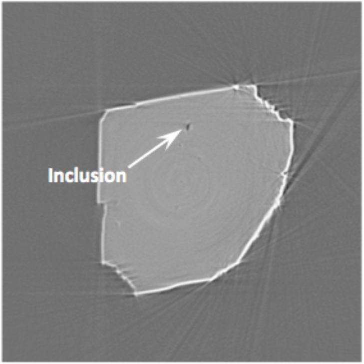
Automatic alignment for Tomographic reconstruction

Tristan van Leeuwen

Mathematical Institute, Utrecht University, The Netherlands

Tomography and Applications Discrete Tomography and Image Reconstruction 2015


Inclusion

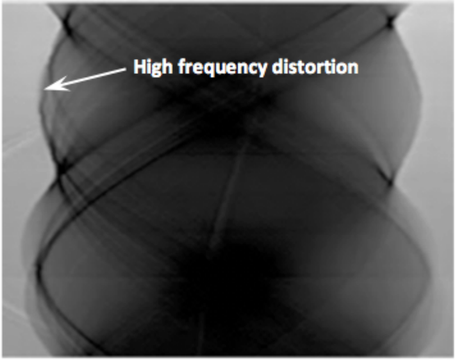

High frequency distortion

Table of contents

Mathematical formulation & approach

Express the reconstruction problem as an optimization problem:

$$\min_{\mathbf{x}} \|W\mathbf{x} - \mathbf{p}\|_2^2,$$

where

- $\mathbf{x} \in \mathbb{R}^n$ - image
- $\mathbf{p} \in \mathbb{R}^{lm}$ - projection data
- $W \in \mathbb{R}^{lm \times n}$ - projection matrix
- n - number of pixels (voxels)
- l - number of angles
- m - number of detector pixels

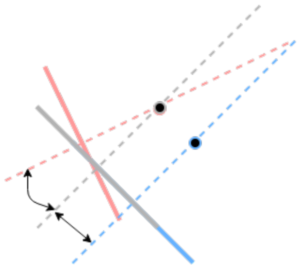
What if the matrix W is not known exactly?

What if the matrix W is not known exactly?

We parametrize W using a number of *alignment parameters* as $W(\mathbf{a})$.

What if the matrix W is not known exactly?

We parametrize W using a number of *alignment parameters* as $W(\mathbf{a})$.



For example, in a 2D parallel beam geometry, $\mathbf{a}$ consists of an angle and offset perturbation for each projection angle.

Bi-level optimization problem

$$\min_{\mathbf{x},\mathbf{a}} \|W(\mathbf{a})\mathbf{x} - \mathbf{p}\|_2^2,$$

Bi-level optimization problem

$$\min_{\mathbf{x},\mathbf{a}} \|W(\mathbf{a})\mathbf{x} - \mathbf{p}\|_2^2,$$

$$\min_{\mathbf{x},\mathbf{a}} \sum_{i=1}^{I} \|W_i(\mathbf{a}_i)\mathbf{x} - \mathbf{p}_i\|_2^2,$$

- W_i - projection matrix for the i^{th} angle
- $\mathbf{a}_i$ - alignment parameters for the i^{th} angle,
- $\mathbf{p}_i$ - projection data for the i^{th} angle.

Can we retrieve both $\mathbf{x}$ and $\mathbf{a}$?

Can we retrieve both $\mathbf{x}$ and $\mathbf{a}$?

- If W has full row-rank, we can fit the data for *any* $\mathbf{a}$
- How well we can retrieve $\mathbf{a}$ depends on $\mathbf{x}$
- Solution is at best unique up to global shifts and rotations

How do we solve the bi-level optimization problem?

$$\min_{\mathbf{x},\mathbf{a}} \sum_{i=1}^{I} \| W_i(\mathbf{a}_i)\mathbf{x} - \mathbf{p}_i \|_2^2,$$

How do we solve the bi-level optimization problem?

$$\min_{\mathbf{x},\mathbf{a}} \sum_{i=1}^{I} \|W_i(\mathbf{a}_i)\mathbf{x} - \mathbf{p}_i\|_2^2,$$

Exploit structure:

- quadratic in $\mathbf{x}$, non-linear in $\mathbf{a}$

How do we solve the bi-level optimization problem?

$$\min_{\mathbf{x},\mathbf{a}} \sum_{i=1}^{I} \|W_i(\mathbf{a}_i)\mathbf{x} - \mathbf{p}_i\|_2^2,$$

Exploit structure:

