DARTMOUTH
Department of
Mathematics

DARTMOUTH
Guarini School of Graduate
and Advanced Studies

Generalized Sparsity-Promoting Solvers and Uncertainty Quantification for Bayesian Inverse Problems

CMCC Webinar

5th March 2024

Jonathan Lindbloom†, Jan Glaubitz, and Anne Gelb

Outline

1. Overview
2. Generalizations
3. Priorconditioning
4. Application to UQ
5. Other directions

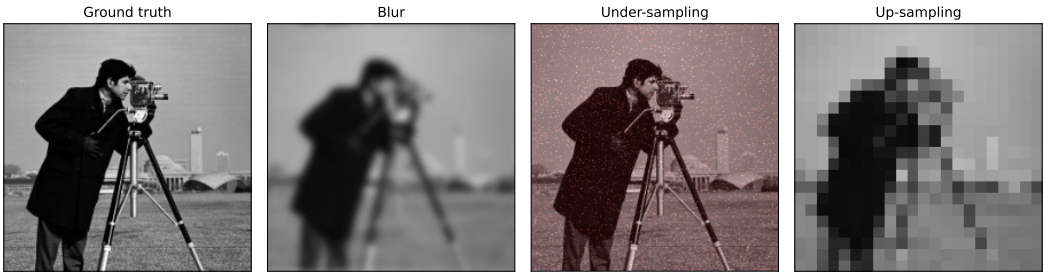
OVERVIEW

Problem setup

We would like to reconstruct an unknown signal $\boldsymbol{x} \in \mathbb{R}^n$ given an indirect noisy observation

$$\boldsymbol{y} = \boldsymbol{F}\boldsymbol{x} + \boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \sigma^2 \boldsymbol{I}_n).$$

Here $\boldsymbol{y}, \boldsymbol{\epsilon} \in \mathbb{R}^m$ with linear forward operator $\boldsymbol{F} \in \mathbb{R}^{m \times n}$.



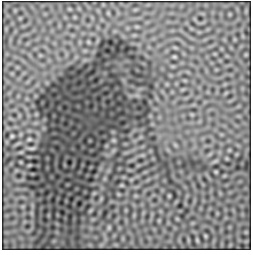
Maximum likelihood

Observation (no noise)

Least-squares reconstruction

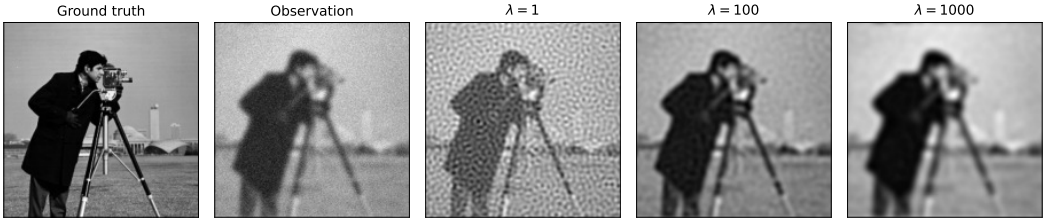
Observation (with noise)

Least-squares reconstruction



$$\boldsymbol{x}^{\mathrm{MLE}} = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\arg\min} \ \|\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y}\|_2$$

ℓ_2 regularization



$$\arg\min_{\boldsymbol{x}\in\mathbb{R}^n} \frac{1}{2\sigma^2}\|\boldsymbol{F}\boldsymbol{x}-\boldsymbol{y}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{R}\boldsymbol{x}\|_2^2$$

Statistical interpretation

$$\boldsymbol{y} \,|\, \boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{F}\boldsymbol{x}, \sigma^2 \boldsymbol{I}_n\right)$$
$$\boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{0}, \left(\lambda \boldsymbol{R}^T \boldsymbol{R}\right)^{-1}\right)$$

$$\implies \boldsymbol{x} \,|\, \boldsymbol{y} \sim \mathcal{N}\left(\boldsymbol{\mu}_{\text{post}}, \boldsymbol{Q}_{\text{post}}^{-1}\right), \quad \boldsymbol{Q}_{\text{post}} = \sigma^{-2}\boldsymbol{F}^T\boldsymbol{F} + \lambda \boldsymbol{R}^T\boldsymbol{R}, \quad \boldsymbol{\mu}_{\text{post}} = \sigma^{-2}\boldsymbol{Q}_{\text{post}}^{-1}\boldsymbol{F}^T\boldsymbol{y}$$

$$-\log \pi(\boldsymbol{x} \mid \boldsymbol{y}) + C = \frac{1}{2\sigma^2}\|\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{R}\boldsymbol{x}\|_2^2$$
$$:= \mathcal{G}(\boldsymbol{x})$$

Statistical interpretation

$$\boldsymbol{y} \,|\, \boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{F}\boldsymbol{x}, \sigma^2 \boldsymbol{I}_n\right)$$
$$\boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{0}, \left(\lambda \boldsymbol{R}^T \boldsymbol{R}\right)^{-1}\right)$$

$$\implies \boldsymbol{x} \,|\, \boldsymbol{y} \sim \mathcal{N}\left(\boldsymbol{\mu}_{\text{post}}, \boldsymbol{Q}_{\text{post}}^{-1}\right), \quad \boldsymbol{Q}_{\text{post}} = \sigma^{-2}\boldsymbol{F}^T\boldsymbol{F} + \lambda \boldsymbol{R}^T\boldsymbol{R}, \quad \boldsymbol{\mu}_{\text{post}} = \sigma^{-2}\boldsymbol{Q}_{\text{post}}^{-1}\boldsymbol{F}^T\boldsymbol{y}$$

$$-\log \pi(\boldsymbol{x} \mid \boldsymbol{y}) + C = \frac{1}{2\sigma^2}\|\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{R}\boldsymbol{x}\|_2^2$$
$$:= \mathcal{G}(\boldsymbol{x})$$

Statistical interpretation

$$\boldsymbol{y} \,|\, \boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{F}\boldsymbol{x}, \sigma^2 \boldsymbol{I}_n\right)$$
$$\boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{0}, \left(\lambda \boldsymbol{R}^T \boldsymbol{R}\right)^{-1}\right)$$

$$\implies \boldsymbol{x} \,|\, \boldsymbol{y} \sim \mathcal{N}\left(\boldsymbol{\mu}_{\text{post}}, \boldsymbol{Q}_{\text{post}}^{-1}\right), \quad \boldsymbol{Q}_{\text{post}} = \sigma^{-2}\boldsymbol{F}^T\boldsymbol{F} + \lambda \boldsymbol{R}^T\boldsymbol{R}, \quad \boldsymbol{\mu}_{\text{post}} = \sigma^{-2}\boldsymbol{Q}_{\text{post}}^{-1}\boldsymbol{F}^T\boldsymbol{y}$$

$$-\log \pi(\boldsymbol{x} \mid \boldsymbol{y}) + C = \frac{1}{2\sigma^2}\|\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \frac{\lambda}{2}\|\boldsymbol{R}\boldsymbol{x}\|_2^2$$
$$:= \mathcal{G}(\boldsymbol{x})$$

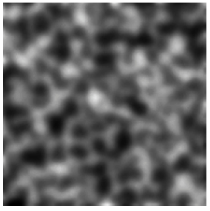
Gaussian vs. non-Gaussian priors

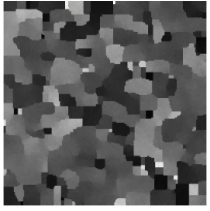
Gaussian:

- MAP estimation under a Gaussian prior amounts to solving a linear system / least squares problem 👍
- Sampling high-dimensional Gaussians has been studied extensively 👍

Non-Gaussian:

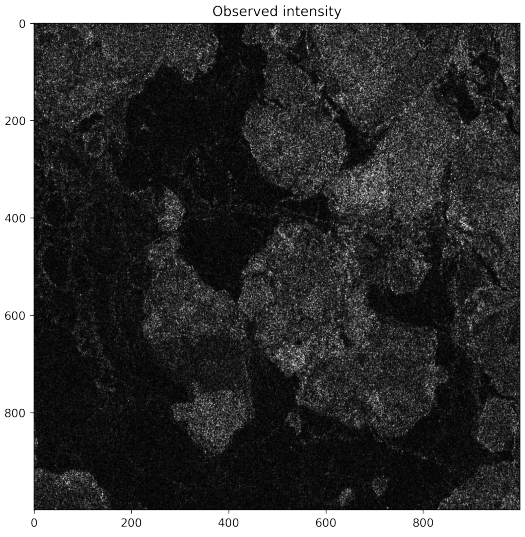
- MAP estimation usually requires minimizing non-quadratic/nonsmooth objective 👎
- To sample must usually resort to MCMC, but even this is challenging 👎

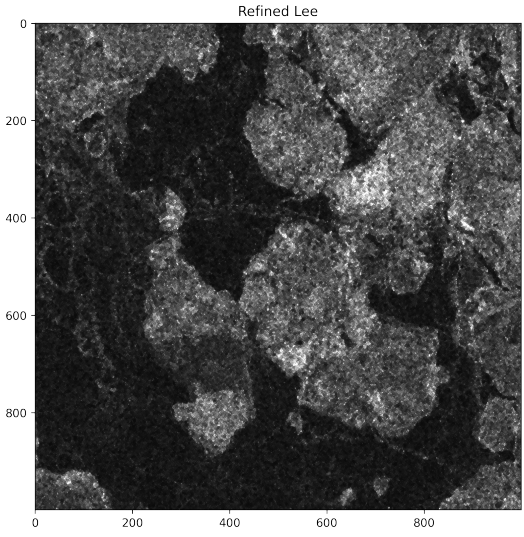


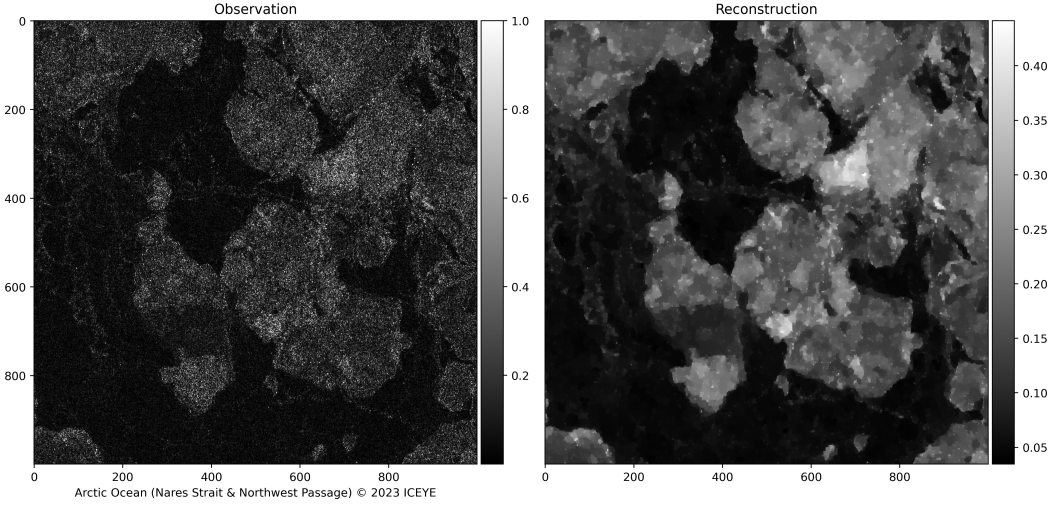


M. Markkanen, L. Roininen, J. M. Huttunen, and S. Lasanen, Cauchy difference priors for edge-preserving bayesian inversion, Journal of Inverse and Ill-posed Problems, 27 (2019), pp. 225–24.

Sentinel-1, courtesy of ESA


Observed intensity
0
200
400
600
800
0
200
400
600
800


Refined Lee
0
200
400
600
800
0
200
400
600
800


Observation
Reconstruction
0
200
400
600
800
1.0
0.8
0.6
0.4
0.2
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
Arctic Ocean (Nares Strait & Northwest Passage) © 2023 ICEYE

Example: Laplace prior

$$\pi(\boldsymbol{x}) \propto \exp\left\{-\lambda\|\boldsymbol{R}\boldsymbol{x}\|_1\right\} \qquad \boldsymbol{x}^\star = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\arg\min}\ \frac{1}{2\sigma^2}\|\boldsymbol{F}\boldsymbol{x}-\boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{R}\boldsymbol{x}\|_1$$

Solvers?

- Iterative shrinkage-thresholding algorithms (ISTA, FISTA, MFISTA, etc.)
  - A. Beck, First-Order Methods in Optimization, SIAM, 2017.
- Split Bregman method
  - T. Goldstein and S. Osher, The split bregman method for l1-regularized problems, SIAM journal on imaging sciences, 2 (2009), pp. 323–343.
- Iteratively re-weighted least squares (IRLS) / Iterative alternating sequential (IAS) methods
  - I. Daubechies, R. DeVore, M. Fornasier, and C. S. Güntürk, Iteratively reweighted least squares minimization for sparse recovery, Communications on Pure and Applied Mathematics, 63 (2010), pp. 1–38.
  - D. Calvetti, E. Somersalo, and A. Strang, Hierarchical Bayesian models and sparsity: ℓ_2-magic, Inverse Problems, 2019.

