Fast "online" migration with Compressive Sensing

Felix J. Herrmann
Fast "online" migration with Compressive Sensing
Felix J. Herrmann, Ning Tu, and Ernie Esser

with help from Mengmeng Wang & Phil

University of British Columbia
Motivation

Push from processing to inversion exposes challenges w.r.t.
- handling IO for larger and larger datasets
- computational resources needed by wave-equation based inversions

Sparsity-promoting inversions:
- produce hifi/high-resolution results
- but require too many computations & passes through the data (IO), and
- are algorithmically complex

Stifles uptake by industry...
Inversion vs processing
– reverse-time migration (RTM)

RTM imaging via **adjoint**, high-pass filtered to remove low-wavenumber RTM artifacts
Inversion vs processing
– sparsity-promoting least-squares migration (SPLSM)


SPLSM image via inversion, # of wave-equation solves roughly equals 1 RTM w/ all data
Contributions

New “online” scheme that provably inverts large-scale problems by
  ‣ working on small randomized subsets of data (e.g. shots) only
  ‣ making the objective strongly convex by thresholding the dual variable

Extremely simple “three liner” implementation that
  ‣ limits # of passes through data & offers flexible parallelism
  ‣ is easily extendible to include e.g. on-the-fly source estimation & multiples

Application areas include:
  ‣ least-squares migration & AVA
Sparsity promotion

Basis Pursuit (BP):

$$\min_{x} \|x\|_1$$
subject to $Ax = b$

- undergirds most sparse recovery problems & compressive sensing (CS)
- designed for underdetermined systems
- needs many iterations
ISTA
– Iterative Shrinkage Thresholding Algorithm

1. for $k = 0, 1, \cdots$
2. $z_{k+1} = x_k - t_k A^* (Ax_k - b_k)$
3. $x_{k+1} = S_\lambda(z_{k+1})$
4. end for

*where $S_\lambda(x) = \text{sign}(x) \cdot \max(|x| - \lambda, 0)$ is soft thresholding and $t_k$ are step lengths

- simple but converges slowly, especially for $\lambda$ small
- BP corresponds to non-trivial limit $\lambda \to 0^+$
- requires (complicated) continuation strategies for $\lambda$

[Daubechies, 03; Figueiredo and Nowak, ’03; Yin et al., ‘08; Beck and Teboulle, ‘09]
Observations

Contributions from “optimizers” yielded robust solvers such as SPGL1
- relatively fast because of continuation methods that relax the constraint
- black boxes with clever state-of-the-art “tricks”

But, their
- convergence is too slow for realistic seismic problems w/ expensive matvecs & IO
- implementation is rather complicated & somewhat inflexible
- design is not optimized for overdetermined problems
SPLSM w/ CS
– slow convergence

SPLSM image via **inversion** w/ **fixed** randomized simultaneous shots and in the presence of modelling errors
Seismic problems are
- often overdetermined
- often “inverted” by applying the (scaled) adjoint (e.g. migration)

\[
A \quad x = b \\
\tilde{x} = A^H b
\]
Least-squares inversion

Consistent & inconsistent overdetermined systems can be solved by

$$\minimize_x \frac{1}{2} \|Ax - b\|_2^2$$

which requires

- multiple matrix-free actions of $$\{A, A^H\}$$
- multiple paths through the data (= many wave-equation solves), and
- does not exploit structure in $$x$$
Example
– noise-free

\[ m = 1000; \] % Number of rows
\[ n = 100; \] % number of columns
\[ nnz = 10; \] % Number of nonzeros

\[ x0 = \text{zeros}(n,1); \]
\[ x0(\text{randperm}(n,nnz)) = \text{randn}(nnz,1); \] % Sparse vector
\[ A = \text{randn}(m,n); \] % Tall system
\[ b = A*x0; \] % data

\[ xcor = A'*b; \] % "Migrate image"
\[ xls = \text{lsqr}(A,b); \] % "LS-migrated image"