- quadratic in $\mathbf{x}$, non-linear in $\mathbf{a}$
- for fixed $\mathbf{x}$, separates in I independent low-dimensional problems for $\mathbf{a}_i$

Intermezzo: Variable projection

Intermezzo: Variable projection [Golub & Pereyra, 1973]

$$\min_{\mathbf{x},\mathbf{y}} \|A(\mathbf{y})\mathbf{x} - \mathbf{b}\|_2^2,$$

- eliminate $\mathbf{x}$ explicitly as $\mathbf{x} = A(\mathbf{y})^\dagger \mathbf{b}$,
- formulate a non-linear least-squares problem:

$$\min_{\mathbf{y}} \left\| \left(A(\mathbf{y})A(\mathbf{y})^\dagger - I \right) \mathbf{b} \right\|_2^2,$$

- this approach is superior to solving the joint problem with a GN method.

This idea can be generalized to solve problems of the form

$$\min_{\mathbf{x},\mathbf{y}} f(\mathbf{x},\mathbf{y}),$$

by solving for $\overline{\mathbf{x}}$ s.t. $\nabla_{\mathbf{x}} f(\overline{\mathbf{x}},\mathbf{y}) = 0$ and substituting this back:

$$\min_{y} \overline{f}(\mathbf{y}),$$

with $\overline{f}(\mathbf{y}) = f(\overline{\mathbf{x}}(\mathbf{y}),\mathbf{y})$.

Theorem [Aravkin & van Leeuwen, 2012]

Given a twice-differentiable function $f(\mathbf{x},\mathbf{y})$, define $\overline{f}(\mathbf{y}) = f(\overline{\mathbf{x}},\mathbf{y})$ with $\nabla_{\mathbf{x}} f(\overline{\mathbf{x}},\mathbf{y}) = 0$ then

$$\nabla\overline{f}(\mathbf{y}) = \nabla_{\mathbf{y}} f(\overline{\mathbf{x}},\mathbf{y}),$$

and

$$\nabla^2\overline{f} = \nabla^2_{\mathbf{y}} f - \nabla_{\mathbf{x},\mathbf{y}} f^T \left(\nabla^2_{\mathbf{x}} f\right)^{-1} \nabla_{\mathbf{x},\mathbf{y}} f.$$

Theorem [Aravkin & van Leeuwen, 2012]

Given a twice-differentiable function $f(\mathbf{x},\mathbf{y})$, define $\overline{f}(\mathbf{y}) = f(\overline{\mathbf{x}},\mathbf{y})$ with $\nabla_{\mathbf{x}} f(\overline{\mathbf{x}},\mathbf{y}) = 0$ then

$$\nabla\overline{f}(\mathbf{y}) = \nabla_{\mathbf{y}} f(\overline{\mathbf{x}},\mathbf{y}),$$

and

$$\nabla^2\overline{f} = \nabla^2_{\mathbf{y}} f - \nabla_{\mathbf{x},\mathbf{y}} f^T \left(\nabla^2_{\mathbf{x}} f\right)^{-1} \nabla_{\mathbf{x},\mathbf{y}} f.$$

Corollary

If, in addition, we require that $\nabla^2_{\mathbf{x}} f(\overline{\mathbf{x}},\mathbf{y}) \succeq 0$, then a local minimum $\overline{\mathbf{y}}$ of $\overline{f}$, together with the corresponding $\overline{\mathbf{x}}$ is a local minimum of f.

A gradient-based method for minimizing $\overline{f}$ can be implemented as follows

Algorithm

$$\mathbf{x}^{(k+1)} = \text{argmin}_{\mathbf{x}} f(\mathbf{x}, \mathbf{y}^{(k)})$$

$$\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} - \alpha_k \nabla_{\mathbf{y}} f(\mathbf{x}^{(k+1)}, \mathbf{y}^{(k)})$$

A gradient-based method for minimizing $\overline{f}$ can be implemented as follows

Algorithm

$$\mathbf{x}^{(k+1)} = \text{argmin}_{\mathbf{x}} f(\mathbf{x}, \mathbf{y}^{(k)})$$

$$\mathbf{y}^{(k+1)} = \mathbf{y}^{(k)} - \alpha_k \nabla_{\mathbf{y}} f(\mathbf{x}^{(k+1)}, \mathbf{y}^{(k)})$$

We are completely free to choose the most appropriate method to solve the inner and outer optimization problems.

