Optimal Experimental Design for Constrained Inverse Problems

Transition Workshop @ SAMSI

May 1, 2017

Lars Ruthotto\textsuperscript{1}, Julianne Chung\textsuperscript{2}, Matthias Chung\textsuperscript{2}

\textsuperscript{1}Department of Mathematics and Computer Science, Emory University
\textsuperscript{2}Department of Mathematics, Virginia Tech

lruthotto@emory.edu
Agenda

- **Motivation and Introduction**
  - Example: 2D Tomography
  - Two models for sparsity enforcing design

- **Empirical Bayes Formulation**
  - idea: improve design using training data
  - bi-level optimization problem
  - inequality constraints on both levels.
  - parallel framework to accelerate computation

- **Numerical Examples**

- **Summary / Open Problems**
Motivation and Introduction
Application: 2D Tomography

Discrete Radon transform: Given $f_{\text{true}} \in \mathbb{R}^n$ obtain data for angle $\theta$ as

$$d(\theta) = \text{TR}(\theta)f_{\text{true}},$$

- $\text{R}(\theta) \in \mathbb{R}^{n \times n}$ - rotation matrix
- $T \in \mathbb{R}^{nT \times n}$ - tomography operator

OED Question: How to optimally choose $\theta_1, \theta_2, \ldots, \theta_\ell$?
Inverse Problem

Consider the discrete inverse problem

\[ d(p) = M(p)f_{\text{true}} + \epsilon(p) \]

with

- \( d(p) \in \mathbb{R}^m \) - noisy data
- \( M(p) \in \mathbb{R}^{m \times n} \) - linear forward operator
- \( p \in \mathbb{R}^\ell \) - design parameters (angles, offsets, ...)
- \( f_{\text{true}} \in \mathbb{R}^n \) - true model
- \( \epsilon(p) \in \mathbb{R}^m \) - measurement noise (\( \epsilon \in \mathcal{N}(0, \Gamma_\epsilon(p)) \))

Assume that true model(s) satisfy

- \( C_e f - c_e = 0 \) (e.g., sum of intensities known)
- \( C_i f - c_i \geq 0 \) (e.g., non-negativity, box constraints, ...)

Does optimal design differ in constrained/unconstrained problems?
Optimal Experimental Design: Problem A

Idea: Start with *many* angles $\theta_1, \theta_2, \ldots, \theta_\ell$ ($\ell \gg 0$) and consider

$$M(p) = \text{diag}(p)A, \quad A = \begin{bmatrix} \text{TR}(\theta_1) \\ \vdots \\ \text{TR}(\theta_\ell) \end{bmatrix}, \quad \text{and} \quad d(p) = \text{diag}(p)d.$$ (\(p \in \mathbb{R}^\ell \) with \(p \geq 0\) encodes importance of measurements)

Find relevant components in \(p\) by solving the OED problem (upper-level)

$$\min_{p \geq 0} \mathbb{E} \left\| \hat{f}(p) - f_{\text{true}} \right\|_2^2 + \beta \|p\|_1.$$ Here, \(\hat{f}(p)\) solves the inverse (lower-level) problem, i.e.,

$$\hat{f}(p) = \arg\min_f \frac{1}{2} \left\| M(p)f - d(p) \right\|_2^2 + \frac{\alpha^2}{2} \left\| L(f - \mu) \right\|_2^2$$ subject to $f_L \leq f \leq f_H$, $C_e f - c_e = 0$.

