MATH 150-03 / COMP 150-07 Homework #4

December 12, 2016

1 Overview

The homework is due by 11:59 PM on Thursday, December 22nd. 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 hw4. 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 hw4_written.pdf. Please supply clear instructions on how to compile your codes and any necessary compile scripts and please include sample input files with which your code works. Please put any figures with labels you produce in your pdf.
For this assignment, we will consider the problem of non-linear least squares fitting. This is a challenging problem with respect to computation and is well suited for parallel architectures. You will implement the solution in Matlab and on multi-core and GPU systems.
Let $F(x,t) = x_1 \exp\left(-\frac{(t-x_2)^2}{(2x_3^2)}\right) + x_4$ be the non-linear fitting model (only 4 parameters). Given a set of points (t_i, y_i), the goal is do determine the constants x_1, x_2, x_3, x_4 which best fit the data in the least squares sense, by iteratively minimizing the non-linear function $g(x) = \frac{1}{2}\|r(x)\|^2$:

$$\bar{x} = \arg\min_x \left\{ \frac{1}{2}\|r(x)\|^2 \right\} \tag{1.1}$$

where $r_i(x) = y_i - F(x, t_i)$ and the standard Euclidean norm is used.

2 Assignment details

A. (10 pts) Compute the Jacobian and Hessian for the above model.

B. (5 pts) Describe how you would implement the iterative Gauss-Newton scheme for solving the minimization problem (1.1) (with the simplification for the Hessian).

C. (15 pts) Implement in Matlab the Gauss-Newton method for this problem. Your program will be called matlabgn.m . It will take two arguments (two text files) . The first text file will be called data.txt . It will contain the number of data points and the data to fit in the following format (on three lines):
N
t_1, t_2, . . . , t_N
y_1, y_2, . . . , y_N

The second file (initdata.txt) will contain the number of iterations to use and the initial guess for values x_1, x_2, x_3, x_4:
maxiter

x_1, x_2, x_3, x_4

Your program will read in this information, run the Gauss-Newton scheme and then return the four values x_1, x_2, x_3, x_4 which have been computed. You would also return the value of $g(x) = \frac{1}{2}\|r(x)\|^2$ evaluated using the model with the parameters you obtain. You can print these to stdout in human readable format. Use a pivoted LU factorization for the linear solve step and step size search as done in the sample Matlab scripts we have discussed.

Make a set of sample points (ex, pick a parameter vector x and some values of t and generate points using $F(x,t)$, possibly with some added noise). Run your code and make a plot of the data points and the model fit. Please include this plot in your submission. Notice that your algorithm is likely to diverge if your initial guess is not close to a vector x giving a relatively small residual value. The step size optimization and use of pivoting in the factorization for the system solution help improve performance, but the algorithm would need to be more complicated to perform better for this nonlinear problem. If you are interested in this topic, look up trust region methods or speak with me for more information.

D. (40 pts) Write a C code using Intel MKL library to solve the same problem as above, based on your Matlab code. Notice that you will need to figure out how to do a system solve in MKL at each iteration. Please use the pivoted LU factorization to do this. Also use a step size line search algorithm to select a parameter α. See the example Matlab code provided which implements everything for a simpler example. State and explain in your report all the steps of your scheme. For reference, see e.g. https://en.wikipedia.org/wiki/Gauss%E2%80%93Newton_algorithm#Improved_versions and the Numerical Linear Algebra textbook.

Your program should compile to an executable called *fitwithmkl* and take two command line arguments (the path to the data file data.txt and the path to the initial condition file initdata.txt). Where possible please use simple openMP constructs (e.g. for residual, Jacobian evaluations).

Your program will return the same information as the Matlab code.

E. (40 pts) Write a program in C to solve this problem on the GPU. Make use of the CUDA installation and the latest CUSP library which is installed in /cluster/tufts/train/math150/svoron01/software. Notice that the CUDA module installed on the cluster appears to have some issues. Please use instead the CUDA 7.5 library which I have installed above. To use these programs on a GPU node (e.g. salloc -N1 -c8 -t 150 –mem 20G -p gpu), you will require the following setup_paths.sh script:

```
#!/bin/bash
export PATH=/cluster/tufts/train/math150/svoron01/software/cuda-7.5/bin:$PATH
export LD_LIBRARY_PATH=/cluster/tufts/train/math150/svoron01/software/cuda-7.5/lib64:
$LD_LIBRARY_PATH
```

which is given in that directory. When I test your code, I will try to run it against my cuda installation first. If that fails, I will test against the system cuda install.

Your program will compile to the executable *fitwithcuda* and will take the same two command line arguments as above. Where possible please use simple GPU kernels (e.g. for residual, Jacobian evaluations). In particular, this means that you should make use of parallel function evaluations. Feel free to make use of the example programs provided. Unlike the MKL code, no direct solves (i.e. LU factorizations with pivoting) is necessary here. Please use instead one of the iterative algorithms for obtaining a solution to a positive semidefinite system at each iteration and do implement a line search strategy as before. Your program will return the same information as the Matlab code.

Information on the CUSP library and it's available functions is available at http://cusplibrary.github.io/. There is a number of examples for different functionality in /cluster/tufts/train/math150/svoron01/software/cusp_lib/cusplibrary-0.5.1 which you can compile with the nvcc command. We will go over examples of this library in the class. Notice that for

your C codes, with the same input parameters, the returned results should be very similar to those returned by your Matlab code.

How does the runtime of your different codes compare for your sample inputs? Please describe in your pdf.

F. (10 pts) Review of gradients and Jacobians. Let $k(x) = \|Ax-b\|_2^4$ for a matrix A which is $m \times n$ and vector x which is $n \times 1$. Compute the gradient of $k(x)$. What are the dimensions of the gradient? Next, suppose M is a symmetric $n \times n$ matrix and let $h(x) = (x^T Mx)v + x$ where x is the same vector as above and v is an $n \times 1$ vector of all ones. What is the Jacobian of $h(x)$? What are its dimensions? Please justify your calculations.