Algorithms

Bi-level optimization problem

$$\min_{\mathbf{x},\mathbf{a}} \|W(\mathbf{a})\mathbf{x} - \mathbf{p}_i\|_2^2,$$

There are two natural choices:

- project out $\mathbf{a}$ (Align-then-Reconstruct),
- project out $\mathbf{x}$ (Reconstruct-then-Align).

Basic algorithm for A-R (modified)

$$\mathbf{a}_i^{(k+1)} = \underset{\mathbf{a}_i}{\text{argmin}} \, \|W_i(\mathbf{a}_i)\mathbf{x}^{(k)} - \mathbf{p}_i\|_2^2$$

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \Delta\mathbf{x}^{(k)}$$

- Use favourite method to solve alignment problems (independently)

Basic algorithm for A-R (modified)

$$\mathbf{a}_i^{(k+1)} = \underset{\mathbf{a}_i}{\operatorname{argmin}} \, \|W_i(\mathbf{a}_i)\mathbf{x}^{(k)} - \mathbf{p}_i\|_2^2$$

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \Delta\mathbf{x}^{(k)}$$

- Use favourite method to solve alignment problems (independently)
- Use GN method to compute $\Delta\mathbf{x}^{(k)}$:

Basic algorithm for A-R (modified)

$$\mathbf{a}_i^{(k+1)} = \underset{\mathbf{a}_i}{\text{argmin}} \, \| W_i(\mathbf{a}_i)\mathbf{x}^{(k)} - \mathbf{p}_i \|_2^2$$

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \Delta\mathbf{x}^{(k)}$$

- Use favourite method to solve alignment problems (independently)
- Use GN method to compute $\Delta\mathbf{x}^{(k)}$:

$$\Delta\mathbf{x}^{(k)} = W(\mathbf{a}^{(k+1)})^\dagger \left(\mathbf{p} - W(\mathbf{a}^{(k+1)})\mathbf{x}^{(k)}\right)$$

Algorithm for R-A

$$\mathbf{x}^{(k+1)} = W(\mathbf{a}^{(k)})^{\dagger}\mathbf{p}$$

$$\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \Delta\mathbf{a}^{(k)},$$

- Use standard iterative (Krylov) method to apply pseudo-inverse

Algorithm for R-A

$$\mathbf{x}^{(k+1)} = W(\mathbf{a}^{(k)})^{\dagger}\mathbf{p}$$

$$\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \Delta\mathbf{a}^{(k)},$$

- Use standard iterative (Krylov) method to apply pseudo-inverse
- Take $\Delta\mathbf{a}^{(k)}$ to be the (scaled) negative gradient:

Algorithm for R-A

$$\mathbf{x}^{(k+1)} = W(\mathbf{a}^{(k)})^{\dagger}\mathbf{p}$$

$$\mathbf{a}^{(k+1)} = \mathbf{a}^{(k)} + \Delta\mathbf{a}^{(k)},$$

- Use standard iterative (Krylov) method to apply pseudo-inverse
- Take $\Delta\mathbf{a}^{(k)}$ to be the (scaled) negative gradient:

  $\Delta\mathbf{a}^{(k)} \propto -J(\mathbf{x}^{(k+1)}, \mathbf{a}^{(k)})^T \left(W(\mathbf{a}^{(k)})\mathbf{x}^{(k+1)} - \mathbf{p}\right),$

  where $J(\mathbf{x}, \mathbf{a}) = \left(\frac{\partial W(\mathbf{a})\mathbf{x}}{\partial a}\right)$

Numerical examples

Simple linear projection model

$$\left(W_{h,\Delta\theta} f\right)(r,\theta) = \int_\Omega \mathrm{d}s\, f(s)\delta(r + h - n(\theta + \Delta\theta)\cdot s),$$

where $\Omega = [-1,1]^2$, $r \in [-1,1]$, $\theta \in [0,\pi)$, $n(\theta) = (\cos(\theta), \sin(\theta))^T$.

Simple linear projection model

$$(W_{h,\Delta\theta}f)(r,\theta) = \int_{\Omega} \mathrm{d}s\, f(s)\delta(r + h - n(\theta + \Delta\theta)\cdot s),$$

where $\Omega = [-1,1]^2$, $r \in [-1,1]$, $\theta \in [0,\pi)$, $n(\theta) = (\cos(\theta), \sin(\theta))^T$.