Scale mixture representations

$$\begin{aligned} \boldsymbol{y} \,|\, \boldsymbol{x} &\sim \mathcal{N}\left(\boldsymbol{F}\boldsymbol{x}, \sigma^2 \boldsymbol{I}_n\right) \\ \boldsymbol{x} &\sim \pi(\boldsymbol{x}) \end{aligned} \quad \Longleftrightarrow \quad \begin{aligned} \boldsymbol{y} \,|\, \boldsymbol{x} &\sim \mathcal{N}\left(\boldsymbol{F}\boldsymbol{x}, \sigma^2 \boldsymbol{I}_n\right) \\ \boldsymbol{x} \,|\, \boldsymbol{\theta} &\sim \mathcal{N}\left(\boldsymbol{0}, \left(\boldsymbol{R}^T \boldsymbol{D}_{\boldsymbol{\theta}}^{-1} \boldsymbol{R}\right)^{-1}\right) \\ \theta_i &\overset{\text{ind}}{\sim} \text{Hyper-prior} \end{aligned}$$

$$\pi(\boldsymbol{x}) = \int \pi(\boldsymbol{x} \,|\, \boldsymbol{\theta}) \pi(\boldsymbol{\theta}) \, d\boldsymbol{\theta}$$

Scale mixture representations

$$\boldsymbol{y} \,|\, \boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{F}\boldsymbol{x}, \sigma^2 \boldsymbol{I}_n\right)$$
$$\boldsymbol{x} \sim \pi(\boldsymbol{x})$$

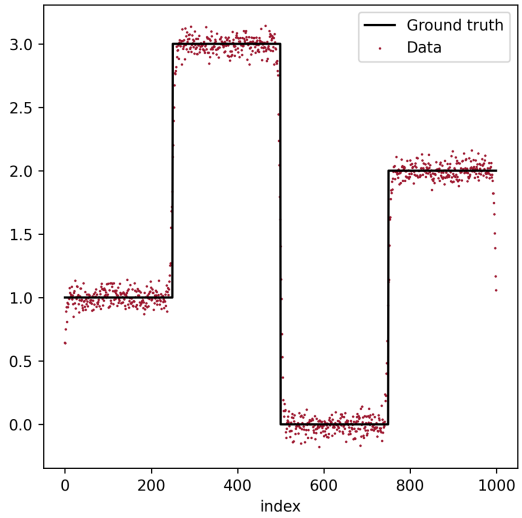
$$\Longleftrightarrow$$

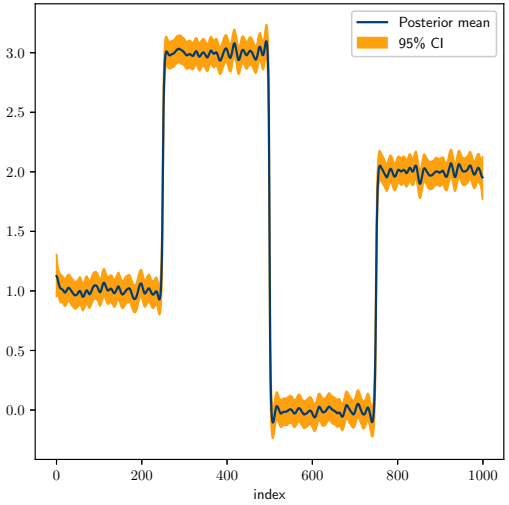
$$\boldsymbol{y} \,|\, \boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{F}\boldsymbol{x}, \sigma^2 \boldsymbol{I}_n\right)$$
$$\boldsymbol{x} \,|\, \boldsymbol{\theta} \sim \mathcal{N}\left(\boldsymbol{0}, \left(\boldsymbol{R}^T \boldsymbol{D}_{\boldsymbol{\theta}}^{-1} \boldsymbol{R}\right)^{-1}\right)$$
$$\theta_i \overset{\text{ind}}{\sim} \text{Hyper-prior}$$

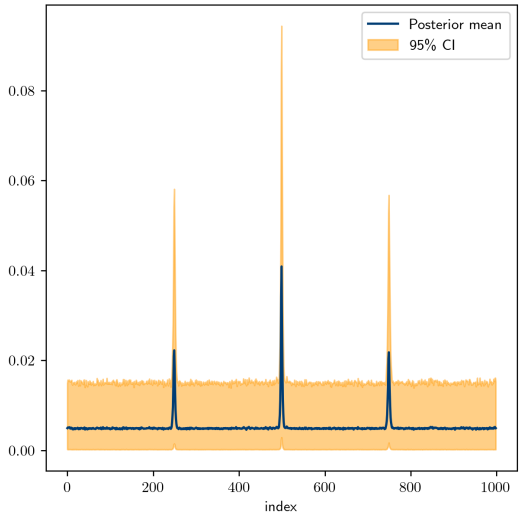
$$\pi(\boldsymbol{x}) = \int \pi(\boldsymbol{x} \,|\, \boldsymbol{\theta}) \pi(\boldsymbol{\theta}) \, d\boldsymbol{\theta}$$

Why scale mixtures?

- We obtain a conditionally-Gaussian representation of the prior, which is computationally convenient
- In the optimization context, this allows us to use simple IRLS/IAS coordinate descent algorithms where the major expense is solving a least squares problem
- In the sampling context, drawing from the unknown conditioned on hyper-parameters also amounts to solving a least squares problem (useful for Gibbs sampling)


Ground truth
Data
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
200
400
600
800
1000
index


Posterior mean
95% CI
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
200
400
600
800
1000
index


Posterior mean
95% CI
0.08
0.06
0.04
0.02
0.00
0
200
400
600
800
1000
index

Generalized Gamma hyper-prior

$$\pi_{\mathrm{GG}}(\theta \mid r, \beta, \vartheta) \propto \theta^{r\beta-1} \exp\left\{-\left(\frac{\theta}{\vartheta}\right)^{r}\right\} \mathbb{1}_{\mathbb{R}_+}(\theta)$$

- Defined for $r \in \mathbb{R} \setminus \{0\}, \beta > 0, \vartheta > 0$

Generalized Gamma hyper-prior

$$\pi_{\mathrm{GG}}(\theta \mid r, \beta, \vartheta) \propto \theta^{r\beta-1} \exp\left\{-\left(\frac{\theta}{\vartheta}\right)^{r}\right\} \mathbb{1}_{\mathbb{R}_+}(\theta)$$

- Defined for $r \in \mathbb{R} \setminus \{0\}, \beta > 0, \vartheta > 0$
- Encompasses exponential, Gamma, inverse Gamma distributions

Generalized Gamma hyper-prior

$$\pi_{\mathrm{GG}}(\theta \mid r, \beta, \vartheta) \propto \theta^{r\beta-1} \exp\left\{-\left(\frac{\theta}{\vartheta}\right)^{r}\right\} \mathbb{1}_{\mathbb{R}_+}(\theta)$$

- Defined for $r \in \mathbb{R} \setminus \{0\}, \beta > 0, \vartheta > 0$
- Encompasses exponential, Gamma, inverse Gamma distributions
- Parameters (r, s) set the parametric family of marginal prior, ϑ controls scale

Generalized Gamma hyper-prior

$$\pi_{\mathrm{GG}}(\theta \mid r, \beta, \vartheta) \propto \theta^{r\beta-1} \exp\left\{ -\left(\frac{\theta}{\vartheta}\right)^{r} \right\} \mathbb{1}_{\mathbb{R}_+}(\theta)$$

- Defined for $r \in \mathbb{R} \setminus \{0\}, \beta > 0, \vartheta > 0$
- Encompasses exponential, Gamma, inverse Gamma distributions
- Parameters (r, s) set the parametric family of marginal prior, ϑ controls scale
- In certain limits, captures ℓ_p-norm priors for $0 < p < 2$, Student-t, Cauchy

Hierarchical representation of ℓ_1 prior

$$\boldsymbol{x}^\star = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\arg\min}\ \frac{1}{2\sigma^2}\|\boldsymbol{F}\boldsymbol{x}-\boldsymbol{y}\|_2^2 + \lambda\|\boldsymbol{R}\boldsymbol{x}\|_1$$

$$\Updownarrow$$

$$(\boldsymbol{x}^\star, \boldsymbol{\theta}^\star) \overset{\eta\to 0}{=} \underset{\boldsymbol{x},\,\boldsymbol{\theta}}{\arg\min}\ \frac{1}{2\sigma^2}\|\boldsymbol{F}\boldsymbol{x}-\boldsymbol{y}\|_2^2 + \frac{1}{2}\|\boldsymbol{D}_{\boldsymbol{\theta}}^{-1/2}\boldsymbol{R}\boldsymbol{x}\|_2^2 + \frac{1}{2}\sum_{i=1}^{k}\log\theta_i - \log\pi(\boldsymbol{\theta})$$

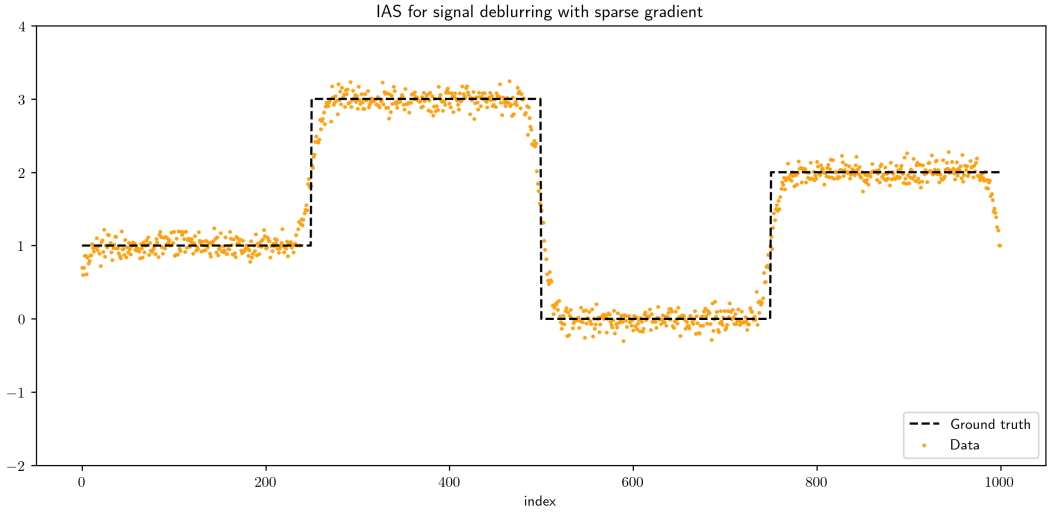
D. Calvetti, E. Somersalo, and A. Strang, Hierarchical Bayesian models and sparsity: ℓ_2-magic, Inverse Problems, 2019.

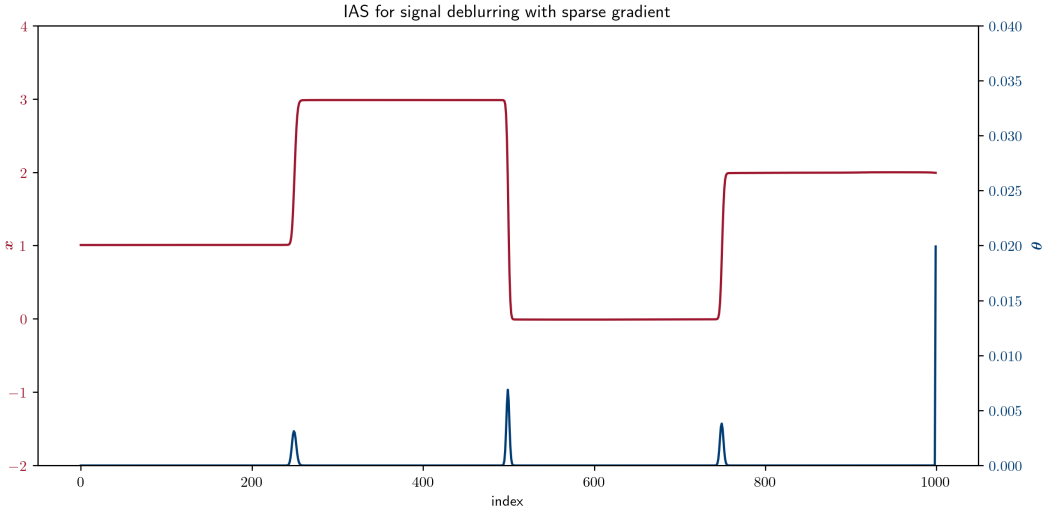
Block coordinate descent (IAS) for MAP estimation

IAS Algorithm

1: **Input:** Data $\boldsymbol{y}$, noise variance σ^2, forward operator $\boldsymbol{F}$, sparsifying transformation $\boldsymbol{R}$, hyper-parameters $(r, \beta, \boldsymbol{\vartheta})$, and initialization $\boldsymbol{x}^{(0)}$
2: **Output:** Approximate MAP estimate $(\boldsymbol{x}^{\mathrm{MAP}}, \boldsymbol{\theta}^{\mathrm{MAP}})$ for the joint posterior $\pi(\boldsymbol{x}, \boldsymbol{\theta}|\boldsymbol{y})$
3: **repeat**
4: Update the hyper-parameters $\boldsymbol{\theta}$ by $\boldsymbol{\theta}^{(k+1)} = \arg\max_{\boldsymbol{\theta}} \pi(\boldsymbol{x}^{(k)}, \boldsymbol{\theta} \mid \boldsymbol{y}) = \arg\min_{\boldsymbol{\theta}} \mathcal{G}(\boldsymbol{x}^{(k)}, \boldsymbol{\theta})$
5: Update the parameter vector $\boldsymbol{x}$ by $\boldsymbol{x}^{(k+1)} = \arg\max_{\boldsymbol{x}} \pi(\boldsymbol{x}, \boldsymbol{\theta}^{(k+1)} \mid \boldsymbol{y}) = \arg\min_{\boldsymbol{x}} \mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta}^{(k+1)})$
6: **until** convergence or the maximum number of iterations is reached

$$\mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta}) = \frac{1}{2\sigma^2}\|\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \frac{1}{2}\|\boldsymbol{D}_{\boldsymbol{\theta}}^{-1/2}\boldsymbol{R}\mathbf{x}\|_2^2 + \sum_{i=1}^{k}\left(\frac{\theta_i}{\vartheta_i}\right)^r - \sum_{i=1}^{k}(r\beta - 3/2)\log(\theta_i) + \delta_{\mathbb{R}_+^N}(\boldsymbol{\theta})$$


IAS for signal deblurring with sparse gradient
4
3
2
1
0
−1
−2
0
200
400
600
800
1000
index
Ground truth
Data


IAS for signal deblurring with sparse gradient
4
3
2
1
0
−1
−2
x
0.040
0.035
0.030
0.025
0.020
0.015
0.010
0.005
0.000
θ
0
200
400
600
800
1000
index

GENERALIZATIONS

Convexity

Previous convexity result

Let $\eta \coloneqq r\beta - \frac{3}{2}$ and $\boldsymbol{R} = \boldsymbol{I}_n$. Then $(\boldsymbol{x}, \boldsymbol{\theta})$ is globally strict convex if $r \geq 1$ and $\eta > 0$, and is locally convex at $(\boldsymbol{x}, \boldsymbol{\theta})$ if either (i) $0 < r < 1$ and $\eta > 0$ or (ii) $r < 0$ and

$$\theta_i < \vartheta_i \left(\eta/(r|r-1|)\right)^{1/r}, \quad i = 1, \ldots, n.$$

Convexity with general $\boldsymbol{R}$

The previous result holds for any $\boldsymbol{R}$ so long as $\ker(\boldsymbol{F}) \cap \ker(\boldsymbol{R}) = \{\boldsymbol{0}_n\}$.

Convexity

Previous convexity result

Let $\eta \coloneqq r\beta - \frac{3}{2}$ and $\boldsymbol{R} = \boldsymbol{I}_n$. Then $(\boldsymbol{x}, \boldsymbol{\theta})$ is globally strict convex if $r \geq 1$ and $\eta > 0$, and is locally convex at $(\boldsymbol{x}, \boldsymbol{\theta})$ if either (i) $0 < r < 1$ and $\eta > 0$ or (ii) $r < 0$ and

$$\theta_i < \vartheta_i \left(\eta/(r|r-1|)\right)^{1/r}, \quad i = 1, \ldots, n.$$

Convexity with general $\boldsymbol{R}$

The previous result holds for any $\boldsymbol{R}$ so long as $\ker(\boldsymbol{F}) \cap \ker(\boldsymbol{R}) = \{\boldsymbol{0}_n\}$.

Convergence

Previous convergence result

When $r = 1$ and $\eta > 0$, IAS converges to the unique global minimizer of $\mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta})$.

General convex case

When $r \geq 1$ and $\eta > 0$, IAS converges to the unique global minimizer of $\mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta})$.

Nonconvex case

For any valid hyper-prior parameters $(r, \beta, \boldsymbol{\vartheta})$, the sequence $\{\boldsymbol{x}^{(k)}, \boldsymbol{\theta}^{(k)}\}$ is bounded and any limit point must be a stationary point of $\mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta})$.

