Homework 1 - CSE 276C - Math for Robotics

Due: Friday, 12 October 2020

1. Implement the $PA = LDU$ decomposition algorithm by yourself (i.e. do not just call a built-in function in Matlab or Python. You may assume the matrix A is square and of full rank. Show that your implementation is functional.

2. Compute the $PA = LDU$ decomposition and the SVD decomposition for each of the following matrices:

   (you can use your own LDU implementation and it is OK to use a pre-defined implementation for SVD).

   a.
$$A_1 = \begin{bmatrix} 4 & 7 & 0 \\ 3 & 2 & 1 \\ 2 & 2 & -6 \end{bmatrix}$$

   b.
$$A_2 = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

   c.
$$A_3 = \begin{bmatrix} 2 & 2 & 5 \\ 3 & 2 & 5 \\ 1 & 1 & 5 \end{bmatrix}$$

3. Solve the following system of equations $Ax = b$ given the below values for A and b. For each system specify if it has zero, one or more solutions. For the systems with zero solutions give the SVD solution. Relate your answers to the SVD decomposition.

   a.
$$A = \begin{bmatrix} 2 & 1 & 3 \\ 2 & 1 & 2 \\ 5 & 5 & 5 \end{bmatrix} \quad b = \begin{bmatrix} 10 \\ -10 \\ 0 \end{bmatrix}$$

   b.

$$A = \begin{bmatrix} 8 & 14 & 0 \\ 2 & 2 & -6 \\ 1 & 2 & 1 \end{bmatrix} \quad b = \begin{bmatrix} 6 \\ 5 \\ 1 \end{bmatrix}$$

c.

$$A = \begin{bmatrix} 4 & 7 & 0 \\ 2 & 2 & -6 \\ 1 & 2 & 1 \end{bmatrix} \quad b = \begin{bmatrix} 18 \\ -12 \\ 8 \end{bmatrix}$$

4. Determine the Nullspace of the following matrices:

   a.

$$A_1 = \begin{bmatrix} 1 & 2 & 0 \\ -1 & 1 & 6 \end{bmatrix}$$

   b.

$$A_1 = \begin{bmatrix} 2 & 1 \\ 3 & 2 \end{bmatrix}$$

5. Consider the linear system of equations $Ax = b$:

$$A = \begin{bmatrix} 1 & 1 & 3 \\ 1 & 2 & 4 \\ 1 & 3 & x \end{bmatrix} \quad b = \begin{bmatrix} 2 \\ 3 \\ y \end{bmatrix}$$

   a. For what values of x will the system have a unique solution?

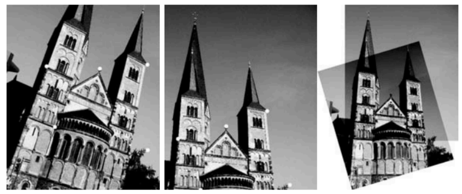
   b. For what values of x and y will the system have no solution?

   c. For what values of x and y will the system have infinitely many solutions?

   Give reasons for your answers and show your work.

6. In generating a mosaic / panorama from a set of images a frequent problem is matching images. When the camera makes a small motion a reasonable assumption is that there exist some rotation matrix $\mathbf{R}$ (2x2) and translation vector $\vec{t}$ (2x1) such that points in one image x_i match to point y_i in the other image, i.e.:

$$y_i = \mathbf{R}x_i + \vec{t}$$



(b) Input images with keypoints (c) Aligned images

The equation system can be unrolled to use standard software tools. We can combine our unknowns into a vector $\vec{u}$ as

$$\vec{u} = \begin{bmatrix} r_{11} \\ r_{12} \\ r_{21} \\ r_{22} \\ t_1 \\ t_2 \end{bmatrix}$$

Write a matrix $\mathbf{M}$ (6x6) with a vector $\vec{d}$ so that we can solve the systems $\mathbf{M}\vec{u} = \vec{d}$. It is perfectly OK to so use temporary variable such as $\vec{x}_{sum}$, $\vec{y}_{sum}$, $\mathbf{X}$.

7. Aligning point clouds.

   The file bunnies.npy contains two point clouds $array\_p\_Om \in R^{3\times 1000}$ and $array\_p\_s \in R^{3\times 1000}$. Each point cloud thus contains 1000 points. Find a rotation R^* and translation p^* that best aligns the two point clouds. That is, solve the following optimization problem for p and R:

$$\min_{p\in R^3, R\in SO(3)} \sum_{i=0}^{999} \left\| p + R^O p_i^m - p_i^s \right\|^2 \tag{1}$$

   where following numpy syntax:

$$\begin{aligned} {}^O p_i^m &= array\_p\_Om[:, i] \\ p_i^s &= array\_p\_s[:, i] \end{aligned}$$

   The python notebook solve.ipynb provides utilities to load the point clouds and visualize them. Please submit your code and also visualizations of the aligned point clouds.

For references material, we encourage you to take a look at classic papers on this topic, such as section III in `https://ieeexplore.ieee.org/document/44063` and `https://arxiv.org/abs/0904.1613`.

Acknowledgement: the point clouds were obtained from the mesh of the Stanford Bunny (`http://graphics.stanford.edu/data/3Dscanrep/`)