CS 2300-001

Programming Assignment 4

Pam Russel

Student: Moses Tulepkaliev

Preface

This document goes over the general process and math used in PA4, the code additioanlly contains comments cooresponding to all the major steps.

Part 1

Matrix Input

Read in the kx9 input matrix as $\begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} & a_{0,4} & a_{0,5} & a_{0,6} & a_{0,7} & a_{0,8} \\ ... & ... & ... & ... & ... & ... & ... & ... & ... \\ a_{k-1,0} & a_{k-1,1} & a_{k-1,2} & a_{k-1,3} & a_{k-1,4} & a_{k-1,5} & a_{k-1,6} & a_{k-1,7} & a_{k-1,8} \end{bmatrix}$

Extract the plane point $\vec{q} = \begin{bmatrix} a_{0,0} \\ a_{0,1} \\ a_{0,2} \end{bmatrix}$, the plane normal vector $\vec{n} = \begin{bmatrix} a_{0,3} \\ a_{0,4} \\ a_{0,5} \end{bmatrix}$, and the projection direction $\vec{v} = \begin{bmatrix} a_{0,6} \\ a_{0,7} \\ a_{0,8} \end{bmatrix}$

Remove row 0 for the input matrix

For each remaining row in the matrix extract 3 points and store as x

Paralell Projection

We solve for the paralell projected point x' by using the equation $x' = Ax + p$ where $A = I^3 - \frac{\vec{v}\vec{n}^T}{\vec{v}\cdot\vec{n}}$ and $p = \frac{\vec{q}\cdot\vec{n}}{\vec{v}\cdot\vec{n}}$, using matrix operations from PA 1-3.

Persepctive projection

We solve for a point's persepctive projection by defining the projection direction as the line from the point to the origin. We then simple solve for a paralell projection with the given projection direction $\vec{v} = -x$.

Part 2

Matrix Input

Read in the kx9 input matrix as $\begin{bmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} & a_{0,4} & a_{0,5} & a_{0,6} & a_{0,7} & a_{0,8} \\ ... & ... & ... & ... & ... & ... & ... & ... & ... \\ a_{k-1,0} & a_{k-1,1} & a_{k-1,2} & a_{k-1,3} & a_{k-1,4} & a_{k-1,5} & a_{k-1,6} & a_{k-1,7} & a_{k-1,8} \end{bmatrix}$

Subpart 1

Process each line j individually.
Extract the plane point $\vec{x} = \begin{bmatrix} a_{j,0} \\ a_{j,1} \\ a_{j,2} \end{bmatrix}$, the plane normal vector $\vec{n} = \begin{bmatrix} a_{j,3} \\ a_{j,4} \\ a_{j,5} \end{bmatrix}$, and the point to find the distance to $p = \begin{bmatrix} a_{j,6} \\ a_{j,7} \\ a_{j,8} \end{bmatrix}$
Normalize $\vec{n}$
Define the implicit form of the plane by finding the coefficient $c = -(n_0x_0 + n_1x_1 + n_2x_2)$
Use the equation from Chapter 11 where $||p - q|| = t = c + n \cdot p$ therefore the distance is t.
Return the absolute value of t because distance is positive.

Subpart 2

Extract the two line points $q_1 = \begin{bmatrix} a_{j,0} \\ a_{j,1} \\ a_{j,2} \end{bmatrix}$ and $q_2 = \begin{bmatrix} a_{j,3} \\ a_{j,4} \\ a_{j,5} \end{bmatrix}$.
We can define the line vector $\vec{v} = q_2 - q_1$
Remove the first row from the input.
For each remainig row j of input extract the p3 triangle vertices $p_1 = \begin{bmatrix} a_{j,0} \\ a_{j,1} \\ a_{j,2} \end{bmatrix}$, $p_2 = \begin{bmatrix} a_{j,3} \\ a_{j,4} \\ a_{j,5} \end{bmatrix}$, and $p_3 = \begin{bmatrix} a_{j,6} \\ a_{j,7} \\ a_{j,8} \end{bmatrix}$
From the Chapter 11 notes we have the equation for the intersection of a line and a triangle as $p + t\vec{v} = p_1 + u_1(p_2 - p_1) + u_2(p_3 - p_1)$
We can rearrange this as $p - p_1 = u_1(p_2 - p_1) + u_2(p_3 - p_1) - t\vec{v}$
We can set up a 3x3 matrix system of equations as $Ax = b$ where $A = \begin{bmatrix} p_2 - p_1 & p_3 - p_1 & -v \end{bmatrix}$ $x = \begin{bmatrix} u_1 \\ u_2 \\ t \end{bmatrix}$ and $b = [p - p_1]$
We can solve for x using the gaussian elminiation alogrithm provided in Chapter 12 of the notes (page 7).
We then check u_1 and u_2 to check if the line intersects the triangle, that is $0 < u_1, u_2 < 1$ and

$u_1 + u_2 \leq 1$ We throw an error that the triangle does not intersect the line if the above condition is not met.
If the above condition is met, we know that it intersects so we calculate the itersection line by using t in the line equation $t\vec{v} + q_1$ to get our intersection point.

Part 3

Matrix Input

Read in the nxn stochastic input matrix as A=$\begin{bmatrix} a_{0,0} & a_{0,1} & ... & a_{0,n-1} \\ ... & ... & ... & ... \\ a_{n-1,0} & a_{n-1,1} & ... & a_{n-1,n-1} \end{bmatrix}$

Input Validation

Iterate through all the elements of the stochastic matrix and check to make sure each element is positive
Iterate through each column individually and add all the elements of each column to make sure they sum to one.

Starting eigenvector

To use the power method we need to provide a strating eigenvector, the closer to the actual eigenvector it is, the less iterations are needed.

To get a good starting approximation, I chose to make each element of the vector the sum of that row's cooresponding columns, as this represents the inlinks, initaliiy defining rank by number of inlinks seems like a good starting point.

Power Method

I implemented the algorithm for the power method that is listed on page 7 of the chapter 15 notes, with a .0001 tolerance and a max of 1000 iterations.

Page Ranking

The output eigenvector is the ranking of each of the pages, the higher the element in the eigenvector, the higher the page ranking. I then sort the indexes of the by the values in the eigenvector.

Notes

At some points you will see what is supposed to be a vector or point represented as an object of the Matrix class, this is due to vector-matrix math not being implemented yet so to do vector-matrix math I convert objects of type Vec to 3x1 Matrix objects.
The page rankings for the pages are listed using 0 indexed pages, that is, if a 10x10 matrix was passed in, the pages would be labeled pages 0-9.
Double equality was implemented by checking within a five decimal point precision.

References

Lecture notes Ch 10-12, Ch 15