Convergence

Previous convergence result

When $r = 1$ and $\eta > 0$, IAS converges to the unique global minimizer of $\mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta})$.

General convex case

When $r \geq 1$ and $\eta > 0$, IAS converges to the unique global minimizer of $\mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta})$.

Nonconvex case

For any valid hyper-prior parameters $(r, \beta, \boldsymbol{\vartheta})$, the sequence $\{\boldsymbol{x}^{(k)}, \boldsymbol{\theta}^{(k)}\}$ is bounded and any limit point must be a stationary point of $\mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta})$.

Learning noise variance

- In some settings, we may not have a good estimate of the noise variance σ^2
- Could try initial reconstruction $\boldsymbol{x}$ to estimate σ^2, but not always feasible
- Can adopt an "uninformative" prior for the noise variance

$$\boldsymbol{y} \,|\, \boldsymbol{x}, \nu \sim \mathcal{N}\left(\boldsymbol{F}\boldsymbol{x}, \nu \boldsymbol{I}_n\right)$$

$$\boldsymbol{x} \,|\, \boldsymbol{\theta} \sim \mathcal{N}\left(\boldsymbol{0}, \left(\boldsymbol{R}^T \boldsymbol{D}_{\boldsymbol{\theta}}^{-1} \boldsymbol{R}\right)^{-1}\right)$$

$$\theta_i \overset{\text{ind}}{\sim} \mathcal{GG}(r, \beta, \vartheta_i)$$

$$\nu \sim \mathcal{GG}(\tilde{r}, \tilde{\beta}, \tilde{\vartheta})$$

Generalized IAS Algorithm

1: **Input:** Data $\boldsymbol{y}$, forward operator $\boldsymbol{F}$, sparsifying transformation $\boldsymbol{R}$, hyper-parameters (r, β, ϑ) and $(\tilde{r}, \tilde{\beta}, \tilde{\vartheta})$, initialization $\boldsymbol{x}^{(0)}$
2: **Output:** Approximate MAP estimate $(\boldsymbol{x}^{\mathrm{MAP}}, \boldsymbol{\theta}^{\mathrm{MAP}}, \nu^{\mathrm{MAP}})$ for the joint posterior $\pi(\boldsymbol{x}, \boldsymbol{\theta}, \nu | \boldsymbol{y})$
3: **repeat**
4: Update $\boldsymbol{\theta}$ by $\boldsymbol{\theta}^{(k+1)} = \arg\max_{\boldsymbol{\theta}} \pi(\boldsymbol{x}^{(k)}, \boldsymbol{\theta}, \nu^{(k)} \mid \boldsymbol{y}) = \arg\min_{\boldsymbol{\theta}} \mathcal{G}(\boldsymbol{x}^{(k)}, \boldsymbol{\theta}, \nu^{(k)})$
5: Update ν by $\nu = \arg\max_{\nu} \pi(\boldsymbol{x}^{(k)}, \boldsymbol{\theta}^{(k+1)}, \nu \mid \boldsymbol{y}) = \arg\min_{\nu} \mathcal{G}(\boldsymbol{x}^{(k)}, \boldsymbol{\theta}^{(k+1)}, \nu)$
6: Update $\boldsymbol{x}$ by $\boldsymbol{x}^{(k+1)} = \arg\max_{\boldsymbol{x}} \pi(\boldsymbol{x}, \boldsymbol{\theta}^{(k+1)}, \nu^{(k+1)} \mid \boldsymbol{y}) = \arg\min_{\boldsymbol{x}} \mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta}^{(k+1)}, \nu^{(k+1)})$
7: **until** convergence or the maximum number of iterations is reached

$$\mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta}, \nu) = \frac{1}{2\nu}\|\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \frac{1}{2}\|\boldsymbol{D}_{\boldsymbol{\theta}}^{-1/2}\boldsymbol{R}\boldsymbol{x}\|_2^2 + \left(\frac{\nu}{\tilde{\vartheta}}\right)^{\tilde{r}} + \sum_{i=1}^{k}\left(\frac{\theta_i}{\vartheta_i}\right)^{r}$$

$$- \left(\tilde{r}\tilde{\beta} - [m+2]/2\right)\log(\nu) - \left(r\beta - 3/2\right)\sum_{i=1}^{k}\log(\theta_i) + \delta_{\mathbb{R}_+^{k+1}}(\boldsymbol{\theta}, \nu),$$

Convexity

Convexity result

Let $\eta = r\beta - 3/2$, $\tilde{\eta} = \tilde{r}\tilde{\beta} - (m+2)/2$. Then $\mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta}, \nu)$ is globally strictly convex if $r, \tilde{r} \geq 1$ and $\eta, \tilde{\eta} > 0$.

- The use of an uninformative noise variance prior such as $r = -1$, $\beta = 1$, $\vartheta \approx 0$ generally leads to a nonconvex objective

Convexity

Convexity result

Let $\eta = r\beta - 3/2$, $\tilde{\eta} = \tilde{r}\tilde{\beta} - (m+2)/2$. Then $\mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta}, \nu)$ is globally strictly convex if $r, \tilde{r} \geq 1$ and $\eta, \tilde{\eta} > 0$.

- The use of an uninformative noise variance prior such as $r = -1$, $\beta = 1$, $\vartheta \approx 0$ generally leads to a nonconvex objective

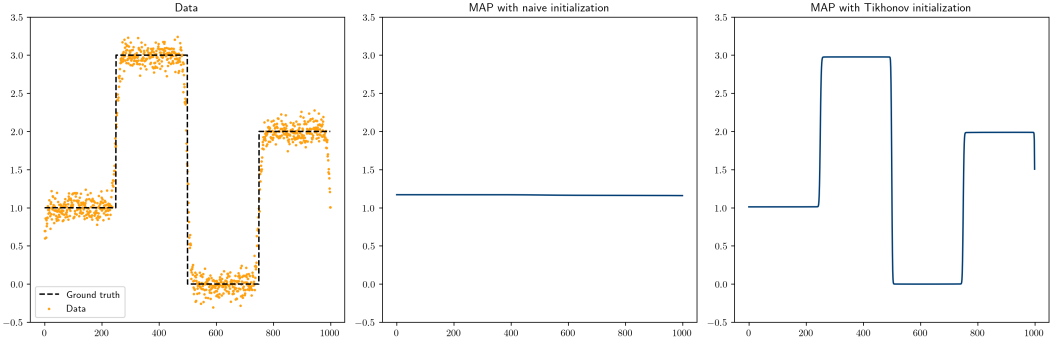
Convergence

General convex case

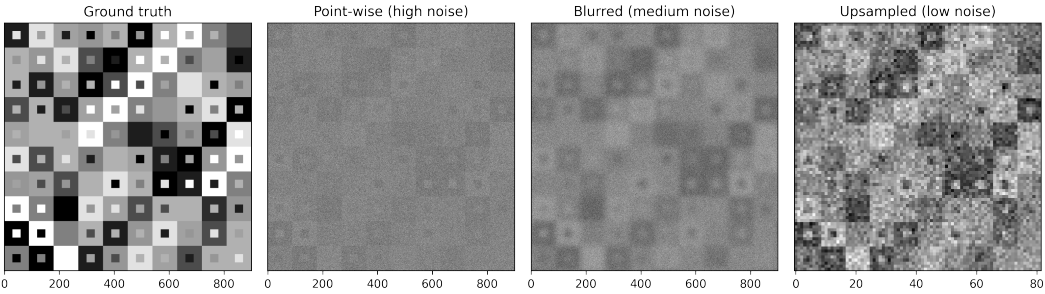
When $r, \tilde{r} \geq 1$ and $\eta, \tilde{\eta} > 0$, generalized IAS converges to the unique global minimizer of $\mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta}, \nu)$.

Nonconvex case

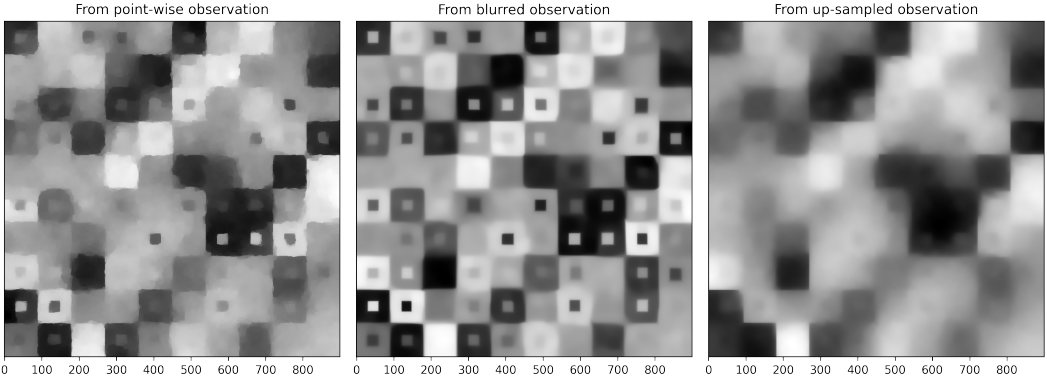
For any valid hyper-prior parameters $(r, \beta, \boldsymbol{\vartheta})$, the sequence $\{\boldsymbol{x}^{(k)}, \boldsymbol{\theta}^{(k)}, \nu^{(k)}\}$ is bounded and any limit point must be a stationary point of $\mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta}, \nu)$.


Data
Ground truth
Data
MAP with naive initialization
MAP with Tikhonov initialization

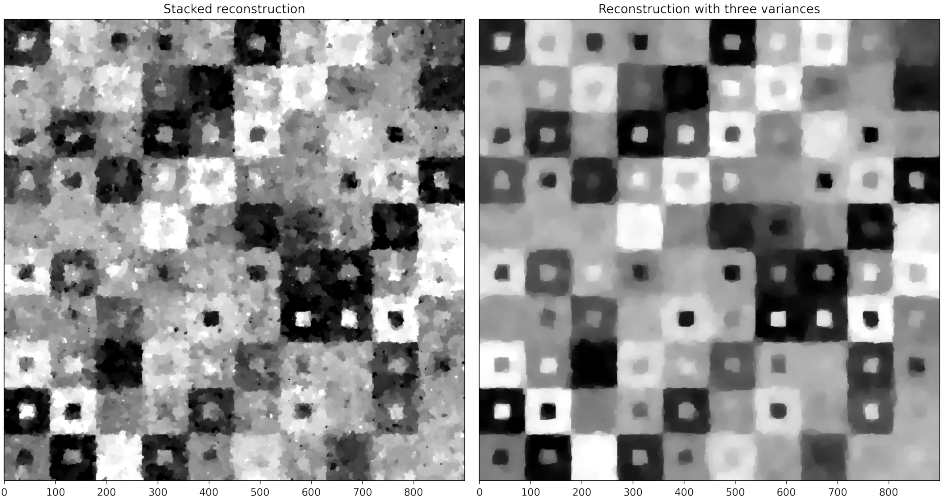
Data fusion


Ground truth
Point-wise (high noise)
Blurred (medium noise)
Upsampled (low noise)
0
200
400
600
800
0
200
400
600
800
0
200
400
600
800
0
20
40
60
80

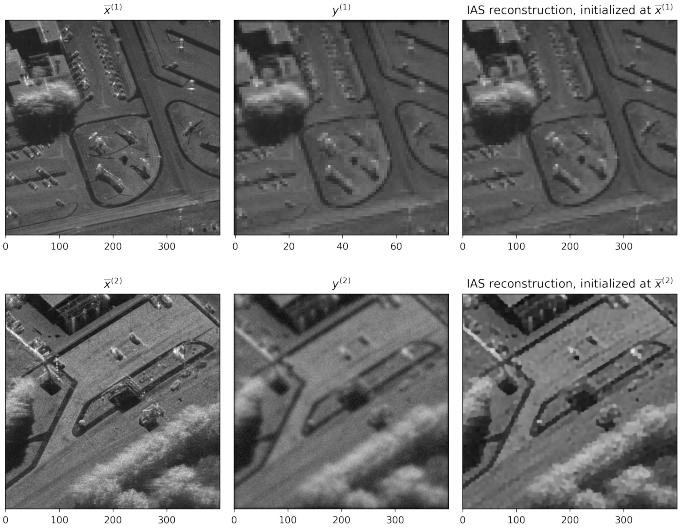
Data fusion


From point-wise observation
From blurred observation
From up-sampled observation
0 100 200 300 400 500 600 700 800
0 100 200 300 400 500 600 700 800
0 100 200 300 400 500 600 700 800

Data fusion


Stacked reconstruction
Reconstruction with three variances
0
100
200
300
400
500
600
700
800
0
100
200
300
400
500
600
700
800

Data fusion


$\overline{x}^{(1)}$
$y^{(1)}$
IAS reconstruction, initialized at $\overline{x}^{(1)}$
$\overline{x}^{(2)}$
$y^{(2)}$
IAS reconstruction, initialized at $\overline{x}^{(2)}$

PRIORCONDITIONING

The most computationally intensive step of the IAS method is the solution of quadratic optimization problems of the form

$$\boldsymbol{x}^\star = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\arg\min} \ \|\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \|\boldsymbol{D}_{\boldsymbol{\theta}}^{-1/2}\boldsymbol{R}\boldsymbol{x}\|_2^2,$$

which is given explicitly by

$$\boldsymbol{x}^\star = \left(\boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R}\right)^{-1}\boldsymbol{F}^T\boldsymbol{y}.$$

or equivalently as the solution of the least squares problems

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{D}_{\boldsymbol{\theta}}^{-1/2}\boldsymbol{R} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_k \end{bmatrix}.$$

Usually, we must use a Krylov iterative method such as CGLS.

The most computationally intensive step of the IAS method is the solution of quadratic optimization problems of the form

$$\boldsymbol{x}^\star = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\arg\min} \ \|\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \|\boldsymbol{D}_{\boldsymbol{\theta}}^{-1/2}\boldsymbol{R}\boldsymbol{x}\|_2^2,$$

which is given explicitly by

$$\boldsymbol{x}^\star = \left(\boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R}\right)^{-1}\boldsymbol{F}^T\boldsymbol{y}.$$

or equivalently as the solution of the least squares problems

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{D}_{\boldsymbol{\theta}}^{-1/2}\boldsymbol{R} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_k \end{bmatrix}.$$

Usually, we must use a Krylov iterative method such as CGLS.

The most computationally intensive step of the IAS method is the solution of quadratic optimization problems of the form

$$\boldsymbol{x}^\star = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\arg\min} \ \|\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y}\|_2^2 + \|\boldsymbol{D}_{\boldsymbol{\theta}}^{-1/2}\boldsymbol{R}\boldsymbol{x}\|_2^2,$$

which is given explicitly by

$$\boldsymbol{x}^\star = \left(\boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R}\right)^{-1}\boldsymbol{F}^T\boldsymbol{y}.$$

or equivalently as the solution of the least squares problems

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{D}_{\boldsymbol{\theta}}^{-1/2}\boldsymbol{R} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_k \end{bmatrix}.$$

Usually, we must use a Krylov iterative method such as CGLS.