Inequality constraints $\Rightarrow$ no closed-form solution for \(\hat{f}(p)\).
Optimal Experimental Design: Problem B

Let \( p \in \mathbb{R}^\ell \) (\( \ell \) small) be the projection angles and

\[
M(p) = \begin{bmatrix}
TR(p_1) \\
\vdots \\
TR(p_\ell)
\end{bmatrix}, \quad \text{and} \quad d(p) = M(p)f_{\text{true}} + \varepsilon.
\]

Find optimal angled by solving the OED problem \((upper-level)\)

\[
\min_{p_{\max} \geq p \geq p_{\min}} \mathbb{E} \left\| \hat{f}(p) - f_{\text{true}} \right\|_2^2.
\]

Here, \( \hat{f}(p) \) solves the inverse \((lower-level)\) problem, i.e.,

\[
\hat{f}(p) = \arg\min_f \frac{1}{2} \left\| M(p)f - d(p) \right\|_2^2 + \frac{\alpha^2}{2} \left\| L(f - \mu) \right\|_2^2
\]

subject to \( f_L \leq f \leq f_H, \quad C_e f - c_e = 0 \).
Related work


### OED for Inverse Problems with Constraints

- will optimal design change when considering constraints?
- can we solve large-scale OED problems?
  - many design parameters
  - high-resolution inversions
  - lots of test data
Empirical Bayes Formulation
Emprirical Bayes Formulation

Let $f_{true}^{(1)}, \ldots, f_{true}^{(N)} \in \mathbb{R}^n$ be true models (training data). Given $p$ simulate datasets $d^{(1)}(p), \ldots, d^{(N)}(p) \in \mathbb{R}^m$ and obtain reconstructions $\hat{f}^{(1)}(p), \ldots, \hat{f}^{(1)}(p)$ by solving lower-level problem.

Goal: Find $p$ that minimizes Mean Squared Error (MSE)

$$MSE(p) = \frac{1}{2N} \sum_{i=1}^{N} \left\| \hat{f}^{(i)}(p) - f_{true}^{(i)} \right\|^2.$$

Solution strategies:

- Reduced approach: Eliminate lower-level problem and treat $\hat{f}(p)$ as (differentiable?) function
- Upper-level problem: Use projected steepest descent, projected Gauss Newton, ...

Note that

$$\nabla_p MSE(p) = \frac{1}{N} \sum_{i=1}^{N} J_{\hat{f}_i}(p)^\top \left( \hat{f}_i(p) - f_{true}^{(i)} \right).$$

Challenge: Compute Jacobian (or sensitivity) $J_{\hat{f}_i}(p)^\top = \nabla_p \hat{f}_i(p)$
Lower Level Problem as Convex QP

Recall, lower-level problem

\[
\hat{f}(p) = \arg \min_{f} \frac{1}{2} \|M(p)f - d(p)\|^2_2 + \frac{\alpha^2}{2} \|L(f - \mu)\|^2_2
\]

subject to \(f_L \leq f \leq f_H, \quad C_e f - c_e = 0.\)

Introducing slack \(s\), we rewrite this as

\[
\min_{f,s} \frac{1}{2} f^\top Q(p)f + b(p)^\top f
\]

subject to \(C_e f - c_e = 0, \quad C_i f - c_i - s = 0, \quad s \geq 0,\)

where

- \(Q(p) = M(p)^\top M(p) + \alpha L^\top L\)
- \(b(p) = -M(p)^\top d(p) - \alpha L^\top \mu\)
- \(C_i = [I_n; -I_n], \quad c_i = [f_L; -f_H] \) (bound constraints)
Lower Level Problem: Optimality Conditions

The Lagrangian of lower level problem is

\[ \mathcal{L}(\mathbf{f}, \lambda_e, \mathbf{s}, \lambda_i) = \frac{1}{2} \mathbf{f}^\top Q(p) \mathbf{f} + \mathbf{b}(p)^\top \mathbf{f} - \lambda_e^\top (C_e \mathbf{f} - \mathbf{c}_e) - \lambda_i^\top (C_i \mathbf{f} - \mathbf{c}_i - \mathbf{s}). \]

Necessary and sufficient conditions: Find \((\mathbf{f}, \lambda_e, \lambda_i, \mathbf{s})\) such that

\[ F(\mathbf{f}, \lambda_e, \lambda_i, \mathbf{s}) = \begin{pmatrix} \nabla_{\mathbf{f}} \mathcal{L} \\ \nabla_{\lambda_e} \mathcal{L} \\ \nabla_{\lambda_i} \mathcal{L} \end{pmatrix} = 0, \quad \lambda_i \odot \mathbf{s} = 0, \quad \lambda_i, \mathbf{s} \geq 0. \]