- Discretize domain using n pixels, use midpoint rule to approximate integral,

Simple linear projection model

$$(W_{h,\Delta\theta}f)(r,\theta) = \int_\Omega \mathrm{d}s\, f(s)\delta(r + h - n(\theta+\Delta\theta)\cdot s),$$

where $\Omega = [-1,1]^2$, $r \in [-1,1]$, $\theta \in [0,\pi)$, $n(\theta) = (\cos(\theta), \sin(\theta))^T$.

- Discretize domain using n pixels, use midpoint rule to approximate integral,
- Finite-width approximation $\delta_\epsilon \to \delta$ as $\epsilon \downarrow 0$,

Simple linear projection model

$$(W_{h,\Delta\theta}f)(r,\theta) = \int_\Omega \mathrm{d}s\, f(s)\delta(r + h - n(\theta + \Delta\theta)\cdot s),$$

where $\Omega = [-1,1]^2$, $r \in [-1,1]$, $\theta \in [0,\pi)$, $n(\theta) = (\cos(\theta), \sin(\theta))^T$.

- Discretize domain using n pixels, use midpoint rule to approximate integral,
- Finite-width approximation $\delta_\epsilon \to \delta$ as $\epsilon \downarrow 0$,
- Use l angles $\theta_i \in [0,\pi)$, m detector positions $r_i \in [-1,1]$,

Simple linear projection model

$$(W_{h,\Delta\theta}f)(r,\theta) = \int_\Omega \mathrm{d}s\, f(s)\delta(r + h - n(\theta+\Delta\theta)\cdot s),$$

where $\Omega = [-1,1]^2$, $r \in [-1,1]$, $\theta \in [0,\pi)$, $n(\theta) = (\cos(\theta),\sin(\theta))^T$.

- Discretize domain using n pixels, use midpoint rule to approximate integral,
- Finite-width approximation $\delta_\epsilon \rightarrow \delta$ as $\epsilon \downarrow 0$,
- Use l angles $\theta_i \in [0,\pi)$, m detector positions $r_i \in [-1,1]$,
- Alignment parameters $(h_i, \Delta\theta_i)$ for each angle

Simple linear projection model

$$(W_{h,\Delta\theta}f)(r,\theta) = \int_\Omega \mathrm{d}s\, f(s)\delta(r + h - n(\theta+\Delta\theta)\cdot s),$$

where $\Omega = [-1,1]^2$, $r \in [-1,1]$, $\theta \in [0,\pi)$, $n(\theta) = (\cos(\theta), \sin(\theta))^T$.

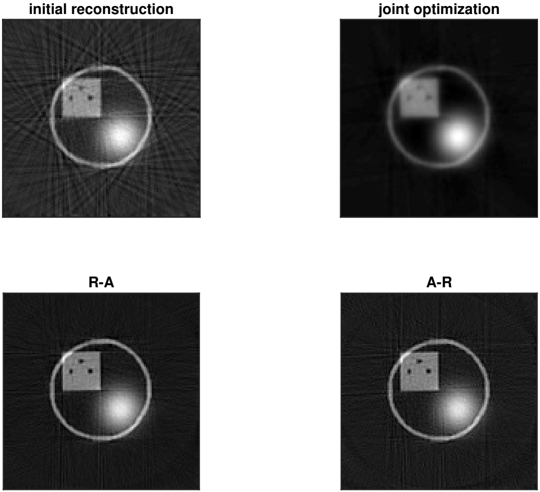
- Discretize domain using n pixels, use midpoint rule to approximate integral,
- Finite-width approximation $\delta_\epsilon \to \delta$ as $\epsilon \downarrow 0$,
- Use l angles $\theta_i \in [0,\pi)$, m detector positions $r_i \in [-1,1]$,
- Alignment parameters $(h_i, \Delta\theta_i)$ for each angle
- Required derivatives can be computed explicitly, use Quasi-Newton method