CGLS Algorithm

Input: Matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ with $\text{rank}(\boldsymbol{A}) = n$, RHS vector $\boldsymbol{b} \in \mathbb{R}^n$, initialization $\boldsymbol{x}^{(0)}$
Output: Approximation $\boldsymbol{x}^\star$ to least squares solution of $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$.
$\boldsymbol{d}^{(0)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{(0)}$
$\boldsymbol{r}^{(0)} = \boldsymbol{A}^T\boldsymbol{d}^{(0)}$
$\boldsymbol{p}^{(0)} = \boldsymbol{r}^{(0)}$
$\boldsymbol{y}^{(0)} = \boldsymbol{A}\boldsymbol{p}^{(0)}$
$k = 1$
repeat
  $\alpha^{(k)} = \|\boldsymbol{r}^{(k-1)}\|_2^2 / \|\boldsymbol{y}^{(k-1)}\|_2^2$
  $\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} + \alpha^{(k)}\boldsymbol{p}^{(k-1)}$
  $\boldsymbol{d}^{(k)} = \boldsymbol{d}^{(k-1)} - \alpha^{(k)}\boldsymbol{y}^{(k-1)}$
  $\boldsymbol{r}^{(k)} = \boldsymbol{A}^T\boldsymbol{d}^{(k)}$
  $\beta^{(k)} = \|\boldsymbol{r}^{(k)}\|_2^2 / \|\boldsymbol{r}^{(k-1)}\|_2^2$
  $\boldsymbol{p}^{(k)} = \boldsymbol{r}^{(k)} + \beta^{(k)}\boldsymbol{p}^{(k-1)}$
  $\boldsymbol{y}^{(k)} = \boldsymbol{A}\boldsymbol{p}^{(k)}$
  $k = k + 1$
until convergence or the maximum number of iterations is reached
return $\boldsymbol{x}^\star$

Preconditioning can improve efficiency. One such technique is called priorconditioning.
Let $\boldsymbol{R_\theta} = \boldsymbol{D_\theta}^{-1/2}\boldsymbol{R}$.

If $\boldsymbol{R}^{-1}$ exists:

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{R_\theta} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_n \end{bmatrix} \quad \begin{matrix} \boldsymbol{x} = \boldsymbol{R_\theta}^{-1}\boldsymbol{w} \\ \rightarrow \end{matrix} \quad \begin{bmatrix} \boldsymbol{F}\boldsymbol{R_\theta}^{-1} \\ \boldsymbol{I}_n \end{bmatrix} \boldsymbol{w} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_n \end{bmatrix}$$

If $\ker(\boldsymbol{R}) = \{\boldsymbol{0}\}$:

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{R_\theta} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_n \end{bmatrix} \quad \begin{matrix} \boldsymbol{x} = \boldsymbol{R_\theta}^{\dagger}\boldsymbol{w} \\ \rightarrow \end{matrix} \quad \begin{bmatrix} \boldsymbol{F}\boldsymbol{R_\theta}^{\dagger} \\ \boldsymbol{I}_k \end{bmatrix} \boldsymbol{w} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_k \end{bmatrix}$$

$\star$ What about if $\ker(\boldsymbol{R}) \neq \{\boldsymbol{0}\}$?

Preconditioning can improve efficiency. One such technique is called priorconditioning.
Let $\boldsymbol{R_\theta} = \boldsymbol{D_\theta}^{-1/2}\boldsymbol{R}$.

If $\boldsymbol{R}^{-1}$ exists:

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{R_\theta} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_n \end{bmatrix} \quad \begin{matrix} \boldsymbol{x} = \boldsymbol{R_\theta}^{-1}\boldsymbol{w} \\ \rightarrow \end{matrix} \quad \begin{bmatrix} \boldsymbol{F}\boldsymbol{R_\theta}^{-1} \\ \boldsymbol{I}_n \end{bmatrix} \boldsymbol{w} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_n \end{bmatrix}$$

If $\ker(\boldsymbol{R}) = \{\boldsymbol{0}\}$:

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{R_\theta} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_n \end{bmatrix} \quad \begin{matrix} \boldsymbol{x} = \boldsymbol{R_\theta}^{\dagger}\boldsymbol{w} \\ \rightarrow \end{matrix} \quad \begin{bmatrix} \boldsymbol{F}\boldsymbol{R_\theta}^{\dagger} \\ \boldsymbol{I}_k \end{bmatrix} \boldsymbol{w} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_k \end{bmatrix}$$

★ What about if $\ker(\boldsymbol{R}) \neq \{\boldsymbol{0}\}$?

Preconditioning can improve efficiency. One such technique is called priorconditioning.
Let $\boldsymbol{R_\theta} = \boldsymbol{D_\theta}^{-1/2}\boldsymbol{R}$.

If $\boldsymbol{R}^{-1}$ exists:

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{R_\theta} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_n \end{bmatrix} \quad \begin{matrix} \boldsymbol{x} = \boldsymbol{R_\theta}^{-1}\boldsymbol{w} \\ \rightarrow \end{matrix} \quad \begin{bmatrix} \boldsymbol{F}\boldsymbol{R_\theta}^{-1} \\ \boldsymbol{I}_n \end{bmatrix} \boldsymbol{w} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_n \end{bmatrix}$$

If $\ker(\boldsymbol{R}) = \{\boldsymbol{0}\}$:

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{R_\theta} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_n \end{bmatrix} \quad \begin{matrix} \boldsymbol{x} = \boldsymbol{R_\theta}^{\dagger}\boldsymbol{w} \\ \rightarrow \end{matrix} \quad \begin{bmatrix} \boldsymbol{F}\boldsymbol{R_\theta}^{\dagger} \\ \boldsymbol{I}_k \end{bmatrix} \boldsymbol{w} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_k \end{bmatrix}$$

★ What about if $\ker(\boldsymbol{R}) \neq \{\boldsymbol{0}\}$?

Preconditioning can improve efficiency. One such technique is called priorconditioning.
Let $\boldsymbol{R_\theta} = \boldsymbol{D_\theta}^{-1/2}\boldsymbol{R}$.

If $\boldsymbol{R}^{-1}$ exists:

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{R_\theta} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_n \end{bmatrix} \underset{\rightarrow}{\boldsymbol{x} = \boldsymbol{R_\theta}^{-1}\boldsymbol{w}} \begin{bmatrix} \boldsymbol{F}\boldsymbol{R_\theta}^{-1} \\ \boldsymbol{I}_n \end{bmatrix} \boldsymbol{w} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_n \end{bmatrix}$$

If $\ker(\boldsymbol{R}) = \{\boldsymbol{0}\}$:

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{R_\theta} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_n \end{bmatrix} \underset{\rightarrow}{\boldsymbol{x} = \boldsymbol{R_\theta}^{\dagger}\boldsymbol{w}} \begin{bmatrix} \boldsymbol{F}\boldsymbol{R_\theta}^{\dagger} \\ \boldsymbol{I}_k \end{bmatrix} \boldsymbol{w} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_k \end{bmatrix}$$

★ What about if $\ker(\boldsymbol{R}) \neq \{\boldsymbol{0}\}$?

$$\boldsymbol{R} = \begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix} \qquad \tilde{\boldsymbol{R}} = \begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \\ & & & -1 \end{bmatrix}$$

For sparsifying transformations with non-trivial kernels, we can use the oblique pseudoinverse, which for $\boldsymbol{W}$ s.t. $\ker(\boldsymbol{R}) = \mathrm{col}(\boldsymbol{W})$ is given by

$$\boldsymbol{R}_{\boldsymbol{\theta}}^{\#} = \left(\boldsymbol{I}_n - \boldsymbol{W}(\boldsymbol{F}\boldsymbol{W})^{\dagger}\boldsymbol{F}\right)\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger}$$

We make the change of variables

$$\begin{bmatrix}\boldsymbol{F}\\ \boldsymbol{R}_{\boldsymbol{\theta}}\end{bmatrix}\boldsymbol{x} = \begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{0}_k\end{bmatrix} \quad \overset{\boldsymbol{x} = \boldsymbol{R}_{\boldsymbol{\theta}}^{\#}\boldsymbol{w}}{\rightarrow} \quad \begin{bmatrix}\boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{\#}\\ \boldsymbol{I}_k\end{bmatrix}\boldsymbol{w} = \begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{0}_k\end{bmatrix}$$

Then, the solution is given by

$$\boldsymbol{x}^{\star} = \underbrace{\boldsymbol{R}_{\boldsymbol{\theta}}^{\#}\boldsymbol{w}^{\star}}_{\in \ker(\boldsymbol{R})^{\perp_{\boldsymbol{F}}}} + \underbrace{\boldsymbol{W}(\boldsymbol{F}\boldsymbol{W})^{\dagger}\boldsymbol{y}}_{\in \ker(\boldsymbol{R})}$$

P. C. Hansen, Discrete Inverse Problems: Insight and Algorithms, SIAM, 2010.

For sparsifying transformations with non-trivial kernels, we can use the oblique pseudoinverse, which for $\boldsymbol{W}$ s.t. $\ker(\boldsymbol{R}) = \mathrm{col}(\boldsymbol{W})$ is given by

$$\boldsymbol{R}_{\boldsymbol{\theta}}^{\#} = \left(\boldsymbol{I}_n - \boldsymbol{W}(\boldsymbol{F}\boldsymbol{W})^{\dagger}\boldsymbol{F}\right)\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger}$$

We make the change of variables

$$\begin{bmatrix}\boldsymbol{F}\\ \boldsymbol{R}_{\boldsymbol{\theta}}\end{bmatrix}\boldsymbol{x} = \begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{0}_k\end{bmatrix} \quad \begin{matrix}\boldsymbol{x} = \boldsymbol{R}_{\boldsymbol{\theta}}^{\#}\boldsymbol{w}\\ \rightarrow\end{matrix} \quad \begin{bmatrix}\boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{\#}\\ \boldsymbol{I}_k\end{bmatrix}\boldsymbol{w} = \begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{0}_k\end{bmatrix}$$

Then, the solution is given by

$$\boldsymbol{x}^{\star} = \underbrace{\boldsymbol{R}_{\boldsymbol{\theta}}^{\#}\boldsymbol{w}^{\star}}_{\in\ker(\boldsymbol{R})^{\perp_{\boldsymbol{F}}}} + \underbrace{\boldsymbol{W}(\boldsymbol{F}\boldsymbol{W})^{\dagger}\boldsymbol{y}}_{\in\ker(\boldsymbol{R})}$$

P. C. Hansen, Discrete Inverse Problems: Insight and Algorithms, SIAM, 2010.

For sparsifying transformations with non-trivial kernels, we can use the oblique pseudoinverse, which for $\boldsymbol{W}$ s.t. $\ker(\boldsymbol{R}) = \mathrm{col}(\boldsymbol{W})$ is given by

$$\boldsymbol{R}_{\boldsymbol{\theta}}^{\#} = \left(\boldsymbol{I}_n - \boldsymbol{W}(\boldsymbol{F}\boldsymbol{W})^{\dagger}\boldsymbol{F}\right)\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger}$$

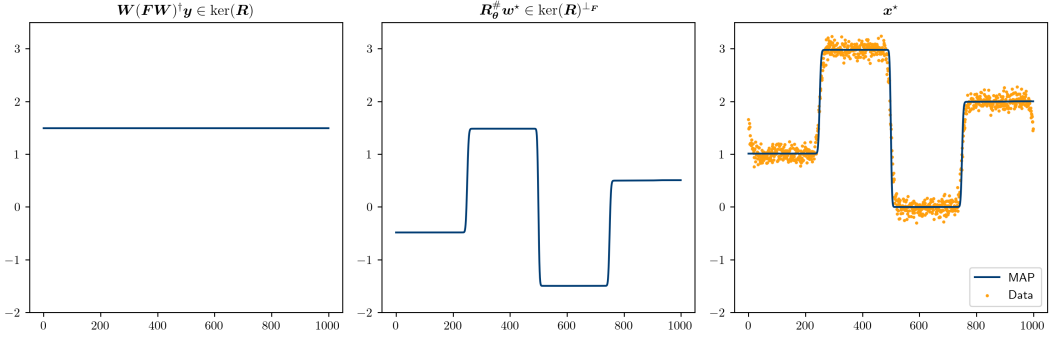
We make the change of variables

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{R}_{\boldsymbol{\theta}} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_k \end{bmatrix} \quad \begin{matrix} \boldsymbol{x} = \boldsymbol{R}_{\boldsymbol{\theta}}^{\#}\boldsymbol{w} \\ \rightarrow \end{matrix} \quad \begin{bmatrix} \boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{\#} \\ \boldsymbol{I}_k \end{bmatrix} \boldsymbol{w} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_k \end{bmatrix}$$

Then, the solution is given by

$$\boldsymbol{x}^{\star} = \underbrace{\boldsymbol{R}_{\boldsymbol{\theta}}^{\#}\boldsymbol{w}^{\star}}_{\in \ker(\boldsymbol{R})^{\perp_{\boldsymbol{F}}}} + \underbrace{\boldsymbol{W}(\boldsymbol{F}\boldsymbol{W})^{\dagger}\boldsymbol{y}}_{\in \ker(\boldsymbol{R})}$$

P. C. Hansen, Discrete Inverse Problems: Insight and Algorithms, SIAM, 2010.


$\boldsymbol{W}(\boldsymbol{F}\boldsymbol{W})^{\dagger}\boldsymbol{y} \in \ker(\boldsymbol{R})$
$\boldsymbol{R}_{\theta}^{\#}\boldsymbol{w}^{\star} \in \ker(\boldsymbol{R})^{\perp_{\boldsymbol{F}}}$
$\boldsymbol{x}^{\star}$
MAP
Data

In terms of linear systems?

If $\boldsymbol{R}^{-1}$ exists:

$$\boldsymbol{Q}_0 = \boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R} \quad \overset{\boldsymbol{x}=\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}\boldsymbol{w}}{\rightarrow} \quad \boldsymbol{Q}_l = \boldsymbol{R}_{\boldsymbol{\theta}}^{-T}\boldsymbol{F}^T\boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{-1} + \boldsymbol{I}_n$$

If $\ker(\boldsymbol{R}) = \{\boldsymbol{0}_n\}$:

$$\boldsymbol{Q}_0 = \boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R} \quad \overset{\boldsymbol{x}=\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger}\boldsymbol{w}}{\rightarrow} \quad \boldsymbol{Q}_t = (\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger})^T\boldsymbol{F}^T\boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger} + \boldsymbol{I}_k$$

If $\ker(\boldsymbol{R}) \neq \{\boldsymbol{0}_n\}$:

$$\boldsymbol{Q}_0 = \boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R} \quad \overset{\boldsymbol{x}=\boldsymbol{R}_{\boldsymbol{\theta}}^{\#}\boldsymbol{w}}{\rightarrow} \quad \boldsymbol{Q}_t = (\boldsymbol{R}_{\boldsymbol{\theta}}^{\#})^T\boldsymbol{F}^T\boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{\#} + \boldsymbol{I}_k$$

In terms of linear systems?