Interior point method: Relax the complementarity using \(\sigma > 0\) and \(\mu^k = (\mathbf{s}^k, \lambda_i^k)/m\)

\[ F(\mathbf{f}, \lambda_e, \lambda_i, \mathbf{s}; \sigma, \mu) = \begin{pmatrix} Q(p)\mathbf{f} + \mathbf{b}(p) - C_e^\top \lambda_e - C_i^\top \lambda_i \\ C_e \mathbf{f} - \mathbf{c}_e \\ C_i \mathbf{f} - \mathbf{c}_i - \mathbf{s} \\ S \Lambda_i \mathbf{e} - \sigma \mu^k \mathbf{e} \end{pmatrix} = 0, \quad \lambda_i, \mathbf{s} > 0, \]

with \(S = \text{diag}(\mathbf{s})\) and \(\Lambda_i = \text{diag}(\lambda_i)\). Use Mehrotra’s algorithm.
Sensitivities of Interior Point Solution

Find Jacobian $\hat{J}_f(p) \in \mathbb{R}^{n \times \ell}$ such that for small $\Delta p \in \mathbb{R}^\ell$

$$\hat{f}(p + \Delta p) = \hat{f}(p) + \hat{J}_f(p) \Delta p + O(\|\Delta p\|^2)$$

Differentiating $F$ around $(\hat{f}(p), \lambda_e(p), \lambda_i(p), s(p))$ gives

$$0 = \nabla_p F(f, \lambda_e, \lambda_i, \sigma, \mu) + \begin{pmatrix} \vdots & \vdots & \vdots & \vdots \\ \nabla_f F & \nabla_{\lambda_e} F & \nabla_s F & \nabla_{\lambda_i} F \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix} \begin{pmatrix} J_{\hat{f}}(p) \\ J_{\lambda_e}(p) \\ J_s(p) \\ J_{\lambda_i}(p) \end{pmatrix}$$

This gives in our case

$$\begin{pmatrix} J_{\hat{f}}(p) \\ J_{\lambda_e}(p) \\ J_s(p) \\ J_{\lambda_i}(p) \end{pmatrix} = - \begin{pmatrix} Q(p) & -C_e & 0 & -C_i \\ C_e & 0 & 0 & 0 \\ C_i & 0 & -I & 0 \\ 0 & 0 & \Lambda_i & S \end{pmatrix}^{-1} \begin{pmatrix} \nabla_p (Q(p) \hat{f}(p) + b(p)) \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Building $J_{\hat{f}} \in \mathbb{R}^{m \times \ell}$ costly when $\ell$ large $\rightarrow$ matrix-free mode / store factorization?
Approach to sensitivity computation is very flexible. Need to compute

$$\nabla_p(Q(p)f + b(p)) = \nabla_p(M(p)^\top M(p)f - M(p)^\top d(p))$$

This requires product rule and methods for computing

$$\nabla_p(M(p)f), \quad \nabla_p(M(p)^\top d), \quad \text{and} \quad \nabla_p d(p).$$

Example: For OED problem A we have

- $$\nabla_p(M(p)f) = \text{diag}(Af)$$
- $$\nabla_p(M(p)^\top d) = (\text{diag}(d)A)^\top$$
- $$\nabla_p(d(p)) = \text{diag}(d)$$
Solving OED Problem A

Two step procedure from Haber, Horesch 2008 but with projected Gauss-Newton-CG.