\text{lsqr} converged at iteration 12 to a solution with relative residual 9e-07.
Example
– noisy

\( m = 1000; \) \( \% \) Number of rows
\( n = 100; \) \( \% \) number of columns
\( nnz = 10; \) \( \% \) Number of nonzeros

\( x_0 = \text{zeros}(n,1); \)
\( x_0(\text{randperm}(n,nnz)) = \text{randn}(nnz,1); \) \( \% \) Sparse vector
\( A = \text{randn}(m,n); \) \( \% \) Tall system
\( b = A*x_0; \) \( \% \) data
\( b = b + 0.5*\text{std}(b)*\text{randn}(m,1); \) \( \% \) noisy data

\( x_{cor} = A'*b; \) \( \% \) "Migrate image"
\( x_{ls} = \text{lsqr}(A,b); \) \( \% \) "LS-migrated image"

\text{lsqr} \text{ converged at iteration} 12 \text{ to a solution with relative residual} 0.44.
for k=1:niter
    inds = randperm(m);
    rk = inds(1:batch);
    Ark = A(rk,:);
    brk = b(rk);
    tk = norm(Ark*xk-brk)^2/norm(Ark'*((Ark*xk-brk))')^2;
    zk = zk-tk*Ark'*((Ark*xk-brk));
    xk = sign(zk).*max(abs(zk)-lambda,0)
end
Fast randomized least squares

Hot topic in “big data” and randomized algorithms
   ‣ sketching techniques that randomly sample rows & solve [Li, Nguyên & Woodruff, ’14]

\[
\minimize_x \frac{1}{2} \|RM(Ax - b)\|_2^2
\]

   ‣ randomized preconditioning, e.g. w/ QR factorization on reduced system [Avron et al., ’10]
   ‣ randomized Kaczmarz [Strohmer & Vershynin,’09; Zouzias & Freris, ’13]

These do not exploit structure (e.g. sparsity) & may require infeasible storage.
Leveraging the fold & threshold
– Randomized Iterative Shrinkage Thresholding Algorithm (RISTA)

Work /w for each iteration w/ independent randomized subsets of rows only

- simultaneous sourcing/phase encoding
- compressive sensing

\[ A \times x = b \]
\[ A_k \times x = b_k \]

\[ n_s n_f \ll n_s n_f \]
RISTA
– Randomized Iterative Shrinkage Thresholding Algorithm

1. \textbf{for } k = 0, 1, \ldots
2. \quad z_{k+1} = x_k - t_k A_k^* (A_k x_k - b_k)
3. \quad x_{k+1} = S_{\lambda_k} (z_{k+1})
4. \textbf{end for}

*where \( S_{\lambda}(x) = \text{sign}(x) \cdot \max(|x| - \lambda, 0) \) is soft thresholding and \( t_k \) are step lengths

\begin{itemize}
  \item relates to delicate “approximate” message passing theory [Montanari, ’09]
  \item reduces IO & works on “small” subsets of (block) rows in parallel
  \item only converges for special \( \{ A, A^H \} \) and tuned \( \lambda_k \)’s
  \item havocs continuation strategies & does not converge
\end{itemize}
Solution path

One−norm of solution vector

Relative two−norm residual

- w/o rerandomization
- w/ rerandomization

fails to converge
Relaxed sparsity objective

Consider $\lambda \to \infty$

\[
\begin{align*}
\text{minimize} \quad & \lambda \|x\|_1 + \frac{1}{2} \|x\|^2 \\
\text{subject to} \quad & Ax = b
\end{align*}
\]

- strictly convex objective known as “elastic” net in machine learning
- corresponds to Basis Pursuit for “large enough” $\lambda$
- corresponds to [Lorentz et. al.,’14]
  - sparse Kaczmarz for single-row $A_k$’s
  - linearized Bregman for full $A$’s
RISKA