If $\boldsymbol{R}^{-1}$ exists:

$$\boldsymbol{Q}_0 = \boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R} \quad \overset{\boldsymbol{x}=\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}\boldsymbol{w}}{\rightarrow} \quad \boldsymbol{Q}_t = \boldsymbol{R}_{\boldsymbol{\theta}}^{-T}\boldsymbol{F}^T\boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{-1} + \boldsymbol{I}_n$$

If $\ker(\boldsymbol{R}) = \{\boldsymbol{0}_n\}$:

$$Q_0 = \boldsymbol{F}^T\boldsymbol{F} + R^T D_{\boldsymbol{\theta}}^{-1} R \quad \overset{x=R_{\boldsymbol{\theta}}^{\dagger} w}{\rightarrow} \quad Q_t = (R_{\boldsymbol{\theta}}^{\dagger})^T \boldsymbol{F}^T \boldsymbol{F} R_{\boldsymbol{\theta}}^{\dagger} + I_k$$

If $\ker(\boldsymbol{R}) \neq \{\boldsymbol{0}_n\}$:

$$Q_0 = \boldsymbol{F}^T\boldsymbol{F} + R^T D_{\boldsymbol{\theta}}^{-1} R \quad \overset{x=R_{\boldsymbol{\theta}}^{\#} w}{\rightarrow} \quad Q_t = (R_{\boldsymbol{\theta}}^{\#})^T \boldsymbol{F}^T \boldsymbol{F} R_{\boldsymbol{\theta}}^{\#} + I_k$$

In terms of linear systems?

If $\boldsymbol{R}^{-1}$ exists:

$$\boldsymbol{Q}_0 = \boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R} \quad \overset{\boldsymbol{x}=\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}\boldsymbol{w}}{\rightarrow} \quad \boldsymbol{Q}_t = \boldsymbol{R}_{\boldsymbol{\theta}}^{-T}\boldsymbol{F}^T\boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{-1} + \boldsymbol{I}_n$$

If $\ker(\boldsymbol{R}) = \{\boldsymbol{0}_n\}$:

$$\boldsymbol{Q}_0 = \boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R} \quad \overset{\boldsymbol{x}=\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger}\boldsymbol{w}}{\rightarrow} \quad \boldsymbol{Q}_t = (\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger})^T\boldsymbol{F}^T\boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger} + \boldsymbol{I}_k$$

If $\ker(\boldsymbol{R}) \neq \{\boldsymbol{0}_n\}$:

$$\boldsymbol{Q}_0 = \boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R} \quad \overset{\boldsymbol{x}=\boldsymbol{R}_{\boldsymbol{\theta}}^{\#}\boldsymbol{w}}{\rightarrow} \quad \boldsymbol{Q}_t = (\boldsymbol{R}_{\boldsymbol{\theta}}^{\#})^T\boldsymbol{F}^T\boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{\#} + \boldsymbol{I}_k$$

In terms of linear systems?

If $\boldsymbol{R}^{-1}$ exists:

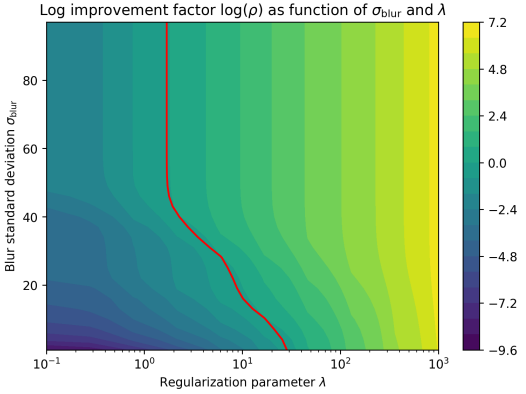
$$\boldsymbol{Q}_0 = \boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R} \quad \overset{\boldsymbol{x}=\boldsymbol{R}_{\boldsymbol{\theta}}^{-1}\boldsymbol{w}}{\rightarrow} \quad \boldsymbol{Q}_t = \boldsymbol{R}_{\boldsymbol{\theta}}^{-T}\boldsymbol{F}^T\boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{-1} + \boldsymbol{I}_n$$

If $\ker(\boldsymbol{R}) = \{\boldsymbol{0}_n\}$:

$$\boldsymbol{Q}_0 = \boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R} \quad \overset{\boldsymbol{x}=\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger}\boldsymbol{w}}{\rightarrow} \quad \boldsymbol{Q}_t = (\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger})^T\boldsymbol{F}^T\boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger} + \boldsymbol{I}_k$$

If $\ker(\boldsymbol{R}) \neq \{\boldsymbol{0}_n\}$:

$$\boldsymbol{Q}_0 = \boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}^T\boldsymbol{D}_{\boldsymbol{\theta}}^{-1}\boldsymbol{R} \quad \overset{\boldsymbol{x}=\boldsymbol{R}_{\boldsymbol{\theta}}^{\#}\boldsymbol{w}}{\rightarrow} \quad \boldsymbol{Q}_t = (\boldsymbol{R}_{\boldsymbol{\theta}}^{\#})^T\boldsymbol{F}^T\boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{\#} + \boldsymbol{I}_k$$



Consider the condition number improvement factor

$$\rho = \frac{\kappa\left(\boldsymbol{Q}_0\right)}{\kappa\left(\boldsymbol{Q}_t\right)}$$

- $\rho > 1 \Rightarrow$ priorconditioning gives a good preconditioner
- $\rho = 1 \Rightarrow$ indifferent
- $\rho < 1 \Rightarrow$ priorconditioning worsens the conditioning

Implementing pseudoinverses

To implement prior conditioning efficiently, we need a good way to compute matvecs and rmatvecs with

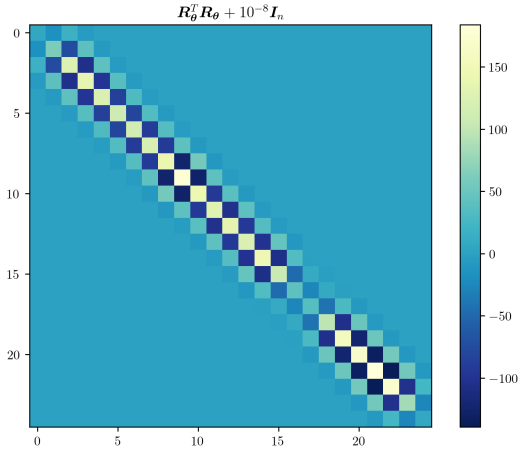
$$\boldsymbol{R}_{\boldsymbol{\theta}}^{\#} = \left(\boldsymbol{I}_n - \boldsymbol{W}(\boldsymbol{F}\boldsymbol{W})^{\dagger}\boldsymbol{F}\right)\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger}$$

- In typical cases $\dim(\ker(\boldsymbol{R}))$ is small, so $\boldsymbol{W}$ is "skinny" and $(\boldsymbol{F}\boldsymbol{W})^{\dagger}$ can be computed quickly using a QR factorization (and only once)
- However, the QR approach is very costly for $\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger}$ since it must be re-computed for each $\boldsymbol{\theta}$, and $\boldsymbol{R}_{\boldsymbol{\theta}} \in \mathbb{R}^{k\times n}$ with $k \approx n$ (or larger)

One strategy is to use the approximations

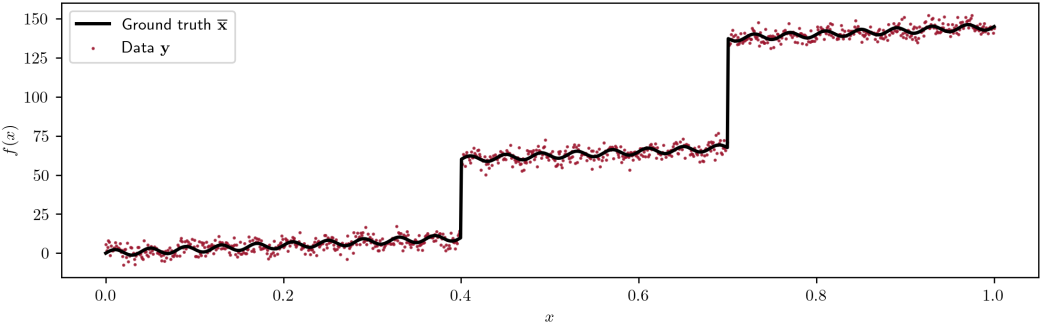
$$\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger} \approx (\boldsymbol{R}_{\boldsymbol{\theta}}^T \boldsymbol{R}_{\boldsymbol{\theta}} + \delta \boldsymbol{I}_n)^{-1} \boldsymbol{R}_{\boldsymbol{\theta}}^T,$$
$$(\boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger})^T \approx \boldsymbol{R}_{\boldsymbol{\theta}} (\boldsymbol{R}_{\boldsymbol{\theta}}^T \boldsymbol{R}_{\boldsymbol{\theta}} + \delta \boldsymbol{I}_n)^{-1}$$

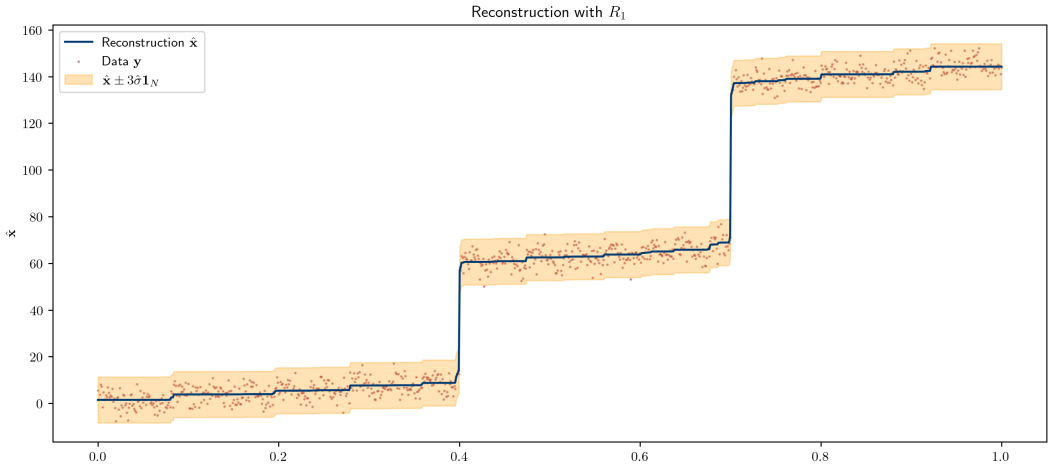
- If $\boldsymbol{R}_{\boldsymbol{\theta}}^T \boldsymbol{R}_{\boldsymbol{\theta}}$ has bandwidth p, costs $\mathcal{O}(p^2 n)$ flops using a sparse Cholesky routine 👍
- Re-computed for each new $\boldsymbol{\theta}$ 👎

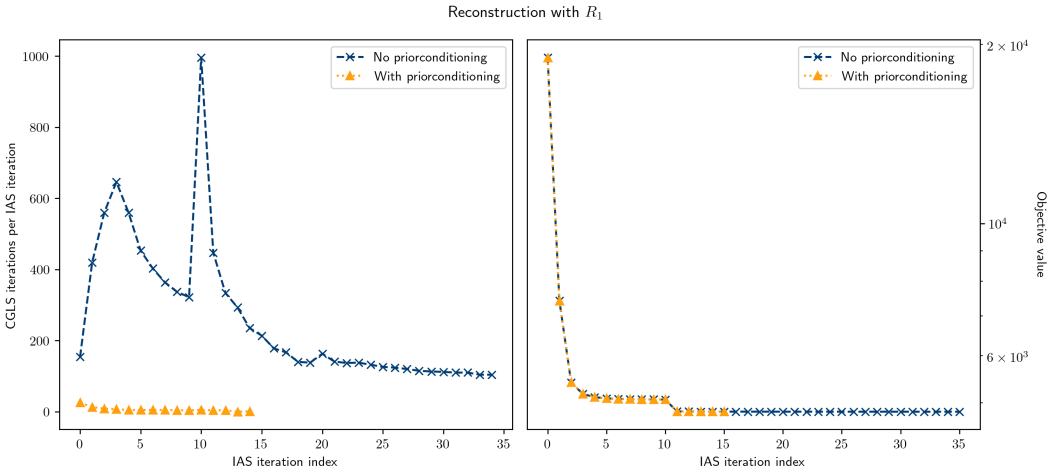


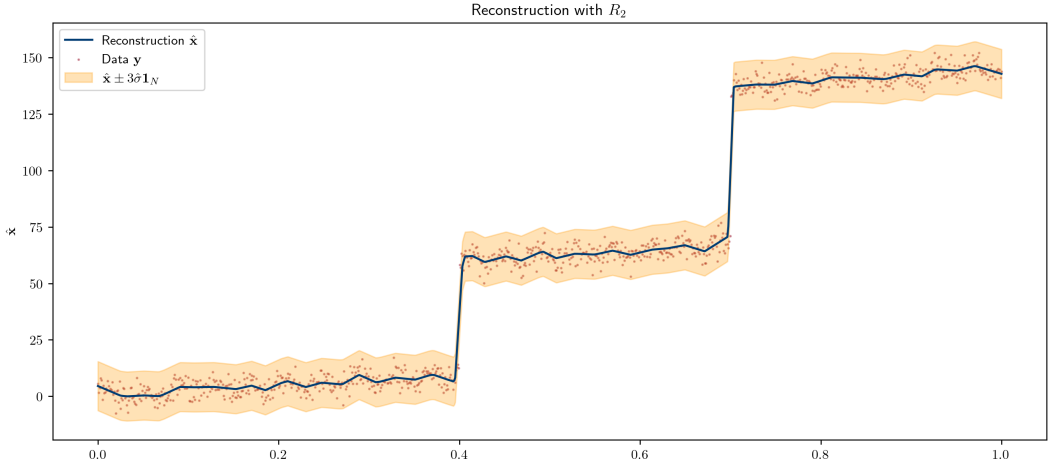
$$\boldsymbol{R}_1 = \begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \end{bmatrix} \quad \boldsymbol{R}_2 = \begin{bmatrix} -1 & 2 & -1 & \\ & \ddots & \ddots & \ddots \\ & & -1 & 2 & 1 \end{bmatrix} \quad \boldsymbol{R}_3 = \begin{bmatrix} -1 & 3 & -3 & 1 & \\ & \ddots & \ddots & \ddots & \ddots \\ & & -1 & 3 & -3 & 1 \end{bmatrix}$$