Step 1: Consider the relaxed $\ell_0$-problem for some $\beta > 0$

$$p^* = \arg \min_{p \geq 0} \mathbb{E} \left\| \hat{f}(p) - f_{\text{true}} \right\|^2 + \beta \|p\|_1 .$$

Step 2: Let $p^* = Q\bar{p}$ where $Q \in \mathbb{R}^{\ell \times k}$ for $k \ll \ell$. Fine tune the non-zero weights by solving the small-scale problem.

$$\min_{\bar{p} \geq 0} \mathbb{E} \left\| \hat{f}(Q\bar{p}) - f_{\text{true}} \right\|^2$$
Computational Challenges

Recall MSE

\[ \text{MSE}(p) = \frac{1}{2N} \sum_{i=1}^{N} \left\| \hat{f}^{(i)}(p) - f^{(i)}_{\text{true}} \right\|^2, \quad \nabla_p \text{MSE}(p) = \frac{1}{N} \sum_{i=1}^{N} J_f(p)^T \left( \hat{f}_i(p) - f^{(i)}_{\text{true}} \right). \]

Computationally challenging for large scale OED (many parameters, high-resolution recon, many training data).
Structure is similar to parameter estimation problems with many measurements (PDE constrained optimization, machine learning, . . .)

Some ideas

1. Sampling: subsample training data in OED phase.
2. Parallelization: Distribute terms in MSE. IPQP problems solved in parallel, factorizations or preconditioners stored on workers for sensitivity calculations.
Who is julia?

- open source, started 2009, public since 2012
- started by Alan Edelman @ MIT → large developer community
- **dynamic programming language (multiple dispatch)**
- sophisticated type systems (though optional)
- **designed for speed and efficiency (just-in-time compiler)**
- **designed for parallel/cloud computing**
- standard library mostly written in julia
- easy to connect with C, C++, Fortran, Python,"
Selected JuliaInv Packages on Github

**EikonallInv.jl**
A Julia package for solving the inverse eikonal equation on a rectangular mesh.
*Julia*  
1 star  
1 watch  
Updated on Jan 6

**MaxwellFrequency**
*Julia*  
3 stars  
Updated 15 days ago

**DivSigGrad.jl**
Julia Package for Inverse Conductivity Problems
*Julia*  
1 star  
1 watch  
Updated 8 days ago

**FWI.jl**
A Julia package for solving the 2D and 3D acoustic Full-Waveform Inversion on a regular rectangular mesh.
*Julia*  
1 star  
Updated on Jan 15

**jInv.jl**
Flexible Framework for Parallel PDE Parameter Estimation
*Julia*  
8 stars  
9 forks  
Updated 8 days ago

**JTetra**
Tetrahedral mesh support for jInv
*Julia*  
Updated on Jan 23

**Multigrid.jl**
A multigrid package in Julia: smoother and multigrid.
*Julia*  
3 stars  
Updated on Jan 15

**MUMPS.jl**
Forked from Pbelle/MUMPS.jl, MUMPS interface for Julia

**KrylovMethods.jl**
Simple and fast Julia implementation of Krylov subspace methods for linear systems.
*Jupyter Notebook*  
18 stars  
4 forks
Acknowledgements

The Julia community (particularly Jiahao Chen, MIT).

Main jlnv collaborators:

Eldad Haber  
Eran Treister  
Samy Wu

Dave Marchant  
Christoph Schwarzbach  
Patrick Belliveau

LR and SW are supported by the National Science Foundation grant DMS 1522599
Step 1: Prepare forward problems on workers

jInv allows for automatic or user-defined scheduling. Here: Automatic.

1. Send data 1 to worker 1 and data 2 to worker 2
2. Get remote reference from worker 1 and send problem 3
3. Get remote reference from worker 1 and 2
Step 2: Compute Misfit

**Note:** Data locations are fixed.

**Worker 1**
- \( f_1, p\text{For}[1] \)
- \( f_3, p\text{For}[3] \)

\( \hat{p} \)

**Main Process**
- Ref[1]
- Ref[2]
- Ref[3]

\( \hat{f}_1, \hat{V}_1 \)

\( \hat{f}_3, \hat{A}_3 \)

sensitivities, factorizations or preconditioners stored on workers

**Worker 2**
- \( f_2, p\text{For}[2] \)

\( \hat{f}_2, p\text{For}[2] \)
Simple Parallelization using Multiple Dispatch

Example: Linear Least Squares Problem with 10 training models.