– Randomized IS Kaczmarz Algorithm w/ linearized Bregman

1. \textbf{for} \ k = 0, 1, \cdots
2. \quad \mathbf{z}_{k+1} = \mathbf{z}_k - t_k \mathbf{A}_k^* (\mathbf{A}_k \mathbf{x}_k - \mathbf{b}_k)
3. \quad \mathbf{x}_{k+1} = S_\lambda (\mathbf{z}_{k+1})
4. \textbf{end for}

*where \( t_k = \frac{\| \mathbf{A}_k \mathbf{x}_k - \mathbf{b}_k \|^2}{\| \mathbf{A}_k^* (\mathbf{A}_k \mathbf{x}_k - \mathbf{b}_k) \|^2} \) are the step lengths

\begin{itemize}
  \item exceedingly simple flexible “three line” algorithm
  \item gradient descend on the dual problem, which provably converges
  \item total different role for \( \lambda \)
\end{itemize}
RISKA
– Randomized IS Kaczmarz Algorithm w/ linearized Bregman

1. for $k = 0, 1, \ldots$
2. $z_{k+1} = (z_k) - t_k A_k^* (A_k x_k - b_k)$
3. $x_{k+1} = S_\lambda(z_{k+1})$
4. end for

*where $t_k = \frac{\|A_k x_k - b_k\|^2}{\|A_k^* (A_k x_k - b_k)\|^2}$ are the step lengths

- exceedingly simple flexible “three line” algorithm
- gradient descend on the dual problem, which provably converges
- total different role for $\lambda$
RISTA
– Randomized Iterative Shrinkage Thresholding Algorithm

1. for $k = 0, 1, \ldots$
2. \[ z_{k+1} = x_k - t_k A_k^* (A_k x_k - b_k) \]
3. \[ x_{k+1} = S_{\lambda_k} (z_{k+1}) \]
4. end for

*where $S_{\lambda}(x) = \text{sign}(x) \cdot \max(|x| - \lambda, 0)$ is soft thresholding and $t_k$ are step lengths

- relates to delicate “approximate” message passing theory [Montanari, ’09]
- reduces IO & works on “small” subsets of (block) rows in parallel
- only converges for special $\{A, A^H\}$ and tuned $\lambda_k$’s
- havocs continuation strategies
Converges
Solution paths

\[ ||Ax - b||_2 \]

\[ \text{Pareto Curve} \]

\[ \text{Solution Path } \lambda = 1 \]

\[ \text{Solution Path } \lambda = 5 \]
Extension
– inconsistent systems

\[
\begin{align*}
\text{minimize} & \quad \lambda \|x\|_1 + \frac{1}{2} \|x\|^2 \\
\text{subject to} & \quad \|Ax - b\| \leq \sigma
\end{align*}
\]

via projections onto norm balls

1. \textbf{for} \; k = 0, 1, \ldots \\
2. \quad z_{k+1} = z_k - t_k A^*_k P_{\sigma} (A_k x_k - b_k) \\
3. \quad x_{k+1} = S_\lambda (z_{k+1}) \\
4. \textbf{end for}

*where \( P_{\sigma} (A_k x_k - b_k) = \max\{0, 1 - \frac{\sigma}{\|A_k x_k - b_k\|}\} \cdot (A_k x_k - b_k) \)
Role of threshold

\( \lambda \to \infty \)

- solution corresponds to BP (or BPDN)
- difficult to solve (like \( \lambda \to 0^+ \) for ISTA)
- thresholded components first step guaranteed to be in support

\[ 1 \ll \lambda \ll \infty \]

- iterations “auto tune” and do not wander off too far from optimal Pareto curve
- when threshold too large RISTA still makes progress
- room for acceleration w/ kicking techniques
Application

Least-squares (RTM) migration:

\[ \delta m = \sum_{ij} \nabla F_{ij}^H(m_0, q_{ij}) \delta d_{ij} \]

- too expensive to invert
- can we invert by touching data once?
Fast SPLSM w/ CS
– w/ randomized source subsets

\[
\begin{align*}
\text{minimize} & \quad \lambda \|x\|_1 + \frac{1}{2} \|x\|^2 \\
\text{subject to} & \quad \sum_{i,j} \|\nabla F_{ij}(m_0, q_{ij}) C^* x - \delta d_{ij}\| \leq \sigma
\end{align*}
\]