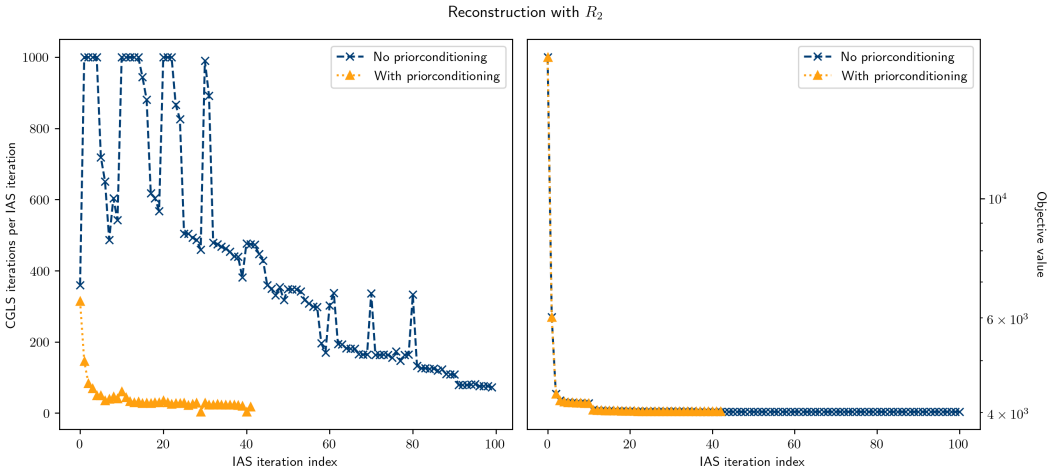
$$\boldsymbol{W}_1 = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix} \quad \boldsymbol{W}_2 = \begin{bmatrix} 1 & 1 \\ 1 & 2 \\ \vdots & \vdots \\ 1 & n-1 \\ 1 & n \end{bmatrix} \quad \boldsymbol{W}_3 = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 3 \\ 1 & 3 & 6 \\ \vdots & \vdots & \vdots \\ 1 & n-2 & \\ 1 & n-1 & n(n-1)/2 \\ 1 & n & n(n+1)/2 \end{bmatrix}$$

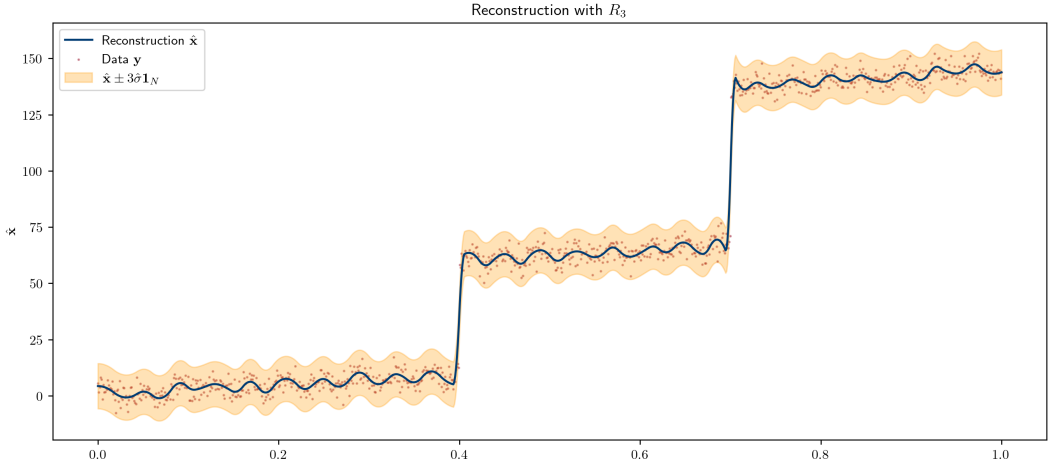

Ground truth $\overline{\mathbf{x}}$
Data $\mathbf{y}$
150
125
100
75
50
25
0
$f(x)$
0.0
0.2
0.4
0.6
0.8
1.0
x

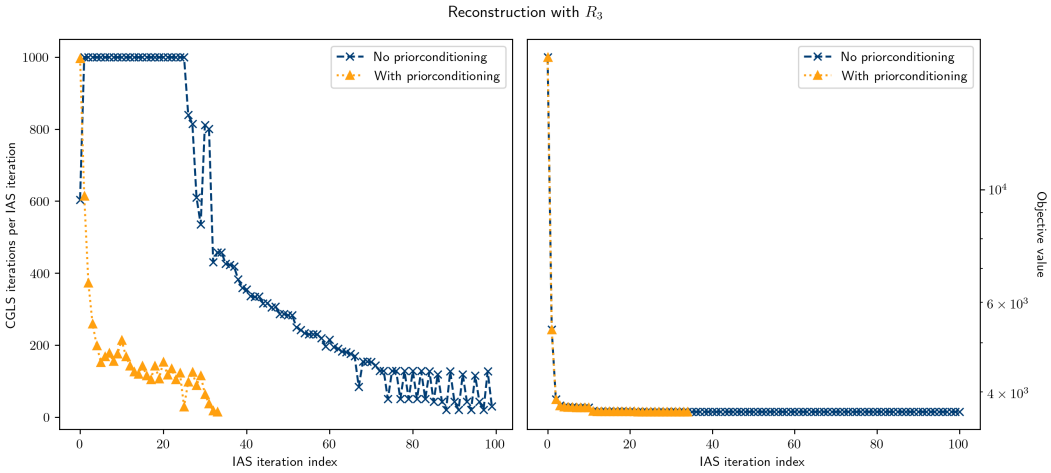

Reconstruction with R_1
Reconstruction $\hat{\mathbf{x}}$
Data $\mathbf{y}$
$\hat{\mathbf{x}} \pm 3\hat{\sigma}\mathbf{1}_N$
$\hat{\mathbf{x}}$
160
140
120
100
80
60
40
20
0
0.0
0.2
0.4
0.6
0.8
1.0


Reconstruction with R_1
No priorconditioning
With priorconditioning
CGLS iterations per IAS iteration
IAS iteration index
No priorconditioning
With priorconditioning
Objective value
IAS iteration index


Reconstruction with R_2
Reconstruction $\hat{\mathbf{x}}$
Data $\mathbf{y}$
$\hat{\mathbf{x}} \pm 3\hat{\sigma}\mathbf{1}_N$
$\hat{\mathbf{x}}$
150
125
100
75
50
25
0
0.0
0.2
0.4
0.6
0.8
1.0


Reconstruction with R_2
No priorconditioning
With priorconditioning
CGLS iterations per IAS iteration
IAS iteration index
No priorconditioning
With priorconditioning
Objective value
IAS iteration index


Reconstruction with R_3
Reconstruction $\hat{\mathbf{x}}$
Data $\mathbf{y}$
$\hat{\mathbf{x}} \pm 3\hat{\sigma}\mathbf{1}_N$
$\hat{\mathbf{x}}$
150
125
100
75
50
25
0
0.0
0.2
0.4
0.6
0.8
1.0


Reconstruction with R_3
No priorconditioning
With priorconditioning
CGLS iterations per IAS iteration
IAS iteration index
No priorconditioning
With priorconditioning
Objective value
IAS iteration index

However, the factorization approach is infeasible for large problems (even if sparse).

An alternative approach is to use an iterative method for computing pseudoinverses.

Idea: the matrix-vector product $\boldsymbol{z} = \boldsymbol{R}^\dagger \boldsymbol{v}$ can be computed approximately by applying the CG method to the solution of

$$\boldsymbol{R}^T \boldsymbol{R} \boldsymbol{z} = \boldsymbol{R}^T \boldsymbol{v}$$

with any initialization $\boldsymbol{z}^{(0)} \in \mathrm{col}(\boldsymbol{R}^T)$, e.g., $\boldsymbol{z}^{(0)} = \boldsymbol{0}_n$.

M. R. Hestenes, Pseudoinversus and conjugate gradients, Communications of the ACM, 1975.

However, the factorization approach is infeasible for large problems (even if sparse).

An alternative approach is to use an iterative method for computing pseudoinverses.

Idea: the matrix-vector product $\boldsymbol{z} = \boldsymbol{R}^{\dagger}\boldsymbol{v}$ can be computed approximately by applying the CG method to the solution of

$$\boldsymbol{R}^T\boldsymbol{R}\boldsymbol{z} = \boldsymbol{R}^T\boldsymbol{v}$$

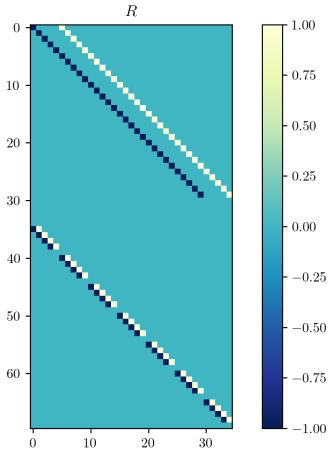
with any initialization $\boldsymbol{z}^{(0)} \in \mathrm{col}(\boldsymbol{R}^T)$, e.g., $\boldsymbol{z}^{(0)} = \boldsymbol{0}_n$.

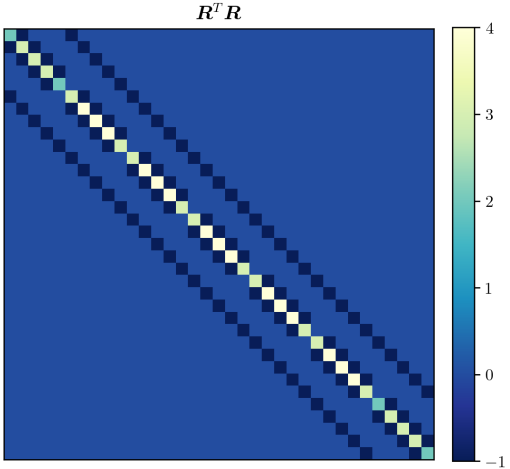
M. R. Hestenes, Pseudoinversus and conjugate gradients, Communications of the ACM, 1975.

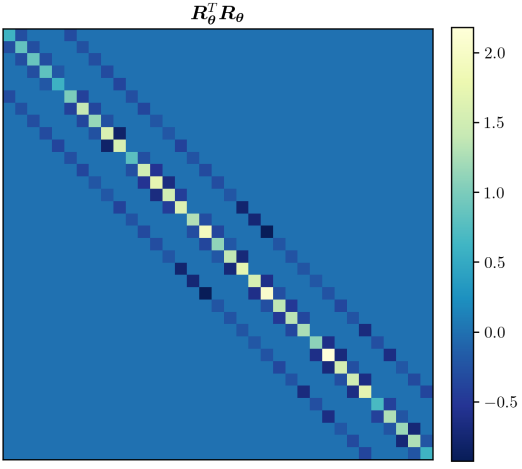
Case study: $\boldsymbol{R}$ = discrete gradient with Neumann boundary

$$\boldsymbol{R}_l = \begin{bmatrix} -1 & 1 & & \\ & \ddots & \ddots & \\ & & -1 & 1 \\ & & & 0 \end{bmatrix} \qquad \boldsymbol{R} = \begin{bmatrix} \boldsymbol{R}_{n_1} \otimes \boldsymbol{I}_{n_2} \\ \boldsymbol{I}_{n_1} \otimes \boldsymbol{R}_{n_2} \end{bmatrix}$$

$$\mathbf{1}_{n_1 n_2} \in \ker(\boldsymbol{R})$$







$$\boldsymbol{R}^T\boldsymbol{R} = \boldsymbol{B}_{\mathrm{DCT}}^T\boldsymbol{\Lambda}\boldsymbol{B}_{\mathrm{DCT}}$$

$$\boldsymbol{R}_{\boldsymbol{\theta}}^T\boldsymbol{R}_{\boldsymbol{\theta}} = ???$$

Preconditioning

Can we use the unweighted Laplacian as a preconditioner for the CG method computing $\boldsymbol{z} = \boldsymbol{R}_{\boldsymbol{\theta}}^{\dagger}\boldsymbol{v}$?

Lemma B.1. *Let $A, M \in \mathbb{R}^{N\times N}$ be symmetric matrices such that $A, M \succeq 0$ (the matrices are positive semi-definite) and $\mathrm{col}(M) = \mathrm{col}(A)$, and let $\mathbf{b} \in \mathrm{col}(A)$. Let L be any matrix such that $M = LL^T$, such as (but not limited to) the spectral square root. Then the problem*

$$\text{find } \mathbf{x} \in \mathrm{col}(A) \text{ s.t. } A\mathbf{x} = \mathbf{b} \tag{B.3}$$

has a unique solution $\mathbf{x}^\star$, which is also given by $\mathbf{x}^\star = (L^T)^\dagger \mathbf{z}^\star$ where $\mathbf{z}^\star$ is the unique solution to the problem

$$\text{find } \mathbf{z} \in \mathrm{col}(A) \text{ s.t. } L^\dagger A (L^T)^\dagger \mathbf{z} = L^\dagger \mathbf{b}. \tag{B.4}$$

Preconditioning

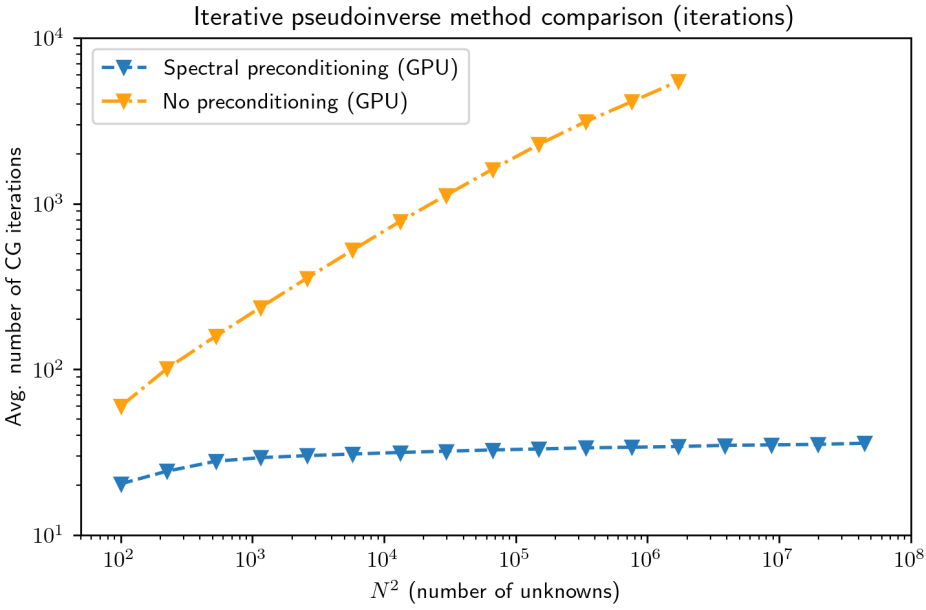
Algorithm B.2 Preconditioned conjugate gradient method for SPD system.

