Exercises – Row Action Methods and AIR Tools

Per Christian Hansen, DTU Compute

Preparation

Download the AIR Tools MATLAB software from the home page http://www2.imm.dtu.dk/~pch/AIRtools, unpack the zip file, and place all the functions in a directory. Also download the additional files from the home page http://www2.imm.dtu.dk/~pch/AIRtools/Tutorial and put them in the *same* directory. Start MATLAB and go to the directory you created above. Have fun with the exercises!

There are plenty of exercises; if you don't finish all of them today, you can continue on the second day of computer-labs. Also, some of the exercises below include the use of the CGLS algorithm; if you are not already familiar with this method, just think of it as "yet another iterative method."

Exercise 1: how to set up a test problem and solve it

This exercise demonstrates how to create a simple test problem and solve it by means of the functions in AIR Tools.

1. Generate a noise-free test problem with a smooth image by means of the function **odftomo**, using the call **[A,bex,xex] = odftomo(N)**. The image is *N*-by-*N* and you should use *N* = 64; for the other parameters use the default values. Show the exact image **xex** in a figure window; note that the variable **xex** is a vector, so it must be reshaped into an *N*-by-*N* image via **reshape(xex,N,N)**.

2. Add noise to the exact data **bex** by means of the following commands:

```
eta = 0.02;
e = randn(size(b));
e = eta*norm(b)*e/norm(e);
b = bex + e;
```

This adds a noise vector **e** with Gaussian white noise, scaled such that $||\,\mathbf{e}\,||_2 / ||\,\mathbf{bex}\,||_2 = 0.02$ (corresponding to approximately 2% noise).

3. Compute the first 9 iterations from the projected ART algorithm by means of:

```
options.nonneg = true;
Xart = kaczmarz(A,b,1:9,[],options);
```

This use of the variable **options** enforces nonnegativity constraints on the reconstructions. The output variable **Xart** is an array with 9 columns, and column no. *k* holds the *k*-th iteration (here, one iteration is defined as one sweep through the columns of the matrix).

4. Show the 9 iterations; you should see that the reconstructions start to get noisy after iteration 3 or 4 (i.e., that semi-convergence takes place).

5. Repeat steps 3 and 4 with the CGLS algorithm (use the function **cglsAIR**). You should see that CGLS is able to produce a smoother – and thus better – reconstruction after about 4 iterations.

Exercise 2: semi-convergence of ART and CGLS

This exercise demonstrates how to investigate the semi-convergence of the ART and CGLS algorithms, by comparing the reconstructions to the exact solution.

1. Follow steps 1 and 2 from the previous exercise to generate a test problem with noisy data; but this time use the function **binarytomo** to generate a binary test image, and add 3% noise.

2. Compute the first 100 iterations from the ART algorithm, again using **options** to enforce non-negativity.

3. Compute the "error history", i.e., the norm of error in the *k*-th iterate as a function of *k*:

```
for k=1:100
   err(k) = norm( x - Xart(:,k) );
end
```

This computes the 2-norm (or scaled RMS) of the error; alternatively, you can use use **norm(…,1)** to compute the 1-norm (the sum of absolute deviations) or **norm(…,inf)** to compute the infinity-norm (the largest absolute deviation).

4. Plot the error history. You should see that the error initially decreases, while at some point it starts to increase slowly again. If you use the 2-norm error, then the minimum should occur after about 60 iterations.

5. Repeat steps 3 and 4 using the CGLS algorithm; you can improve the CGLS reconstructions by setting negative values to zero by means of the command:

```
Xcgls(Xcgls < 0) = 0;
```

where **Xcgls** is the array of CGLS iterates. The minimum should now occur after about 20 iterations.

6. Which algorithm gives the best reconstruction?

Exercise 3: the influence of the relaxation parameter

This exercise illustrates that the choice of the relaxation parameter has an influence on the rate of semi-convergence – but not on the accuracy of the reconstruction.

1. Generate the same test problem as in Exercise 2, except that perhaps the problem size could be reduced to $N = 32$ to reduce computing time. Keep the noise level at 3%.

2. For each of the relaxation parameters **lambda** = 0.05, 0.1, 0.15, 0.2, 0.3, 0.5, 0.7, and 1 compute and plot the error histories. Use Kaczmarz's method and a maximum of 50 iterations. If most of (or all) the minima are attained at 100 iterations, please generate a new instance of the noise and run the experiment again. Observe what happens to the error histories as **lambda** increases.

3. Use the function **trainLambdaART** to find a good value of the parameter **lambda** by training (such that the smallest error is obtained within 100 iterations). What is this value? Is it in accordance with the results from step 2?

Exercise 4: stopping rules

This exercise illustrates how to use the built-in stopping rules – which is important in real applications where we cannot study the error histories.

1. Generate the same test problem as in Exercise 3 (again using *N* = 32 and 3% noise).

2. Use training to determine a good relaxation parameter **lambda**.

3. Use the ART method with the **lambda**-value from step 2 and the Discrepancy Principle stopping criterion. For the parameter needed in this method, use **taudelta = norm(e);** where **e** is the error vector. Set the maximum number of iterations to 30. What is the error in the reconstruction?

4. Repeat step 3, but using the NCP stopping rule – which has the advantage that it does not need an estimate of the norm of the errors. Compare this reconstruction to the one from step 3.

5. The above results vary when you use different noise realizations – try that.

Exercise 5: the advantage of including the non-negativity projection in the iterative algorithm

This exercise illustrates that including the projection in each step of the algorithm gives better reconstructions than if the projection is applied after the iterations vectors have been computed.

1. Generate the same test problem as in Exercise 3 (again using *N* = 32 and 3% noise), and use the value of lambda you found in Exercise 4.

2. Run *k* = 50 ART iterations with no non-negativity projection, and determine the best reconstruction.

3. Now apply the projection to the ART iterates by means of **X(X < 0) = 0;** – where **X** is the output from **kaczmarz** – and determine the best reconstruction of these "after-the-fact projected solutions."

4. Finally, run *k* = 50 ART iterations with non-negativity projection (set **options.nonneg = true**), and determine the best reconstruction. You see that the last approach gives the best reconstruction.

Exercise 6: comparison with filtered back projection (inverse Radon transform)

This exercise illustrates how one can compare a reconstruction computed by one of the AIR Tools algorithms with the one computed by means of filtered back projection, as implemented in MATLAB's **iradon** (inverse Radon transform) function.

The procedure is somewhat tedious, as MATLAB does not allow easy control of the geometry assumed in **iradon** – hence we must use the corresponding function radon (for the forward computation) to generate the matrix A associated with the algebraic formulation.

Run the script **Ex6** to perform this exercise; the comment lines in the file explain the details.

Exercise 7: verification of the Landweber filter factors (no coding involved)

Use the SVD of A to verify, by insertion, that the iteration

$$x^{k+1} = x^k + \lambda A^T (b - A x^k), \quad k = 1,2,\ldots$$

is satisfied when Landweber iterate x^k is given by

$$x^k = V \Phi^{[k]} \Sigma^{-1} U^T b , \qquad \Phi^{[k]} = \mathrm{diag}(\phi_i{}^{[k]}), \qquad \phi_i{}^{[k]} = 1 - (1 - \lambda \sigma_i^2)^k .$$

Also show that for $\lambda \sigma_i^2 << 1$ we have $\phi_i{}^{[k]} \approx k \lambda \sigma_i^2$.