Algebraic Methods for Computed Tomography

Preparation

Download the AIR Tools MATLAB software from the home page

http://www.compute.dtu.dk/~pch/AIRtools

unpack the zip file, and place all the functions in a directory. Start MATLAB and go to the directory you created above. Have fun with the exercises!

Our Point of View

The goal of these exercises is to illustrate various aspects of algebraic iterative methods, where the key issue is that we represent the forward model by means of a matrix A instead of a transformation (such as the Radon transform). To illustrate the difference, consider how we generate a test problem and compute a reconstruction by means of the two approaches:

**ALGEBRAIC**

\[
\begin{align*}
N &= 64; \\
\text{theta} &= 3:3:180; \\
[A,b,x] &= \text{paralletomo}(N,\text{theta}); \\
x_{\text{art}} &= \text{kaczmarz}(A,b,k);
\end{align*}
\]

**TRANSFORM**

\[
\begin{align*}
N &= 64; \\
\text{theta} &= 3:3:180; \\
X &= \text{phantom}(N); \\
S &= \text{radon}(X,\text{theta}); \\
X_{\text{fbp}} &= \text{iradon}(S,\text{theta});
\end{align*}
\]

At this stage we have the relations

\[
\begin{align*}
X &= \text{reshape}(x,N,N) \\
S &\approx \text{reshape}(b,p,n\theta) \\
X_{\text{fbp}} &\approx \text{reshape}(x_{\text{art}},N,N)
\end{align*}
\]

where \(n\theta = \text{length(\text{theta})}\) and \(p = \text{length(b)}/n\theta\).

Part 1 – Setting Up and Solving CT Systems

Exercise 1. A Very Small System

We consider the following CT problem with a \(2 \times 2\) image and 5 parallel rays (where the geometry is scaled such that the length of each ray through one pixel is 1):
Convince yourself that the corresponding system of linear equations $Ax = b$ takes the form

$$
\begin{pmatrix}
1 & 0 & 0 & 0 \\
1 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 \\
0 & 1 & 0 & 1 \\
0 & 0 & 0 & 1
\end{pmatrix}
\begin{pmatrix}
x_1 \\
x_2 \\
x_3 \\
x_4
\end{pmatrix}
= 
\begin{pmatrix}
1 \\
3 \\
5 \\
7 \\
4
\end{pmatrix}.
$$

(1)

Can you guess the solution? Now enter this system into MATLAB:

```matlab
>> A = [1 0 0 0;1 0 1 0; 0 1 1 0; 0 1 0 1; 0 0 0 1]
>> b = [1; 3; 5; 7; 4]
```

Then use the function `kaczmarz` from AIR TOOLS to perform 20 iterations with starting vector $x^{(0)} = 0$. Note that one “iteration” in AIR TOOLS always means one cycle or pass over all the rows of the matrix! The iteration vectors are stored as the columns of the $4 \times 20$ output matrix $X$:

```matlab
>> kmax = 20;
>> X = kaczmarz(A,b,1:kmax);
```

(The semicolon suppresses printing on the screen.) To see the last iteration vector, write $X(:,kmax)$ – is it correct, i.e., is $A\cdot X(:,kmax) = b$?

**Exercise 2. Convergence Study**

The exact solution to the above problem (the “ground truth”) is $\bar{x} = (1, 3, 2, 4)^T$. Let us study how fast the iteration vectors of Kaczmarz’s method converge to this solution, i.e., in each iteration we look at the largest error:

$$
e_k = \max_{1 \leq j \leq 4} |\bar{x}_j - x^{(k)}_j|, \quad k = 1, 2, \ldots, 20.
$$

(2)

In MATLAB this computation takes the form

```matlab
>> for i=1:k, e(i,1) = max(abs(x-X(:,i))); end
```
Now let us plot the results:

```matlab
>> figure(1)
>> subplot(2,1,1), plot(X')
>> subplot(2,1,2), semilogy(e)
```

On the top plot you should see the components of the iteration vector $x^{(k)}$ converge to $\bar{x}$, and on the bottom plot you should see that the largest error in each iteration decreases very fast.

As a matter of fact, after the first iteration for this problem we have the relation between the maximum errors:

$$e_{k+1} = C \cdot e_k,$$

$k = 2, 3, \ldots, 20,$

(3)

where $C$ is a constant. What is the value of $C$?

**Exercise 3. Setting Up a More Realistic Test Problem**

Let us now use the function `paralleltomo` from AIR TOOLS to generate a more realistic test problem $A\, x = b$. The ground truth image is $N \times N$ with $N = 64$, and we use the projection angles $3^\circ, 6^\circ, 9^\circ, \ldots, 180^\circ$. Hence, in MATLAB we write:

```matlab
>> N = 64;
>> theta = 3:3:180;
>> [A,b,x] = paralleltomo(N,theta);
```

Note that both $x$ and $b$ are vectors, because that is how we formulate the problem in linear algebra terminology; how long are the two vectors?

