MATH 150-03 / COMP 150-07 Homework #2

October 17, 2016

1 Overview

The homework is due by 11:59 PM on Friday, October 21st. All code is to be uploaded to the Tufts cluster. Find your directory in /cluster/tufts/train/math150. Make a directory called submit. Inside the submit directory make a directory called hw2. Please put all submission files there by the due date. Please scan your responses to the non-programming parts (or type them up), and put in the folder with the name hw2_written.pdf. For the last problem, you will use the sparse matrix functionality of the GNU GSL library: https://www.gnu.org/software/gsl/manual/html_node/Sparse-Matrices.html. Notice that sparse BLAS is not supported by the older version of GSL that is installed on the cluster. Please go ahead and install the newest (2.2) version of GSL locally using (./configure –prefix=install_dir ; make; make install). Notice that this must be done on a compute node of the cluster, this will not work on a login node. After this is done, you must set the necessary environmental variables prior to compilation of your code. See the setup_paths.sh script in the sample codes I posted and adjust the paths as necessary to your own installation. Notice that GSL is used only for the last part of the homework. Don't use GSL for part (C).

2 Assignment details

A. (5 pts) Suppose the following compressed column sparse format vectors are specified for a 5×4 sparse matrix:
i = [1, 3, 2, 3, 0, 2, 1, 3]
p = [0, 3, 4, 7, 8]
d = [1, 2.1, 3.1, 2.9, 3.1, 2.2, 4.6, 3.5]
Write down the matrix which corresponds to this sparse representation.

B. (10 pts) Given the following matrices:

$$A = \begin{bmatrix} 0.00 & 0.00 & 3.10 & 4.60 \\ 15.00 & 0.00 & 9.40 & 5.20 \\ 0.00 & 0.00 & 0.00 & 0.00 \\ 2.10 & 2.90 & 0.00 & 9.70 \\ 4.10 & 0.00 & 0.00 & 12.50 \end{bmatrix} \quad , \quad B = \begin{bmatrix} 0.00 & 0.00 & 2.10 \\ 13.00 & 9.20 & 0.00 \\ 0.00 & 0.00 & 0.10 \\ 2.10 & 0.00 & 0.00 \\ 4.10 & 0.00 & 0.00 \\ 16.10 & 0.00 & 15.20 \end{bmatrix}$$

State how to store these matrices in coordinate sparse format (e.g. Matlab sparse format (i,j,nnz)), compressed column formats, and compressed row formats. How would you store each format in a binary file so that you can reconstruct the matrix out of the contents of the file? Compare the cost of each format in bytes. See the GSL page for sparse matrices for format details.

C. (25 pts) Suppose we use the compressed row sparse format for storing sparse matrices.

Write a program in C or C++ which compiles to an executable called sparsematmult and reads in a text file of the following comma separated format in one line:

data/Acsr.bin, data/Bcsr.bin, data/v1.bin, data/v2.bin, data/outfile.txt

The matrices Acrs.bin and Bcrs.bin are stored in compressed row sparse format. The vectors data/v1.bin and data/v2.bin are vectors stored in dense format.

Compressed row format is specified as follows in a binary file (examples to be given in class)
m (int)
n (int)
numnnz (int)
i array (numnnz ints)
p array (m+1 ints)
d array (doubles)

Notice that for part (E), the above is not identical to the GSL format, it is something that I chose to use (though it's reasonably close). Write routines to compute the matrix vector product and the transpose matrix vector product with a sparse matrix, stored in compressed row sparse format. Your program should compute the following products:

$$A * v1, B * v2, A^T * v2, B^T * v1$$

Write the results of these computations in human readable (reasonably formatted) text format in the file outfile (as specified in the input file).

Your code should take advantage of the sparsity of the matrices (do not convert to a dense matrix and code the multiplications exactly as in the dense case). You may assume that the inputs supplied are of the right dimensions for the above operations to be possible. Do not use GSL for this part. Your code should not use any external libraries other than the standard C or C++ libraries.

