

Caltech

Recurrent Network (Regression)

EnKF

Kovachki & Stuart

Invese Problems

Ensemble Kalman Inversion

Numerics

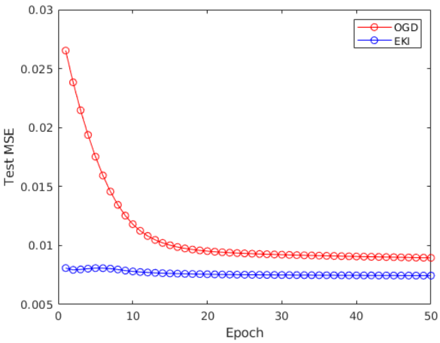


Figure: Temps

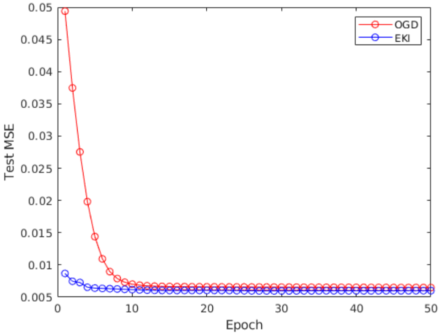


Figure: Sunspots

	Melbourne Temperatures		Zürich Sunspots	
	First	Final	First	Final
OGD	2.653×10^{-2}	8.954×10^{-3}	4.939×10^{-2}	6.480×10^{-3}
EKI	$\mathbf{8.086 \times 10^{-3}}$	$\mathbf{7.448 \times 10^{-3}}$	$\mathbf{8.671 \times 10^{-3}}$	$\mathbf{6.006 \times 10^{-3}}$

LSTM (Batched Online Regression)

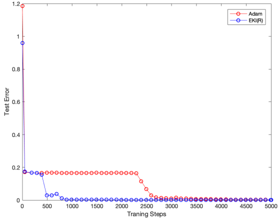


Figure: $T = 100$

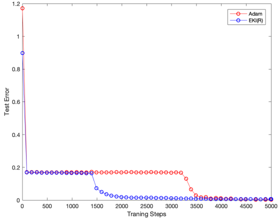


Figure: $T = 200$

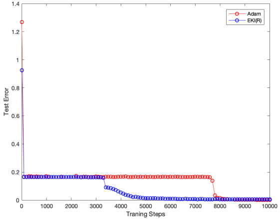


Figure: $T = 300$

$\mathcal{O}(10^5)$ reduction in the number of examples needed.

Conclusion

- Machine learning as inverse/filtering problem
- EKI as a minimization scheme/natural gradient decent
- Generalizations to original method
- Numerics show promise as alternative for:
  - SGD
  - OGD
  - BPTT
- Future directions:
  - Parallelize method/State-of-the-art FFN(s)
  - State-of-the-art RNN(s)
  - Ensemble for model reduction/UQ
  - Non-differentiable models
  - Non-differentiable loss/reinforcement learning

Kovachki, Stuart 2019 Inverse Problems https://doi.org/10.1088/1361-6420/ab1c3a

References I

- Bertozzi L. A., Flenner A. Diffuse Interface Models on Graphs for Classification of High Dimensional Data. *Multiscale Model. Simul.* 10(3). pp. 1090–1118. 2012.
- Bertozzi L. A., Luo X., Stuart M. A., Zygalakis C. K. Uncertainty Qunatification in the Classification of High Dimensional Data. *Preprint*. arXiv:1703.08816. 2017.
- Kantas N., Luo X., Beskos A., Jasra A. Sequential Monte Carlo methods for high-dimensional inverse problems. *SIAM/ASA Journal on Uncertainty Quantification*, 2. pp. 464-489. 2014.
- Iglesias M., Law K., Stuart M. A. Ensemble Kalman methods for inverse problems. *Inverse Problems*, 29. p. 045001. 2013.
- Schillings C., Stuart M. A. Analysis of the ensemble Kalman filter for inverse problems. *SIAM J Numerical Analysis*, 55(3). pp. 1264-1290. 2017.

References II

Amari S. Natural Gradient Works Efficiently in Learning. *Neural Computation*, 10. pp. 251–276. 1998.

Ollivier Y. The Extended Kalman Filter is a Natural Gradient Descent in Trajectory Space. *CoRR*, 1901.00696. 2019.