To display these two vectors as images (the object and the sinogram, respectively), use the following MATLAB commands:

```matlab
>> figure(1), imagesc(reshape(x,N,N)), colorbar
>> ntheta = length(theta);
>> p = length(b)/ntheta; % Number of rays.
>> figure(2), imagesc(theta,1:p,reshape(b,p,ntheta)), colorbar
```

Finally, let us convince ourselves that the system matrix $A$ is indeed a very sparse matrix, i.e., it consists mainly of zeros. To do that, use the MATLAB function `spy(A)` to display the nonzeros of the matrix.

**Exercise 4. Using Kaczmarz’s Method to Solve the Test Problem**

Now we use Kaczmarz’s method to solve the above test problem, and to get a feeling of the convergence let us perform 20 iterations and plot these iterations as images:

```matlab
>> k = 1:20;
>> X = kaczmarz(A,b,k);
>> figure(4)
>> for i=1:kmax, subplot(4,5,i), imagesc(reshape(X(:,i),N,N)), end
```
You should see a very slow progression of the reconstructions towards the ground truth image.

Finally, try to run `kaczmarz` with 50, 100, 200, 400 and 800 iterations and compute the corresponding maximum errors. This can be done as follows (be prepared that it will take a little while):

```matlab
>> k = [50,100,200,400,800];
>> X = kaczmarz(A,b,k);
>> e = zeros(length(k),1)
>> for i=1:length(k), e(i) = norm(x-X(:,i),inf); end
>> e
```

You can also plot the results using `semilogy(k,e)`. You should see that the convergence is really quite slow.

### Part 2 – Convergence Analysis and Null Space

#### Exercise 5. Using Cimmino’s Method to Solve the Test Problem

Perform the same experiments as in Exercise 4, but using Cimmino’s method instead of Kaczmarz’s method. All you need to do, is to substitute `cimmino` for `kaczmarz` in your MATLAB code.

Compared to Kaczmarz, would you say that the convergence of Cimmino is faster, slower, or about the same?

#### Exercise 6. Convergence Analysis for Kaczmarz and Cimmino

We will now perform a more thorough examination of the convergence of Kaczmarz’s and Cimmino’s methods for the above test problem. Following the convergence results from the slides, and to simplify our analysis, we will assume that the maximum errors $e_k$ in each iteration satisfy the relation

$$ e_k = C^k e_0, \quad k = 1, 2, 3, \ldots, $$

where $e_0$ is the initial error and $C$ is a constant that depends on the problem and the method.

Now assume that we know the error for two iterations $k$ and $K$ with $K > k$. Show, using (4), that we can compute the constant $C$ as

$$ C = \exp \left( \frac{\log(e_K/e_k)}{K - k} \right). $$

Do this for Kaczmarz’s and Cimmino’s methods in the two previous exercises, e.g., using the results that you computed for 400 and 800 iterations. What are the two constants for the two methods? Which one is better?
Exercise 7. The Significance of the Null Space

The above figure shows a small problem with a 3 × 3 image and 8 rays that are pairwise parallel (you can think of this situation as being recorded by 2 detector pixels and the object rotated 0°, 45°, 90° and 135°. The corresponding system matrix is

\[
A = \begin{pmatrix}
1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\
1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\
& s & s & 0 & 0 & 0 & 0 & 0 & 0 \\
& 0 & 0 & 0 & 0 & s & 0 & s & s \\
& s & s & 0 & 0 & s & 0 & s & s \\
& 0 & 0 & s & 0 & 0 & s & 0 & s \\
& 0 & s & s & 0 & 0 & s & 0 & s & s
\end{pmatrix}, \quad s = 1/\sqrt{2}. \tag{6}
\]

Consider an example where the ground truth image, represented as a vector, is

\[\bar{x} = (1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8 \ 9)^T. \tag{7}\]

Compute the corresponding projection data \(b = A\bar{x}\) and perform “enough” iterations with Kaczmarz’s method so ensure that the iterations have converged. Can you reconstruct \(\bar{x}\) in (7)? Also try to compute the solution by means of Cimmino’s method – you’ll need to perform more iterations with this method to ensure convergence. Does this produce a better reconstruction of \(\bar{x}\)?

To explain the difficulty with reconstructing this \(\bar{x}\), consider the following questions: what is the rank of \(A\) and what is the null space \(N(A)\)? (MATLAB: \texttt{rank} and \texttt{null}) Can you propose a different \(\bar{x}\) which can be perfectly reconstructed?