By iterating

1. for \( k = 0, 1, \cdots \)
2. \( \Omega \in [1 \cdots n_f], \Sigma \in [1 \cdots n_s] \) for \( \#\Omega \ll n_f, \#\Sigma \ll n_s \)
3. \( A_k = \{\nabla F_{ij}(m_0, \bar{q}_{ij}) C^*\}_{i \in \Omega, j \in \Sigma} \) with \( \bar{q}_{ij} = \sum_{l=1}^{n_s} w_l q_{i,l} \)
4. \( b_k = \{\delta d_{ij}\}_{i \in \Omega, j \in \Sigma} \) with \( \delta d_{ij} = \sum_{l=1}^{n_s} w_l d_{i,l} \)
5. \( z_{k+1} = z_k - t_k A_k^* P_\sigma (A_k x_k - b_k) \)
6. \( x_{k+1} = S_\lambda(z_{k+1}) \)
7. end for
Fast SPLSM w/ CS
– experimental setup

Data:
- 320 sources and receivers
- 72 frequency slices ranging from 3 – 12 Hz
- \( \delta \mathbf{d} = \mathbf{F}(\mathbf{m}) - \mathbf{F}(\mathbf{m}_0) \), generated with separate modeling engine

Experiments:
- one pass through the data with different batch/block sizes
- simultaneous vs sequential shots
- choose \( \lambda \) according to \( \max (t_1 \cdot A_1^*b_1) \) and number of iterations
- no source estimation – use correct source for linearized inversions
Fast SPLSM w/ CS
– 360 iterations, each w/ 8 frequencies/sim. shots
Fast SPLSM w/ CS
– 90 iterations, each w/ 16 frequencies/sim. shots
Fast SPLSM w/ CS
– 23 iterations, each w/ 32 frequencies/sim. shots
Fast SPLSM w/ CS
– 90 iterations, each w/ 16 frequencies/sim. shots
Fast SPLSM w/ CS
– 90 iterations, each w/ 16 frequencies/sequential shots

Fast SPLSM w/ CS
– on-the-fly source estimation

\[
\begin{align*}
\text{minimize} & \quad \lambda \|x\|_1 + \frac{1}{2} \|x\|^2 \\
\text{subject to} & \quad \sum_{i,j} \|\nabla F_{ij}(m_0, q_{ij})C^*x - \delta d_{ij}\| \leq \sigma
\end{align*}
\]

By iterating

1. for \( k = 0, 1, \ldots \)
2. \( \Omega \in [1 \cdots n_f], \Sigma \in [1 \cdots n_s] \) for \#\Omega \ll n_f, \#\Sigma \ll n_s
3. \( A_k = \{\nabla F_{ij}(m_0, s_i q_{ij})C^*\}_{i \in \Omega, j \in \Sigma} \) with \( q_{ij} = \sum_{l=1}^{n_s} w_l q_{i,l} \)
4. \( b_k = \{\delta d_{ij}\}_{i \in \Omega, j \in \Sigma} \) with \( \delta d_{ij} = \sum_{l=1}^{n_s} w_l \delta d_{i,l} \)
5. \( s_i = \frac{\sum_{j \in \Sigma} \langle \nabla F[m_0, q_{ij}]C^*x, \nabla F[m_0, q_{ij}]C^*x \rangle}{\sum_{j \in \Sigma} \langle \nabla F[m_0, q_{ij}]C^*x, \nabla F[m_0, q_{ij}]C^*x \rangle} \), \( A_k = \{\nabla F_{ij}(m_0, s_i q_{ij})C^*\}_{i \in \Omega, j \in \Sigma} \)
6. \( z_{k+1} = z_k - t_k A_k^*P_\sigma(A_k x_k - b_k) \)
7. \( x_{k+1} = S_\lambda(z_{k+1}) \)
8. end for
Fast SPLSM w/ source estimation
– experimental setup

