Rapid Implementation of Advanced Tomography Algorithms using the ASTRA Toolbox with Spot Operators

Folkert Bleichrodt
Centrum Wiskunde & Informatica (CWI)

Workshop on Large-scale Tomography
January 26, 2015
Outline

• I: The ASTRA toolbox with Spot operators

• II: Advanced reconstruction using the Student's t penalty
Part I: The ASTRA toolbox with Spot operators
Linear projection model

Based on line integrals:

\[
\int_{L} f(\xi) \, |d\xi| \approx \sum_{j=1}^{N} W_{ij} x_j
\]

The set of projections result in a linear system of equations

\[
Wx = p
\]
Towards large scale tomography

- 1022 projection images
- Of size 1000 x 524
- 16-bit per pixel
- 1 GB, projection data
- 8 GB, reconstruction
- 1 TB, projection matrix
From theory to high-performance algorithms

- Theoretic discoveries
- Algorithm development
- Software prototyping
- High performance large-scale software implementations

- Literature study
- Matlab/python
- CUDA (GPU) parallelization
From theory to high-performance algorithms

- Theoretic discoveries
- Algorithm development
- Software prototyping
- High performance large-scale software implementations
- Literature study
- Matlab/python
- CUDA (GPU) parallelization
From theory to high-performance algorithms

- Theoretic discoveries
- Algorithm development
- Software prototyping
- High performance large-scale software implementations
- Literature study
- Matlab/python
- ASTRA toolbox/Spot operators
Why Matlab?

- Rapid prototyping
- Many available packages
- High-level language syntax
The limitations of Matlab

Typical MATLAB implementations:

- Use full matrices
- Are implemented for a fixed geometry (parallel beam or cone beam)
- Do not scale well

```
>> sparse_reconstruct
Error using sparse
Out of memory. Type HELP MEMORY for your options.

Error in spalloc (line 17)
s = sparse([],[],[],m,n,nzmax);
```
The Spot toolbox

- A Spot operator can be used as matrix in Matlab, without storing the matrix in memory
- A Spot operator is based on the implementation of the forward and backward products of the matrix
- Spot operators can be combined in the same way as matrices can be combined

<table>
<thead>
<tr>
<th>Spot operator $W$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f(x) := A^x$</td>
</tr>
<tr>
<td>$g(y) := A'^y$</td>
</tr>
</tbody>
</table>

$W^x := f(x)$
$W'^y := g(y)$
Matrix operations supported by Spot

- Products: $A \times B$
- Concatenation: $[A,B]$ or $[A;B]$
- Powers: $A^n$
- Division: $A/B$
- Subscripted reference: $A(m:n, k:l)$
- Subscripted assignment: $A(m:n, k:l) = B$

and more ...
### Predefined Spot operators

#### Index of Operators

- opBernoulli
- opBinary
- opBlockDiag
- opBlockOp
- opCTranspose
- opChol
- opClass
- opConj
- opConvolve
- opCurvelet
- opDCT
- opDCT2
- opDFT
- opDFT2
- opDiag
- opDictionary
- opDirac
- opEmpty
- opExcise
- opExtend
- opEye
- opFactorization
- opFoG
- opFunction
- opGaussian
- opHaar
- opHaar2
- opHadamard
- opHeaviside
- opHermitian
- opImag
- opInverse
- opKron
- opLDL
- opLU
- opMask
- opMatrix
- opMinus
- opOnes
- opOrthogonal
- opPInverse
- opPermutation
- opPower
- opQR
- opReal
- opRestriction
- opSparseBinary
- opStack
- opSubsAsgn
- opSubsRef
- opSum
- opToepGauss
- opToepSign
- opToeplitz
- opTranspose
- opUnaryMinus
- opWavelet
- opWavelet2
- opWindow
- opZeros
- opiChol
- opiLU
The ASTRA Spot operator