D. (10 pts) Derive the solution to the following optimization problem (commonly referred to as Tikhonov or ℓ_2 regularization):

$$\bar{x} = \arg\min_x \left\{ \|Ax - b\|^2 + \lambda \|x\|^2 \right\}$$

Hint: set $A = USV^*$ (assume the use of the full SVD and proceed as done in class). Notice that the effect of the regularization is to filter the singular values of A. In this question, you will explore the effect of this filter. Suppose in Matlab or Octave, we have a set of singular values for a matrix specified using the command logspace(0,-2,100) and logspace(0,-8,100). Plot the singular values and the filtered singular values in the solution $\bar{x}$ (Tikhonov regularization) for several values of lambda (choose at least 3 for each case). Describe what you observe the effect of Tikhonov regularization to be.

E. (50 pts) Implement Tikhonov regularization with smoothing in C for sparse matrices using the GNU GSL library. This corresponds to the optimization problem:

$$\bar{x} = \arg\min_x \left\{ \|Ax - b\|^2 + \lambda \|x\|^2 + \lambda_2 \|Lx\|^2 \right\}$$

where L is some kind of gradient or Laplacian operator. Derive the linear system which solves for $\bar{x}$. Notice that the gradient for the above functional is well defined.

You should make use of the CG routine you wrote for the previous homework, but you will now make use of the sparse BLAS support in GSL for more efficient matrix-vector operations with sparse matrices. The matrices A and L will be stored in compressed row sparse format, in the

same types of binary matrices as for part (C). Your program should compile into an executable called runtikhonov and should take as command like argument a name of a text file with options, as before. The file will have on the first line (comma separated, as previously):
data/A.bin, data/L.bin, data/b.bin, data/x0.bin, lambdamax, lambdamin, num_lambdas, lambda2, maxiters, data/x.bin, data/residuals.txt, data/solutionnorms.txt
The matrices A and L will be sparse matrices written in compressed sparse row format in binary files (see example codes for reading and writing these). The vector b will be the right hand side vector in binary format and x_0 will be the initial guess vector in binary format. The program would run the *continuation scheme* for Tikhonov regularization we have discussed, using the CG algorithm for the linear system $(A^TA + \lambda I + \lambda_2 L^TL)x = A^Tb$ with varying parameter λ from value lambdamax to lambdamin (the parameter λ_2 is kept fixed):

```
rhs = A'*b;
I = eye(n,n);
for i=1:num_lambdas
    lambda = lambdas(i);
    fprintf('USING lambda = %f\n', lambda);
    if i>1
        x0 = x_sol;
    end

    M = AtA + lambda*I + lambda2*LtL;
    [x_sol] = cg(M, rhs, x0, TOL, maxiters);

    residuals(i) = norm(A*x_sol - b);
end
```

Adjust the parameter λ as done in the file run_tikhonov2.m in the sample scripts for week1.

Notice that A and L are sparse matrices. Write your code in a way that exploits sparsity. Do not explicitly form A^TA or L^TL, to preserve sparsity. The result of the Tikhonov regularization is written to a binary file specified in the input, data/x.bin above. In addition compute the residuals for each lambda value and supply them in the text output file data/residuals.txt, as above. Also compute the solution norms (regular Euclidean norm) at each lambda and supply them in data/-solutionnorms.txt. Pick a reasonable format to display the residual data. Try to make use of the GSL routines as much as possible. Notice again, that you are using here the GSL sparse BLAS routines for matrix-matrix and/or matrix-vector multiplications; although you are using your own CG implementation from HW1 and modifying it for use with sparse matrices. It may be helpful to implement the code in Matlab first before you implement it in C to see the simplifications you can make.

Based on the sample input I provide, make a few plots of the residual norm and solution norm versus lambda value, as well as the final solution itself for different smoothing regularization paramaters λ_2.