Sarah Kirby

Module 3 Assignment

Program output:

Adjacency Matrix

Eigen Min: -5.411264653989054 Max: 10.554587092710952

Zero count: 10

Laplacian Matrix

Eigen Min: -5.974826820616312E-14 Max: 22.541310152105993

Zero count: 11

The Adjacency matrix has eigen values that range from ~ -5.4 to ~10.6. There are 10 eigen values that are 0.

The Laplacian matrix has eigen values ranging from 0 to ~22.5. (The minimum eigen value doesn't appear as 0 because the program has rounding errors.) The Laplacian matrix has 11 eigen values that are 0.

Program (also included as Module3.java):

package assignments;  
  
import graph.GraphUtils;  
import graph.MGraph;  
import org.apache.commons.math3.linear.EigenDecomposition;  
import org.apache.commons.math3.linear.SparseRealMatrix;  
import org.apache.tinkerpop.gremlin.tinkergraph.structure.TinkerGraph;  
  
public class Module3 {  
  
 private final String GRAPH\_FILE = "GraphDatabases\\students.graphml";  
  
 private final double zero = 0.000001;  
  
 private TinkerGraph graph;  
  
 public Module3()  
 {  
 graph = createGraph();  
 getGraphMatrices(graph);  
 }  
  
 public TinkerGraph createGraph()  
 {  
 TinkerGraph graph = GraphUtils.*readGraphML*(GRAPH\_FILE);  
 return graph;  
 }  
  
 // This example demonstrates how to obtain 2 of the most commonly used matrices in graph analytics  
  
 public void getGraphMatrices(TinkerGraph g)  
 {  
 boolean directed = false;  
  
 MGraph graph = new MGraph(g, directed);  
  
 //Adjacency  
 SparseRealMatrix adjacency = graph.getAdjacency();  
 EigenDecomposition eigenDecompAdj = new EigenDecomposition(adjacency);  
 double[] eigenAdj = eigenDecompAdj.getRealEigenvalues();  
  
 System.*out*.println("Adjacency Matrix");  
 //printArray(eigenAdj);  
 System.*out*.println("Eigen Min: " + getMin(eigenAdj) + " Max: " + getMax(eigenAdj));  
 System.*out*.println("Zero count: " + countZeros(eigenAdj));  
  
 //Laplacian  
 System.*out*.println("\nLaplacian Matrix");  
 SparseRealMatrix laplacian = graph.getLaplacian();  
 EigenDecomposition eigenDecompLapl = new EigenDecomposition(laplacian);  
 double[] eigenLapl = eigenDecompLapl.getRealEigenvalues();  
  
 //printArray(eigenLapl);  
 System.*out*.println("Eigen Min: " + getMin(eigenLapl) + " Max: " + getMax(eigenLapl));  
 System.*out*.println("Zero count: " + countZeros(eigenLapl));  
 }  
  
 public void printArray(double[] arr){  
 System.*out*.print("Eigen values:");  
 for(int i = 0;i < arr.length;i++){  
 System.*out*.print(" " + arr[i]);  
 }  
 System.*out*.println("");  
 }  
  
 public double getMin(double[] arr){  
 double result = Double.*MAX\_VALUE*;  
 for(int i = 0;i < arr.length;i++){  
 if(arr[i]< result){  
 result = arr[i];  
 }  
 }  
 return result;  
 }  
  
 public double getMax(double[] arr){  
 double result = Double.*MIN\_VALUE*;  
 for(int i = 0;i < arr.length;i++){  
 if(arr[i] > result){  
 result = arr[i];  
 }  
 }  
 return result;  
 }  
  
 public int countZeros(double[] arr){  
 int result = 0;  
 for(int i = 0;i < arr.length;i++){  
 if(arr[i] > -1\*zero && arr[i] < zero){  
 //System.out.println("zero: " + arr[i]);  
 result++;  
 }  
 }  
 return result;  
 }  
  
 public static void main(String[] args)  
 {  
 new Module3();  
 }  
}