Part 3 – The Optimization Viewpoint

Exercise 8. Consistent and Inconsistent Overdetermined Problems

Let us try to experimentally verify the part of the overview table on slide 49 for the case \(m > n = r\), which states that:
• For consistent systems \( b \in \mathcal{R}(A) \), both Kaczmarz and Cimmino converge to the least-squares solution \( x_{LS} \) which is identical to the weighted least-squares solution \( x_{LS,M} \).

• For inconsistent systems \( b \notin \mathcal{R}(A) \), Cimmino converges to \( x_{LS,M} \) which is different from \( x_{LS} \), while Kaczmarz exhibits cyclic convergence.

We note that in MATLAB, for a full-rank matrix we compute the least-squares solution \( x_{LS} \) by means of \( x_{LS} = A\backslash b \).

Generate the test problem from Exercise 1 with a \( 5 \times 4 \) system matrix \( A \), and solve the consistent system \( Ax = b \) by means of Kaczmarz and Cimmino. Check that both methods converge to the same solution, and that this solution is identical to the least-squares solution \( x_{LS} \).

We will now change the right-hand side \( b \) by adding a component that is orthogonal to the range \( \mathcal{R}(A) \):

\[
\tilde{b} = b + 0.05 e \quad \text{with} \quad e = \begin{pmatrix} 1 \\ -1 \\ 1 \\ -1 \\ 1 \end{pmatrix}.
\] (8)

Use MATLAB to verify that \( e \) is orthogonal to \( \mathcal{R}(A) \), i.e., that

\[
c_i \cdot e = 0 \quad \text{for} \quad i = 1, 2, 3, 4.
\] (9)

Then create the perturbed right-hand side \( \tilde{b} \), solve the corresponding inconsistent system with Cimmino, and compare with the least-squares solution \( \tilde{x}_{LS} = A\backslash \tilde{b} \). Now you should see that Cimmino converges to a vector that is different from \( \tilde{x}_{LS} \).

If you also use Kaczmarz then you actually don’t see the cyclic convergence because \( kaczmarz \) returns the iteration vectors after each sweep over the rows, see the figure on slide 24.

**Exercise 9. The Advantage of Constraints**

We know from the underlying physics of the problem that the attenuation coefficients we wish to reconstruct – the elements of the vector \( x \) – are nonnegative. And very often we also know an upper bound on the elements of this vector. Hence it often makes a lot of sense to include box constraints in the reconstruction process.

In this exercise we return to the test problem from Exercises 3–4, but here we will impose the box constraints

\[
x \in \mathcal{C} = [0,1]^n \quad \Leftrightarrow \quad 0 \leq x_i \leq 1 \quad \text{for} \quad i = 1, 2, \ldots, n.
\] (10)

These constraints are specified in the function \( kaczmarz \) by means of the \texttt{options} input, as follows:
>> k = 1:20;
>> options.box = 1;
>> X = kaczmarz(A,b,k,[],options);

Setting the upper bound options.box to 1 automatically invokes the lower bound 0. Plot the 20 reconstructions as images; you should see that they are considerably better than those from Exercise 4.

**Exercise 10. Convergence Analysis for the Constrained Algorithm**

Do the constrained iterates converge faster? To answer this question, we will repeat Exercises 4 and 6 for the constrained algorithm, still looking at iterations 50, 100, 200, 400 and 800, and once again estimating the constant $C$ in (4) via the expression in (5). You should now obtain a slightly smaller value of $C$ which supports that the constrained method converges faster.

*Optional bonus question.* One might argue that we can also obtain good results by using the unconstrained Kaczmarz algorithm and then projecting the computed result on the set $C = [0,1]^n$, i.e., by “chopping” all elements outside this interval. In MATLAB, if $X$ is the output from kaczmarz without constraints, then projection or “chopping” is done like this:

$$X = \max(X,0); \quad X = \min(X,1);$$

You should see that it is preferable to include the constraints in the algorithm.

**Exercise 11. Comparison with Filtered Back Projection**

In this last exercise we compare the constrained Kaczmarz reconstruction for $k = 800$ iterations from the previous exercise with the FBP solution computed by means of fbp(A,b,theta) – this is a function from AIR Tools that is equivalent to MATLAB’s iradon except that it take the system matrix as input.

When displaying the FBP solution, note that some elements are outside the interval $[0,1]$, so you may want to “chop” them or set the figure color axis to this interval by means of caxis([0,1]). Which reconstruction is better?

Repeat this comparison using a *limited-angle* problem with $N = 64$ and theta = 3:3:120. You should see that the constrained Kaczmarz solution is much better than the FBP solution.