Math 425 Applied & Comput. Lin. Algebra Fall 2024 Handout

Mini-Projects

This document provides the information about the mini-projects including the list and description of the topics. Here are the guidelines to keep in mind.

(1) The projects will be completed **individually**. You need to let me know by *Friday, November 15* a ranked list of four topics chosen from among the topics below. You will record your ranked choices in Google spreadsheet that I will share. I will then determine the assignments.
(2) Each project involves learning something novel from written sources such such as textbooks. I provided references. Your responsibility is to understand the details of the project by reading these sources and discussing them with me if anything needs clarification.
(3) Each project also involves some amount of coding. When you are turning in your work, make sure you turn in your code. You need to also provide your code electronically (as MATLAB files) so that I can run them if I wish to check something. Similarly, if you have to run experiments or simulations, you need to show your computations. Tables, graphs, plots, etc. are welcome when reporting the results of your computations.
(4) You need to submit a short report that is four to five pages long, based on the reference(s). The report must be typed (LaTeX preferred) and it must include an introduction to the topic you are covering. It should give information about the mathematical background and the details of the implementation. The report must be written in your own words.
(5) The reports are due *Saturday, December 21.* This is the Saturday at the end of the finals week.

Project Descriptions

• *Implementing the Revised Simplex Algorithm (Optimization).* The simplex algorithm is a fast and reliable method of solving linear optimization problems. It is based on the geometry of the feasible solutions set of the optimization problem

$$\text{minimize } c^T \cdot x \text{ subject to } Ax = b \quad x \geq 0$$

where you can assume that A is an $m \times n$ matrix of rank $m \leq n$, $b \in \mathbb{R}^m$, and $c \in \mathbb{R}^n$. Everything you need can be found in *Introduction to Linear Optimization* by Bertisimas and Tsitsiklis (Chapter 3). I am expecting from you to learn the basic algorithm (Section 3.2), why it works (see optimality conditions in Section 3.1), and implement the *revised simplex method* (Section 3.3) in MATLAB.

• *Implementing an Interior Point Algorithm (Optimization).* Another class of algorithms that is used to solve linear (and other convex) optimization problems is known as interior point algorithms. I would like you to learn the sequential logarithmic barrier method. This method is based on a general method called the damped Newton method. The main reference for your project is *Optimization Models* by Calafiore and El Ghaoui. The details of the Newton algorithms can be found in Sections 12.2.4-12.2.6, and the logarithmic barrier method is in Section 12.3. I am expecting from you to learn why this algorithm works and implement it in MATLAB.

• *Implementing a Simple 2D Homography Algorithm (Computer Vision).* The goal of this project is to compute a projective transformation that transforms a given image to another one. This is known as 2D homograhy. The typical situation is the following: two images of the same scene are produced from two different camera positions. One determines points $X_1, \ldots, X_m$ in the first image and corresponding points $X'_1, \ldots, X'_m$ in the second image, and the goal is to find a transformation H such that $HX_i = X'_i$ for $i = 1, \ldots, m$. The main reference for this project is *Multiple View Geometry in Computer Vision* by Hartley and Zisserman. The SFSU library has online copies of this source. You need a library PIN to access it – please visit the library webpage for details. In Section 4.1 the Direct Linear Transformation Algortithm are the details of this algorithm spelled out. You will also have to read parts of Section 2 Projective Geometry and Transformation of 2D to develop the background for understanding this algorithm. You need to write an implementation of this algorithm in MATLAB.

• *Computation of the Camera Matrix (Computer Vision).* The goal of this project is to compute the transformation P that takes given 3D world points $X_1, \ldots, X_m$ to given 2D image points $x_1, \ldots, x_m$ such that $PX_i = x_i$. The basic algorithm to accomplish this task is Algorithm 7.1 in *Multiple View Geometry in Computer Vision* by Hartley and Zisserman. The SFSU library has online copies of this source. You need a library PIN to access it – please visit the library webpage for details. Chapter 6 on Camera Models will provide some of the necessary background. You need to write an implementation of this algorithm in MATLAB.

• *Computation of the Fundamental Matrix (Computer Vision).* In this project one is given points $x_1, \ldots, x_m$ in one image and points $x'_1, \ldots, x'_m$ in another image of the same scene. The goal is to reconstruct *both* cameras P and P' that gave rise to these images and the world points $X_1, \ldots, X_m$ from which the image points came from. An important task on the way to reach this goal is the computation of the fundamental matrix F. A basic algorithm for this task is Algorithm 11.1 in *Multiple View Geometry in Computer Vision* by Hartley and Zisserman. The SFSU library has online copies of this source. You need a library PIN to access it – please visit the library webpage for details. Both Chapter 9 and 10 of the same book will provide the necessary background for this algorithm which you need to implement in MATLAB.

• *Principal Component Analysis (Statistics).* Principal Component Analysis (PCA) is a technique of unsupervised learning widely used to discover the most informative directions in a data set which are directions along which the data varies the most. The goal of this project is to learn how this can be accomplished using basic numerical linear algebra. A starting reference for this project is *Optimization Models* by Calafiore and El Ghaoui where subsection 5.3.2 gives a good introduction. You can obtain the book using LINK+ through the SFSU library. Alternatively, you can ask me to give you a copy. Your main goal is to implement a simple SVD-based PCA algorithm in MATLAB and use it on data similar to the one given in Example 5.3 in the source book.

• *Procrustean Transformation Problems (Robotics, Computer Vision, Protein Analysis).* One is given points $X_1, \ldots, X_m$ and corresponding points $X'_1, \ldots, X'_m$ in 3D. The task is to find a rotation and translation that will transform the first set of points to the second set of points (approximately). The problem can be posed as a simple optimization problem that can be solved by an SVD procedure. Your task is to understand this formulation and implement the procedure in MATLAB. You will also run your algorithm

on simulated input.

• *Splines (Computer Graphics).* Splines are piecewise cubic functions used to represent curves, in particular, in computer graphics. The graph of the curve passes through given points $(x_0, y_0), \ldots, (x_n, y_n)$, and the goal is to find successive cubic functions $u_1(x), \ldots, u_n(x)$ which are defined between consecutive sample points x_j and x_{j+1}. These cubic functions must be also smooth, so there are conditions coming from the derivatives of these cubic polynomials $u_j(x)$. Computing the cubic functions $u_j(x)$ boils down to computing the coefficients in these polynomials using a system of linear equations. These systems are tridiagonal and there are special ways of solving them. Your task is to read relevant section (Section 5.5) in our own textbook *Applied Linear Algebra* and understand how these cubic splines are constructed. I expect you to implement the main reconstruction algorithm in MATLAB. You need to try your algorithm on constructing fonts. For this check out Figure 5.15 and Exercise 5.5.73 in *Applied Linear Algebra*.

• *Tensor Decompositions (from Linear Algebra to Multilinear Algebra).* Tensors are multidimensional analogs of matrices. In today's world, data *are* tensors. In this more open-ended project, your task is to explore various tensor decomposition algorithms. Tensor decompositions are to tensors what SVD is to matrices, though there is no one notion of a tensor decomposition. In your report, you need to explain why your chosen tensor decomposition algorithm is the analog of the SVD in the case of matrices. Start with the following link

`http://www.commsp.ee.ic.ac.uk/~mandic/SPM-Cichocki-Mandic---DeLathauwer.pdf`

I am expecting from you to implement at least one tensor decomposition algorithm in MATLAB.