- GPU accelerated forward and backprojection are linked to a Spot operator “opTomo”

\[
f(x) \rightarrow \text{astra\_create\_sino\_gpu}(\ldots) \\
g(x) \rightarrow \text{astra\_create\_backprojection\_gpu}(\ldots)
\]
Example 1: Least-squares

```matlab
1  % Create a tomography Spot operator 'opTomo'
2  W = opTomo('cuda', proj_geom, vol_geom);

4  % can be used to create projection data as a vector
5  p = W*im(:);

7  % reconstruction using a Krylov subspace method
8  x = lsqr(W,p);
```
Example 2: total variation minimization with Chambolle-Pock

\[
\text{minimize } \| \mathbf{W} \bar{x} - \bar{p} \|_2^2 + \lambda \| \text{TV}(\bar{x}) \|_1
\]

\[
L \leftarrow \|(A, \nabla)\|; \quad \tau \leftarrow 1/L; \quad \sigma \leftarrow 1/L
\]
\[
x = y = z = 0,
\]
repeat
\[
y \leftarrow (y + \sigma (Ax - b))/\max(1_D, |y + \sigma (Ax - b)|)
\]
\[
z \leftarrow \lambda (z + \sigma \nabla x)/\max(\lambda 1_I, |z + \sigma \nabla x|)
\]
\[
x \leftarrow 2(x - \tau A^T y + \tau \text{div } z)
\]
until convergence

---

Example 2: TV-min

```
1  % Tomography operator
2  W = opTomo('cuda', proj_geom, vol_geom);
3  % Total variation operator
4  TV = opTV(m, n);
5
6  x_tv = chambollePock(W, TV, p, 100);
```
Example 2: TV-min

phantom  LSQR  TV-min
Example 2: masked TV-min

```matlab
1  % use a square mask
2  mask = zeros(m,n);
3  % in the middle
4  mask(129:384,129:384) = 1;
5  M = opMask(logical(mask(:)));
6
7  TV_masked = opBlockDiag(M,M)*TV;
8  x_tv = chambollePock(W, TV_masked, p);
```

Mask represents smooth area
Example 2: masked TV-min

phantom image

TV-min

least squares

masked TV-min
Example 3: sparse wavelet reconstruction

```matlab
1  % Projection operator
2  W = opTomo('cuda', proj_geom, vol_geom);
3  % 2D wavelet operator
4  B = opWavelet2(n, n, 'Haar', [], levels);
5  sigma = 200;
6  y_spgl1 = spgl1(W*B', sinogram(:,), [], sigma);
```

solving:

$$\text{minimize } \| y \|_1 \text{ such that } \| WB^Ty - p \|_2 \leq \sigma$$
Example 3: sparse wavelet reconstruction

ground truth  
LSQR  
SPGL1

Reconstruction ($1k \times 1k$) from 15 projections
Part II: Advanced reconstruction using the Student's t penalty
Idea: alternative penalty functions

Instead of least squares:

\[
\minimize_{\vec{x}} \| W \vec{x} - \vec{p} \|_2^2
\]

use alternative general penalties:

\[
\minimize_{\vec{x}} \rho \left( W \vec{x} - \vec{p} \right)
\]
Motivation: outliers

Model inconsistencies cause outliers in the residuals, causing artifacts

Outliers have large penalties in l2-norm
Example: defective camera pixels

\[
\minimize_{\vec{x}} \rho \left( W \vec{x} - \vec{p_1} - \vec{p_2} \right)
\]
Student's t-distribution

Student's t, statistical distribution function:

\[ \rho(r) \propto \prod_{i=1}^{n} (1 + r_i^2 / \nu)^{-(\nu+1)/2} \]

leading to maximum likelihood estimation:

\[ \min_r -\log(\rho(r)) = \min_r \sum_{i=1}^{n} \log(1 + r_i^2 / \nu) \]
Least squares and Student's t

Penalty functions

Gradients
Minimizing the Student's t penalty

Maximum likelihood estimation:

\[
\min_{x} \rho(Wx - \rho)
\]

\[
\rho(r) = \sum_i \log(1 + r_i^2 / \nu)
\]

solved using Newton's method:

\[
x_{k+1} = x_k + \alpha_k s_k
\]

where the descent direction is found by solving:

\[
[H\rho(x_k)] s_k = \nabla \rho(x_k)
\]
Results: random projections

- 256 x 256 x 256 test image
- 180 projections
- Of which 45 replaced by white noise
Results: defective camera pixels

- ground truth
- least-squares solution
- Student’s t solution

![Graph showing MSE vs. number of defective pixels for LSQR and Student’s t solutions. The graph indicates that Student’s t solution has a lower MSE for a given number of defective pixels.]
Conclusions

- The ASTRA-Spot operator enables the use of Matlab scripts to large-scale data
- The Spot operator makes implementing high-performance advanced algorithms easy
- The l2-norm assigns too large penalties to outliers
- Outliers can be ignored automatically by simply adjusting the penalty function