# Math 440 – Computational Inverse Problems Fall 2017

Instructor: Prof. Erkki Somersalo

Exercise 5

1. Build your own CGLS algorithm. You can probably find easily CGLS codes, but please write your algorithm from scratch so that you think through the order in which the matrix-vector products need to be computed. Recall: you should only have **two** matrix-vector products per iteration.

   Test your code with the inverse Laplace transform problem that you implemented for Assignment 4. Run it with two noise levels, a (relatively) high noise (e.g., $\sigma = 5 \times 10^{-3}$), and low noise (e.g., $\sigma = 10^{-5}$).

2. Consider the backwards heat equation of Assignment 1. Download the `HeatEquationExample.m` that gives you a forward solver of the heat equation,

$$u_0(x) = u(x,0) \mapsto u(x,T) = u_T(x),$$

   using the ODE solver `ode15s`. Set for simplicity the boundary values at $x = 0, 1$ equal to zero. This solver computes effectively the matrix-vector product,

$$u_T = \mathsf{A} u_0,$$

   and based on the analysis of the propagation operator, the matrix $\mathsf{A}$ is symmetric and positive definite.

   (a) Define an initial value $u_0^*$ in a fine grid ($N = 350$, for instance) as a box car function,

$$u_0^*(t_j) = \begin{cases} 1, & \text{if } 1/3 < t_j < 2/3, \\ 0 & \text{otherwise.} \end{cases}$$

   Propagate the box-car function for $T = 1$. This function represents the noiseless data for the inverse problem of recovering the initial value. To avoid the infamous inverse crime, as in Assignment 1, choose now the discretization grid ($n = 100$ is a good size), and interpolate the noiseless data from the fine grid ($N = 350$) to the coarse grid ($n = 100$.) You should have the interpolation matrix ready from Assignment 1. Add some noise, e.g., noise level 0.1% of the maximum of the noiseless vector.

   (b) Find an approximation for $u_0$ using your own matrix-free CGLS algorithm with the stopping rule based on Morozov discrepancy principle.

3. Consider the previous example, but with downsampling of the data. Let $\mathsf{P}_k$ be a matrix that downsamples the data, that is, $\mathsf{P}_k$ picks every $k$th data point. For instance, with $k = 2$,

$$\mathsf{P}_2 u = \begin{bmatrix} u_2 \\ u_4 \\ \vdots \\ u_{[n/2]} \end{bmatrix}, \quad [n/s] = \text{integer part of } n/2.$$

   Run your CGLS algorithm using the observation model

$$b = \mathsf{P}_k \mathsf{A} u_0 + \varepsilon,$$

   with $k = 2, 5, 10, 20$. In this case, the matrix is no longer symmetric, so the transpose needs to be computed in two phases: Transpose of $\mathsf{P}_k$ and transpose of $\mathsf{A}$.

   Run also one example in which you pick only very few observation points, e.g., randomly pick five indices out of the $n = 100$ observations in the case of complete data, and run the algorithm. Comment on the performance: Does the result indicate that the null space of the sparsely sampled forward matrix is a problem in this case?