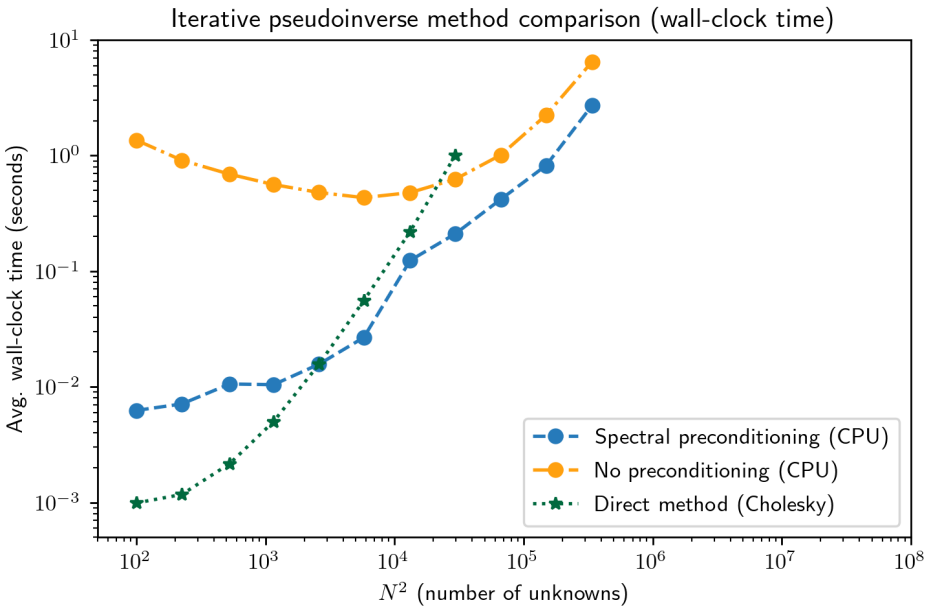
Input: SPD $A \in \mathbb{R}^{N\times N}$, $\mathbf{b} \in \mathbb{R}^N$, $\mathbf{x}^{(0)} \in \mathbb{R}^N$, SPD $M \in \mathbb{R}^{N\times N}$
Output: Solution $\mathbf{x}^\star \approx A^{-1}\mathbf{b}$
$\mathbf{r}^{(0)} = \mathbf{b} - A\mathbf{x}^{(0)}$
$\mathbf{z}^{(0)} = M^{-1}\mathbf{r}^{(0)}$
$\mathbf{d}^{(0)} = \mathbf{z}^{(0)}$
$k = 0$
repeat
  $\alpha^{(k)} = (\mathbf{r}^{(k)})^T\mathbf{z}^{(k)} / (\mathbf{d}^{(k)})^T A\mathbf{d}^{(k)}$
  $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha^{(k)}\mathbf{d}^{(k)}$
  $\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - \alpha^{(k)}A\mathbf{d}^{(k)}$
  $\mathbf{z}^{(k+1)} = M^{-1}\mathbf{r}^{(k+1)}$
  $\beta^{(k+1)} = (\mathbf{r}^{(k+1)})^T\mathbf{z}^{(k+1)} / (\mathbf{r}^{(k)})^T\mathbf{z}^{(k)}$
  $\mathbf{d}^{(k+1)} = \mathbf{z}^{(k+1)} + \beta^{(k)}\mathbf{d}^{(k)}$
  $k = k + 1$
until convergence criterion satisfied
return $\mathbf{x}^\star$

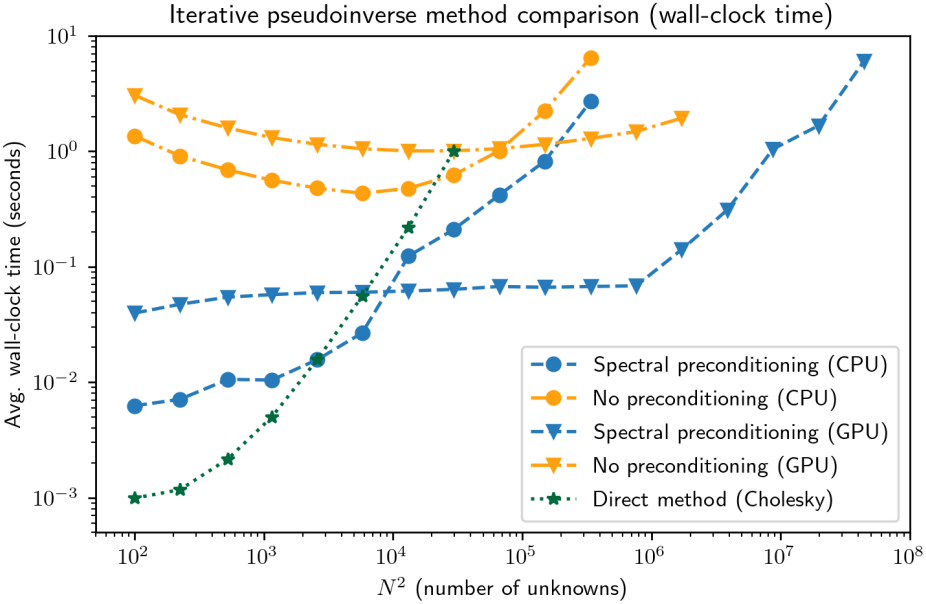
Preconditioning

Algorithm B.1 Preconditioned conjugate gradient method for singular system.

Input: SSPD $A \in \mathbb{R}^{N\times N}$, $\mathbf{b} \in \mathrm{col}(A)$, $\mathbf{x}^{(0)} \in \mathrm{col}(A)$, SSPD $M \in \mathbb{R}^{N\times N}$ s.t. $\mathrm{col}(M) = \mathrm{col}(A)$
Output: Solution $\mathbf{x}^\star \approx A^\dagger \mathbf{b}$
$\mathbf{r}^{(0)} = \mathbf{b} - A\mathbf{x}^{(0)}$
$\mathbf{z}^{(0)} = M^\dagger \mathbf{r}^{(0)}$
$\mathbf{d}^{(0)} = \mathbf{z}^{(0)}$
$k = 0$
repeat
  $\alpha^{(k)} = (\mathbf{r}^{(k)})^T \mathbf{z}^{(k)} / (\mathbf{d}^{(k)})^T A \mathbf{d}^{(k)}$
  $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha^{(k)} \mathbf{d}^{(k)}$
  $\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - \alpha^{(k)} A \mathbf{d}^{(k)}$
  $\mathbf{z}^{(k+1)} = M^\dagger \mathbf{r}^{(k+1)}$
  $\beta^{(k+1)} = (\mathbf{r}^{(k+1)})^T \mathbf{z}^{(k+1)} / (\mathbf{r}^{(k)})^T \mathbf{z}^{(k)}$
  $\mathbf{d}^{(k+1)} = \mathbf{z}^{(k+1)} + \beta^{(k)} \mathbf{d}^{(k)}$
  $k = k + 1$
until convergence criterion satisfied
return $\mathbf{x}^\star$


Iterative pseudoinverse method comparison (iterations)
Spectral preconditioning (GPU)
No preconditioning (GPU)
Avg. number of CG iterations
10^4
10^3
10^2
10^1
10^2
10^3
10^4
10^5
10^6
10^7
10^8
N^2 (number of unknowns)


Iterative pseudoinverse method comparison (wall-clock time)
Avg. wall-clock time (seconds)
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{2}
10^{3}
10^{4}
10^{5}
10^{6}
10^{7}
10^{8}
N^2 (number of unknowns)
Spectral preconditioning (CPU)
No preconditioning (CPU)
Direct method (Cholesky)


Iterative pseudoinverse method comparison (wall-clock time)
Avg. wall-clock time (seconds)
10^{1}
10^{0}
10^{-1}
10^{-2}
10^{-3}
10^{2}
10^{3}
10^{4}
10^{5}
10^{6}
10^{7}
10^{8}
N^2 (number of unknowns)
Spectral preconditioning (CPU)
No preconditioning (CPU)
Spectral preconditioning (GPU)
No preconditioning (GPU)
Direct method (Cholesky)

Generalized IAS Algorithm

Input: Data $\boldsymbol{y}$, forward operator $\boldsymbol{F}$, sparsifying transformation $\boldsymbol{R}$, hyper-parameters (r, β, ϑ) and $(\tilde{r}, \tilde{\beta}, \tilde{\vartheta})$, initialization $\boldsymbol{x}^{(0)}$
Output: Approximate MAP estimate $(\boldsymbol{x}^{\mathrm{MAP}}, \boldsymbol{\theta}^{\mathrm{MAP}}, \nu^{\mathrm{MAP}})$ for the joint posterior $\pi(\boldsymbol{x}, \boldsymbol{\theta}, \nu | \boldsymbol{y})$
repeat
  Update $\boldsymbol{\theta}$ by $\boldsymbol{\theta}^{(k+1)} = \arg\max_{\boldsymbol{\theta}} \pi(\boldsymbol{x}^{(k)}, \boldsymbol{\theta}, \nu^{(k)} \mid \boldsymbol{y}) = \arg\min_{\boldsymbol{\theta}} \mathcal{G}(\boldsymbol{x}^{(k)}, \boldsymbol{\theta}, \nu^{(k)})$
  Update ν by $\nu = \arg\max_{\nu} \pi(\boldsymbol{x}^{(k)}, \boldsymbol{\theta}^{(k+1)}, \nu \mid \boldsymbol{y}) = \arg\min_{\nu} \mathcal{G}(\boldsymbol{x}^{(k)}, \boldsymbol{\theta}^{(k+1)}, \nu)$
  Update $\boldsymbol{x}$ by $\boldsymbol{x}^{(k+1)} = \arg\max_{\boldsymbol{x}} \pi(\boldsymbol{x}, \boldsymbol{\theta}^{(k+1)}, \nu^{(k+1)} \mid \boldsymbol{y}) = \arg\min_{\boldsymbol{x}} \mathcal{G}(\boldsymbol{x}, \boldsymbol{\theta}^{(k-1)}, \nu^{(k+1)})$
until convergence or the maximum number of iterations is reached

Perform update by solving (transformed) least squares problem

$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{R}_{\boldsymbol{\theta}} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_k \end{bmatrix} \quad \boldsymbol{x} = \boldsymbol{R}_{\boldsymbol{\theta}}^{\#} \boldsymbol{w} \quad \begin{bmatrix} \boldsymbol{F}\boldsymbol{R}_{\boldsymbol{\theta}}^{\#} \\ \boldsymbol{I}_k \end{bmatrix} \boldsymbol{w} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{0}_k \end{bmatrix}$$

Least squares problem is solved by CGLS method

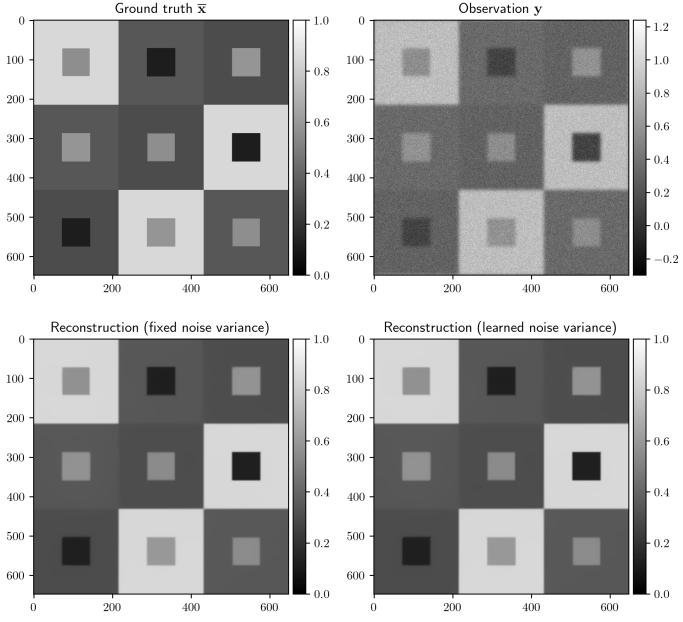
Matvecs computed iteratively using PCG

CGLS Algorithm

Input: Matrix $\boldsymbol{A} \in \mathbb{R}^{m \times n}$ with $\mathrm{rank}(\boldsymbol{A}) = n$, RHS vector $\boldsymbol{b} \in \mathbb{R}^n$, initialization $\boldsymbol{x}^{(0)}$
Output: Approximation $\boldsymbol{x}^*$ to least squares solution of $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$.
$\boldsymbol{d}^{(0)} = \boldsymbol{b} - \boldsymbol{A}\boldsymbol{x}^{(0)}$
$\boldsymbol{r}^{(0)} = \boldsymbol{A}^T\boldsymbol{d}^{(0)}$
$\boldsymbol{p}^{(0)} = \boldsymbol{r}^{(0)}$
$\boldsymbol{y}^{(0)} = \boldsymbol{A}\boldsymbol{p}^{(0)}$
$k = 1$
repeat
  $\alpha^{(k)} = \|\boldsymbol{r}^{(k-1)}\|_2^2 / \|\boldsymbol{y}^{(k-1)}\|_2^2$
  $\boldsymbol{x}^{(k)} = \boldsymbol{x}^{(k-1)} + \alpha^{(k)}\boldsymbol{p}^{(k-1)}$
  $\boldsymbol{d}^{(k)} = \boldsymbol{d}^{(k-1)} - \alpha^{(k)}\boldsymbol{y}^{(k-1)}$
  $\boldsymbol{r}^{(k)} = \boldsymbol{A}^T\boldsymbol{d}^{(k)}$
  $\beta^{(k)} = \|\boldsymbol{r}^{(k)}\|_2^2 / \|\boldsymbol{r}^{(k-1)}\|_2^2$
  $\boldsymbol{p}^{(k)} = \boldsymbol{r}^{(k)} + \beta^{(k)}\boldsymbol{p}^{(k-1)}$
  $\boldsymbol{y}^{(k)} = \boldsymbol{A}\boldsymbol{p}^{(k)}$
  $k = k + 1$
until convergence or the maximum number of iterations is reached
return $\boldsymbol{x}^*$

Algorithm B.1 Preconditioned conjugate gradient method for singular system.

Input: SSPD $A \in \mathbb{R}^{N \times N}$, $\mathbf{b} \in \mathrm{col}(A)$, $\mathbf{x}^{(0)} \in \mathrm{col}(A)$, SSPD $M \in \mathbb{R}^{N \times N}$ s.t. $\mathrm{col}(M) = \mathrm{col}(A)$
Output: Solution $\mathbf{x}^* \approx A^{\dagger}\mathbf{b}$
$\mathbf{r}^{(0)} = \mathbf{b} - A\mathbf{x}^{(0)}$
$\mathbf{z}^{(0)} = M^{\dagger}\mathbf{r}^{(0)}$
$\mathbf{d}^{(0)} = \mathbf{z}^{(0)}$
$k = 0$
repeat
  $\alpha^{(k)} = (\mathbf{r}^{(k)})^T\mathbf{z}^{(k)} / (\mathbf{d}^{(k)})^T A\mathbf{d}^{(k)}$
  $\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha^{(k)}\mathbf{d}^{(k)}$
  $\mathbf{r}^{(k+1)} = \mathbf{r}^{(k)} - \alpha^{(k)} A\mathbf{d}^{(k)}$
  $\mathbf{z}^{(k+1)} = M^{\dagger}\mathbf{r}^{(k+1)}$
  $\beta^{(k+1)} = (\mathbf{r}^{(k+1)})^T\mathbf{z}^{(k+1)} / (\mathbf{r}^{(k)})^T\mathbf{z}^{(k)}$
  $\mathbf{d}^{(k+1)} = \mathbf{z}^{(k+1)} + \beta^{(k)}\mathbf{d}^{(k)}$
  $k = k + 1$
until convergence criterion satisfied
return $\mathbf{x}^*$


Ground truth $\overline{\mathbf{x}}$
Observation $\mathbf{y}$
Reconstruction (fixed noise variance)
Reconstruction (learned noise variance)

APPLICATION TO UQ

Posterior sampling

- Challenging since high-dimensional, strong correlations, and multimodal
- Gibbs approaches (with or without de-correlating transformation such as non-centering) rely on efficient sampling of the Gaussian conditional $\boldsymbol{x} \,|\, \boldsymbol{\theta}$
- Can we apply priorconditioning to make sampling the Gaussian conditional $\boldsymbol{x} \,|\, \boldsymbol{\theta}$ as fast as possible?