**Option 1:** Use single pFor for sequential computation:

1. \[ \text{pFor} = \text{LeastSquaresParam}(A, L, [], D) \]

**Option 2:** Split up sources and use \text{Array\{LeastSquaresParam\}} for parallelization:

1. \[ \text{pFor1} = \text{LeastSquaresParam}(A, L, [], D[:,1:5]) \]
2. \[ \text{pFor2} = \text{LeastSquaresParam}(A, L, [], D[:,6:10]) \]
3. \[ \text{pForp} = [\text{pFor1}; \text{pFor2}] \]

**Option 3:** Distribute least squares problems a priori to reduce communication:

1. \[ \text{pFord} = \text{Array\{RemoteRef\{Channel\{Any\}\}\}}(2) \]
2. \[ \text{pFord}[1] = \text{@spawnat processor1 identity(pFor1)} \]
3. \[ \text{pFord}[2] = \text{@spawnat processor2 identity(pFor2)} \]
Optimization: Projected Gauss-Newton

jInv provides tailored implementation of projected Gauss-Newton for solving

$$\min_{\mathbf{p}} \ f(\mathbf{p}) \ \text{subject to} \ \mathbf{p}_L \leq \mathbf{p} \leq \mathbf{p}_H,$$

for $f : \mathbb{R}^n \to \mathbb{R}$ smooth (in general not convex).
Numerical Experiments
2D Tomography Example: Binary Rectangles

- training data: $40 \times 40 \times 20$
- randomly sized and spaced rectangles
- binary intensities
- 180 projections angles, 40 projections per angle
- $L = I_n$ (identity)
- OED modes:
  1. unconstrained
  2. equality constrained*
  3. non-negativity constrained*
  4. box-constrained*

*: true images used to formulate constraints
Binary Rectangles: OED B Results

MSE vs. angles

- unconstrained
- equality constraints
- non-negativity
- box-constraints

MSE vs. regularization parameter

- unconstrained
- equality
- non-negativity
- box-constrained

\[ \beta \in [10^{-2}, 10^{1}] \]
Binary Rectangle: OED A Results

number of nonzero projections

reconstruction error, MSE

- unconstrained
- equality
- non-negativity
- box-constrained
Binary Rectangle: Reconstruction Results

- **no constraints**
- **equality constraints**
- **non-negativity**
- **box constraints**
2D Tomography Example: Shepp Logan

- training data: $64 \times 64 \times 20$
- intensities between 0 and 1
- 180 projections angles, 64 projections per angle
- $L = I_n$ (identity)
- OED modes:
  1. unconstrained
  2. equality constrained*
  3. non-negativity constrained*
  4. box-constrained*

*: true images used to formulate constraints
Shepp Logan: OED B Results

MSE vs. angles

unconstrained

non-negativity

equality constraints

box-constraints

MSE vs. regularization parameter

unconstrained

equality

non-negativity

box-constrained
Binary Rectangle: Reconstruction Results

- no constraints (nnz=88)
- equality constraints (nnz=74)
- non-negativity (65)
Summary and Outlook
Σ: OED for Constrained Inverse Problems

OED can drastically improve effectiveness of constrained inversions.

Empirical Bayesian Optimal Design
- handles linear equality and inequality constraints
- requirements: training data, resources

Numerical Optimization
- inequalities $\rightarrow$ nonlinear optimality condition
- differentiate solution of interior point method (predictor corrector)
- modular approach: extensible to new designs
- computational challenging $\rightarrow$ parallel processing

Future Work
- Bayesian formulation
- OED for nonlinear constrained inverse problems
- large-scale 3D imaging applications
- application in PDE parameter estimation

Special thanks to y’all at SAMSI and SAMSI sponsors.