Data:
- 320 sources and receivers
- 72 frequency slices ranging from 3 - 12 Hz
- $\delta \mathbf{d} = \nabla \mathbf{F} \delta \mathbf{m}$ inverse crime data

Experiments:
- one pass through the data with the same block size & different frequency-shot ratios
- simultaneous sources $\max (t_1 \cdot \mathbf{A}_1^* \mathbf{b}_1)$
- choose according to
- source estimation with delta Dirac as initial guess
- estimated source scaled w.r.t. true source
Fast SPLSM w/ source estimation
– 80 iterations, each w/ 72 frequencies/4 sim. shots & true source
Fast SPLSM w/ source estimation
– estimated source
Fast SPLSM w/ source estimation
– estimated source

- Estimated source vs. true source:
  - Frequency (HZ)
  - Amplitude
  - Phase
Fast SPLSM w/ source estimation
– 90 iterations, each w/ 16 frequencies/16 sim. shots w/ true source
Fast SPLSM w/ source estimation
– estimated source

[Diagram showing depth and lateral distance]
Fast SPLSM w/ source estimation
– estimated source
Fast SPLSM w/ source estimation
– 90 iterations, each w/ 4 frequencies/64 sim. shots w/ true source
Fast SPLSM w/ source estimation
– estimated source
Fast SPLSM w/ source estimation
– estimated source

**Amplitude vs Frequency (HZ)**
- Estimated source (red dots)
- True source (blue line)

**Phase vs Frequency (HZ)**
- Estimated source (red dots)
- True source (blue line)
Observations

Inversions can be carried out at cost (= batch size X # iterations) of ~1 RTM

For known source function:
  - quality is best for intermediate batch size & # of iterations
  - results for randomly selected sources are of similar quality
  - offers flexibility for parallelism

For unknown source function:
  - source function is best estimated when # of frequencies is not too low
  - quality is similar to cases where the source function is known
Extension
- imaging w/ surface-related multiples

Incorporate predictor of surface-related multiples via areal sources

\[ f(\mathbf{x}, \mathbf{w}) = \sum_{i \in \Omega} \sum_{j \in \Sigma} \left\| \delta \mathbf{d}_{i,j} - \nabla F[m_0, s, \mathbf{q}_j - \delta \mathbf{d}_{i,j}] C^* \mathbf{x} \right\|^2 \]
True image
RTM w/ multiples
Fast SPLSM w/ multiples by SPGI1

Simulation cost $\sim$1 RTM using all the data
Fast SPLSM w/ multiples by RISKA

Simulation cost $\sim 1$ RTM using all the data
Bottom line
– what you need

Access to \{A, A^H\} or \{A^H, A^H A\}
- migration, demigration or migration, Gauss-Newton Hessian
- norms for residual & gradient

Ability to subsample data
- randomized supershots or randomly selected shots in RTM
- or randomized traces (source/receiver) pairs in Kirchhoff migration

Some idea of max entry of $A_k^* b_k$
Conclusions & extensions

Algorithm:
- simple, converges & has very few tuning parameters
- offers maximal flexibility for
  - implementations that strike a balance between data- and model-space parallelism
  - extensions such as source estimation & imaging w/ multiples
  - other overdetermined problems such as AVO
- gets hifi/high-resolution images touching the data only once

Simple structure also offers flexibility to do
- adaptive sampling
- on-line recovery while randomized data streams in
Acknowledgements

This work was in part financially supported by the Natural Sciences and Engineering Research Council of Canada Discovery Grant (22R81254) and the Collaborative Research and Development Grant DNOISE II (375142-08). This research was carried out as part of the SINBAD II project with support from the following organizations: BG Group, BGP, CGG, Chevron, ConocoPhillips, DownUnder GeoSolutions, Hess, Petrobras, PGS, Subsalt Ltd, WesternGeco, and Woodside.

Thank you for your attention!

https://www.slim.eos.ubc.ca/