Perturbation optimization

Idea: drawing a sample from the conditional $\boldsymbol{x} \mid \boldsymbol{\theta}$ can be cast as solving a randomized linear system / least squares problem / quadratic optimization problem.

$$\left(\boldsymbol{F}^T\boldsymbol{F} + \boldsymbol{R}_{\boldsymbol{\theta}}^T\boldsymbol{R}_{\boldsymbol{\theta}}\right)\hat{\boldsymbol{x}} = \boldsymbol{F}^T(\boldsymbol{y} + \mathcal{N}(\boldsymbol{0}_m, \boldsymbol{I}_m)) + \boldsymbol{R}_{\boldsymbol{\theta}}^T\mathcal{N}(\boldsymbol{0}_k, \boldsymbol{I}_k)$$

$$\begin{bmatrix}\boldsymbol{F} \\ \boldsymbol{R}_{\boldsymbol{\theta}}\end{bmatrix}\hat{\boldsymbol{x}} = \begin{bmatrix}\boldsymbol{y} + \mathcal{N}(\boldsymbol{0}_m, \boldsymbol{I}_m) \\ \mathcal{N}(\boldsymbol{0}_k, \boldsymbol{I}_k)\end{bmatrix}$$

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}\in\mathbb{R}^n}{\arg\min} \left\|\boldsymbol{F}\boldsymbol{x} - \boldsymbol{y} - \mathcal{N}(\boldsymbol{0}_m, \boldsymbol{I}_m)\right\|_2^2 + \left\|\boldsymbol{R}_{\boldsymbol{\theta}}\boldsymbol{x} - \mathcal{N}(\boldsymbol{0}_k, \boldsymbol{I}_k)\right\|_2^2$$

Lemma (standard form for sampling)

The least squares solution $\boldsymbol{x}^\star$ of the system

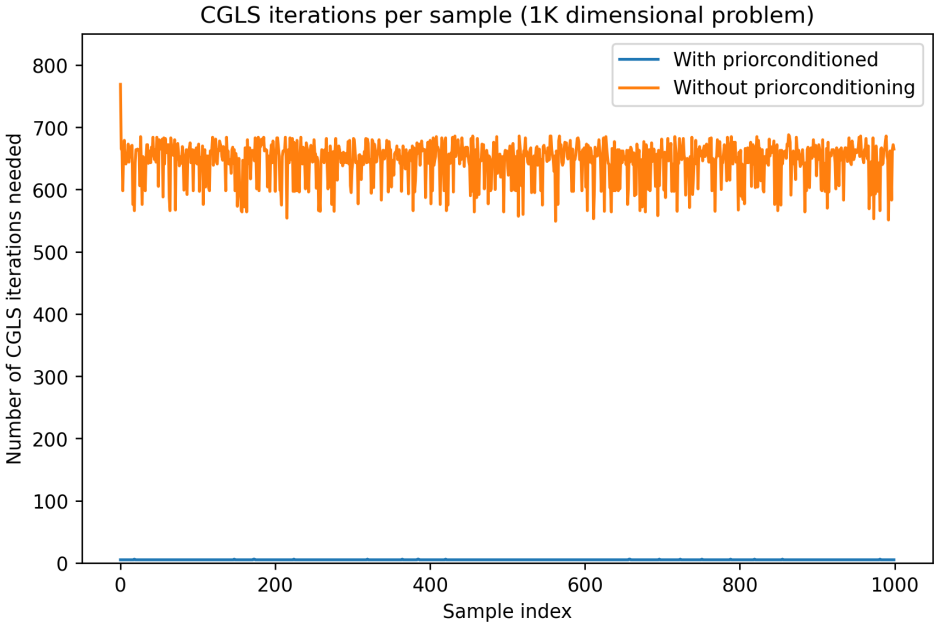
$$\begin{bmatrix} \boldsymbol{F} \\ \boldsymbol{R} \end{bmatrix} \boldsymbol{x} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{b}_2 \end{bmatrix}$$

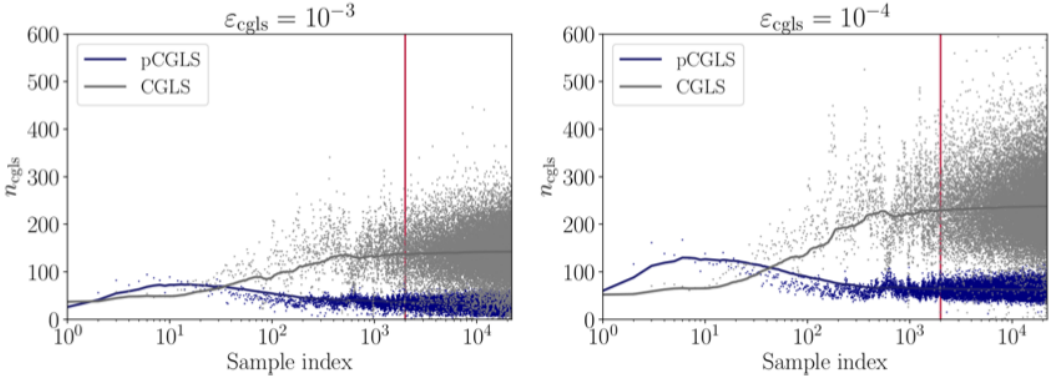
can be expressed as $\boldsymbol{x} = \boldsymbol{x}_{\ker} + \boldsymbol{x}_\perp$, where

$$\boldsymbol{x}_{\ker} = \boldsymbol{W}(\boldsymbol{F}\boldsymbol{W})^\dagger \boldsymbol{b}_1, \quad \boldsymbol{x}_\perp = \boldsymbol{R}^\# \boldsymbol{z},$$

and $\boldsymbol{z}^\star$ is the least squares solution of

$$\begin{bmatrix} \boldsymbol{F}\boldsymbol{R}^\# \\ \boldsymbol{I}_k \end{bmatrix} \boldsymbol{z} = \begin{bmatrix} \boldsymbol{b}_1 \\ \boldsymbol{R}\boldsymbol{R}^\dagger \boldsymbol{b}_2 \end{bmatrix}.$$

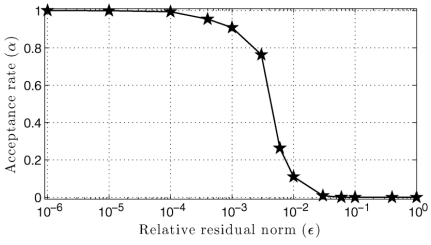

CGLS iterations per sample (1K dimensional problem)
With priorconditioned
Without priorconditioning
Number of CGLS iterations needed
Sample index
0
100
200
300
400
500
600
700
800
0
200
400
600
800
1000



F. Uribe, Y. Dong, and P. C. Hansen, Horseshoe priors for edge-preserving linear Bayesian inversion, SIAM Journal on Scientific Computing, 2023.

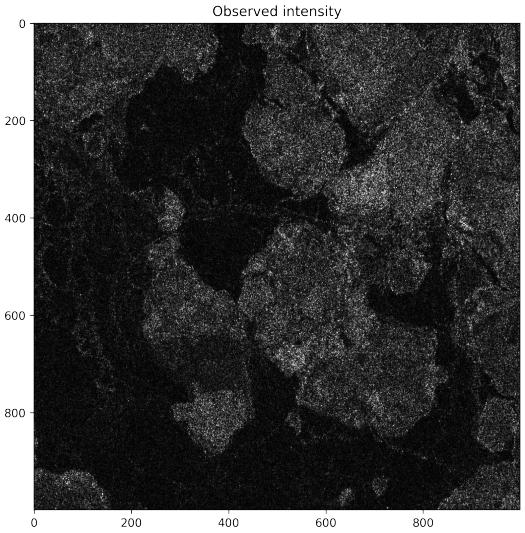
Reversible jump perturbation optimization (RJPO)

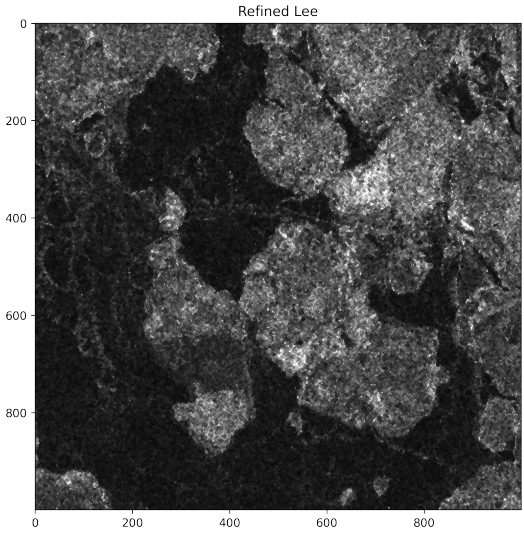
- Solution via direct method can be very costly (Cholesky factorizations)
- Approximate solution by an iterative method may be cheaper, but will not preserve the stationary distribution
- It is possible to equip these methods with a simple accept-reject step based on the residual norm (CG sampler) to correct this bias



C. Gilavert, S. Moussaoui, and J. Idier, Efficient Gaussian sampling for solving large-scale inverse problems using MCMC

OTHER DIRECTIONS


Observed intensity
0
200
400
600
800
0
200
400
600
800


Refined Lee
0
200
400
600
800
0
200
400
600
800

Hierarchical model for multiplicative denoising

$$
\begin{aligned}
[\boldsymbol{f}]_i \,|\, \boldsymbol{u} &\overset{\text{ind}}{\sim} \text{Gamma}\left(L, \frac{L}{[\boldsymbol{A}\boldsymbol{u}]_i}\right), \quad i = 1, \ldots, m, \\
\boldsymbol{u} \,|\, \boldsymbol{\theta} &\sim \mathcal{N}\left(\boldsymbol{0}, \left(\boldsymbol{R}^T \boldsymbol{D}_{\boldsymbol{\theta}}^{-1} \boldsymbol{R}\right)^{-1}\right), \\
[\boldsymbol{\theta}]_i &\overset{\text{iid}}{\sim} \mathcal{GG}\left(r, \beta, \vartheta\right), \quad i = 1, \ldots, k.
\end{aligned}
$$

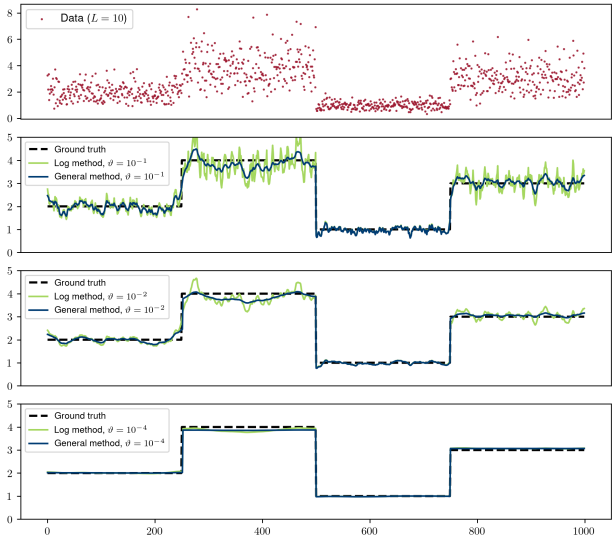
$$
\mathcal{G}(\boldsymbol{u}, \boldsymbol{\theta}) = \sum_{i=1}^{m} L \log\left([\boldsymbol{A}\boldsymbol{u}]_i\right) + L\frac{[\boldsymbol{f}]_i}{[\boldsymbol{A}\boldsymbol{u}]_i} + \frac{1}{2}\|\boldsymbol{D}_{\boldsymbol{\theta}}^{-1/2}\boldsymbol{R}\boldsymbol{u}\|_2^2 - \log \pi(\boldsymbol{\theta}),
$$

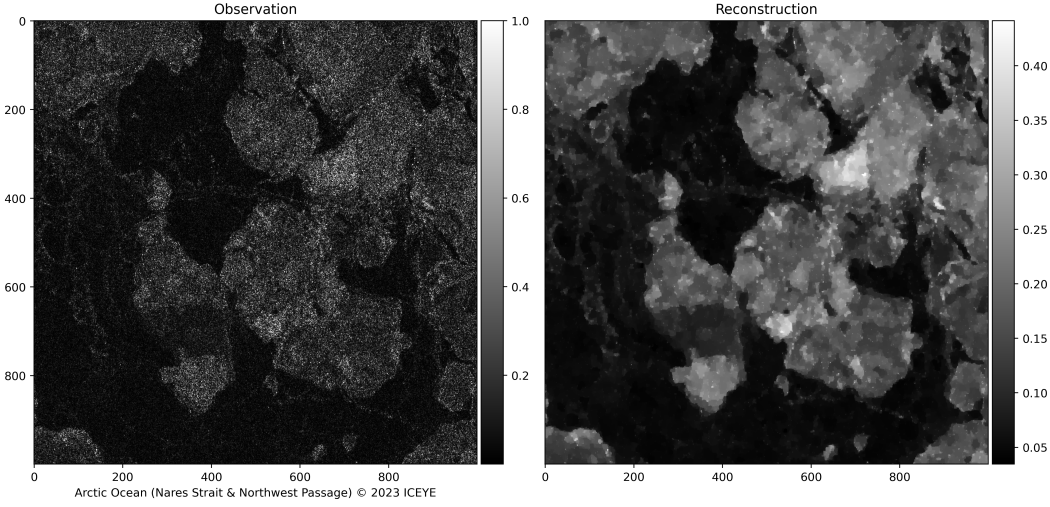
Changes to coordinate descent from Gaussian case?

- The method for performing the $\boldsymbol{\theta}$-update is essentially unchanged.
- The $\boldsymbol{u}$-update corresponds to finding

$$\underset{\boldsymbol{u}\in\mathbb{R}^n_{++}}{\arg\min} \quad \sum_{i=1}^{m} L\log\left([\boldsymbol{A}\boldsymbol{u}]_i\right) + L\frac{[\boldsymbol{f}]_i}{[\boldsymbol{A}\boldsymbol{u}]_i} + \frac{1}{2}\|\boldsymbol{D}_{\boldsymbol{\theta}}^{-1/2}\boldsymbol{R}\boldsymbol{u}\|_2^2$$

- We can apply Newton's method to this subproblem
- Newton's method can be expressed as solving a sequence of least squares problems, to which priorconditioning can be applied!


Data ($L = 10$)
Ground truth
Log method, $\vartheta = 10^{-1}$
General method, $\vartheta = 10^{-1}$
Ground truth
Log method, $\vartheta = 10^{-2}$
General method, $\vartheta = 10^{-2}$
Ground truth
Log method, $\vartheta = 10^{-4}$
General method, $\vartheta = 10^{-4}$
0
200
400
600
800
1000


Observation
Reconstruction
0
200
400
600
800
1.0
0.8
0.6
0.4
0.2
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
Arctic Ocean (Nares Strait & Northwest Passage) © 2023 ICEYE

Questions?

jonathan.t.lindbloom.gr@dartmouth.edu