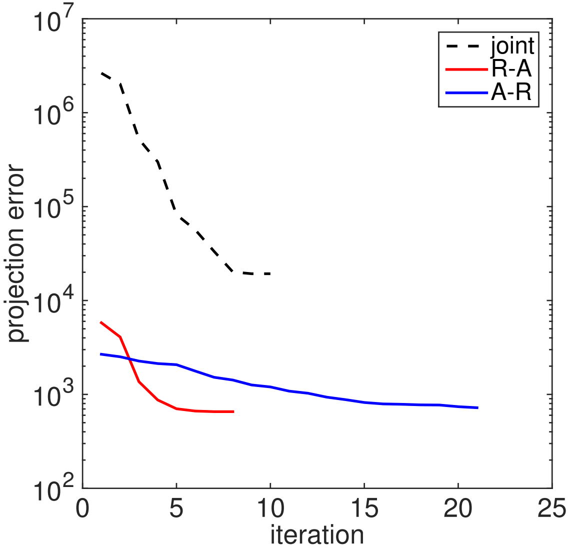
Simple linear projection model

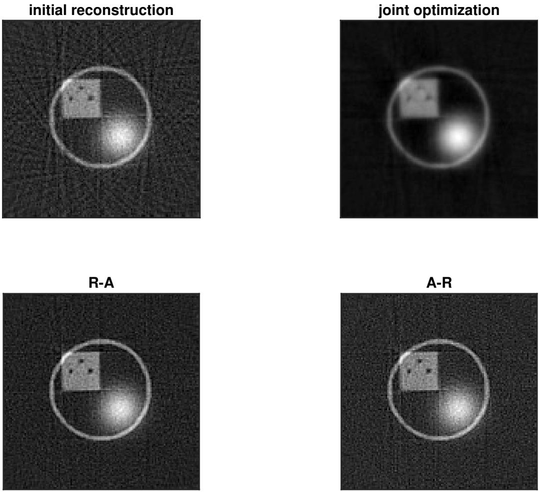
$$(W_{h,\Delta\theta} f)(r,\theta) = \int_\Omega \mathrm{d}s\, f(s)\delta(r + h - n(\theta+\Delta\theta)\cdot s),$$

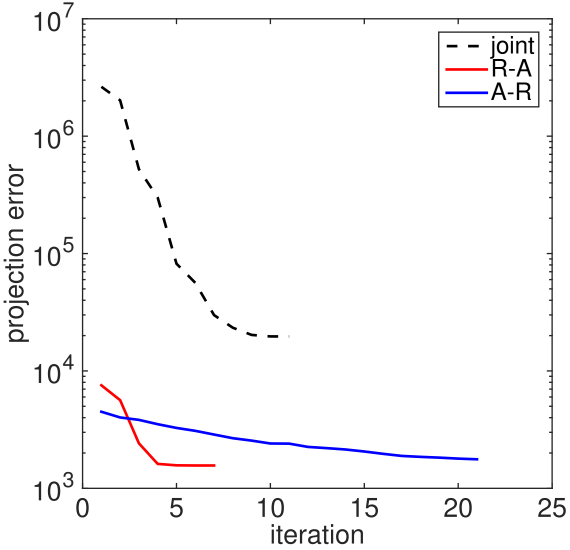
where $\Omega = [-1,1]^2$, $r \in [-1,1]$, $\theta \in [0,\pi)$, $n(\theta) = (\cos(\theta), \sin(\theta))^T$.

- Discretize domain using n pixels, use midpoint rule to approximate integral,
- Finite-width approximation $\delta_\epsilon \to \delta$ as $\epsilon \downarrow 0$,
- Use l angles $\theta_i \in [0,\pi)$, m detector positions $r_i \in [-1,1]$,
- Alignment parameters $(h_i, \Delta\theta_i)$ for each angle
- Required derivatives can be computed explicitly, use Quasi-Newton method


initial reconstruction
joint optimization
R-A
A-R


joint
R-A
A-R
projection error
iteration
10^7
10^6
10^5
10^4
10^3
10^2
0
5
10
15
20
25


initial reconstruction
joint optimization
R-A
A-R


joint
R-A
A-R
projection error
iteration
10^3
10^4
10^5
10^6
10^7
0
5
10
15
20
25

Possible issues:

- Gradient w.r.t. $\mathbf{a}$ might not be available; use appropriate derivative-free method [Nocedal & Wright, 2006]
- Renegade null-space elements may cause problems in A-R approach

Conclusions

- Unified mathematical framework to derive algorithms for automatic alignment

- Unified mathematical framework to derive algorithms for automatic alignment
- Decouple alignment and reconstruction problems by *variational projection*

- Unified mathematical framework to derive algorithms for automatic alignment
- Decouple alignment and reconstruction problems by *variational projection*
- Regularization (e.g., TV) can be included
- Further analysis needed to understand exactly why R-A approach appears to work better

Thank you!

Bibliography

Aravkin, A. Y. & van Leeuwen, T. (2012).
Inverse Problems, **28** (11), 115016.

Golub, G. & Pereyra, V. (1973).
SIAM Journal of Numerical Analysis, **10** (2), 413–432.

Nocedal, J. & Wright, S. J. (2006).
Numerical Optimization.
Amsterdam: Springer.