Abraham Lincoln's Eighth Judicial Circuit (1850) in the Age of MapQuest

Lawrence Hubert

Department of Psychology
The University of Illinois

APA Division 5 Presentation
Jacob Cohen Award for Distinguished Contributions to Teaching and Mentoring
August 8, 2009

So Why This Talk?

Because the Jacob Cohen Award is for teaching and mentoring, it seems only appropriate to provide an interesting example that we all might use in our graduate methodology classes.

Secondly, because of where I now live and teach (in Urbana, Illinois), this example for me is very locally salient (and useful for the various multidimensional scaling and clustering classes I regularly teach).

Maybe most importantly, we can do a little to honor the 200th anniversary of Abraham Lincoln's birth (on February 12, 1809).

Also, I think I have a cooler title than Howard Wainer.

The Eighth Judicial Circuit

Prior to becoming the 16^{th} President of the United States in 1860, Abraham Lincoln worked as a lawyer in the Eighth Judicial Circuit in Illinois.

The Eighth Circuit consisted of fourteen counties (and their county seats), with a total of some one hundred thousand people.

These county seats were visited consecutively in the Spring and Fall of each year, over a period of eleven weeks and with a travel distance of well over four hundred miles.

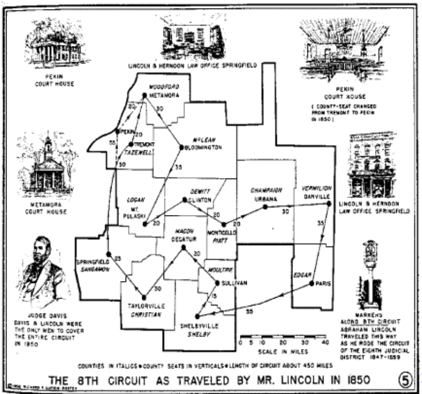
The counties were visited in the (circular) ordering listed on the next slide, starting from the state capital of Springfield. Because I live in Urbana, however, I will give that locale the number one label.

The County Seats of the Eighth Circuit

1: Urbana (Champaign)
2: Danville (Vermilion)
3: Paris (Edgar)
4: Shelbyville (Shelby)
5: Sullivan (Moultrie)
6: Decatur (Macon)
7: Taylorville (Christian)
8: Springfield (Sangamon)
9: Pekin (Tazewell)
10: Metamora (Woodford)
11: Bloomington (McLean)
12: Mt. Pulaski (Logan)
13: Clinton (Dewitt)
14: Monticello (Piatt)

For those who have lived in the Midwest, county names and where they are in relation to you, become very familiar from all the tornado warnings we receive. Thus, if a sighting occurs to the west in Piatt county, I'm on my way to the basement.

An Old Map of the Eighth Judicial Circuit



Lincoln's Mode of Transport on the Circuit

The Traveling Sales(man)person Problem (TSP)

Given a set of cities that a salesperson must visit, plus measures of effort to travel between cities (e.g., distance, time, cost, and so on), find an "optimal" route that starts at some city, visits each of the other cities, and returns to the city of origin.

"Optimality"" might be defined as the tour of least total effort, e.g., the minimal sum of travel times.

The related seriation task of defining an "optimal" path that visits each city once (but does not need to return to the starting city), is a special case: just define a "dummy city" for which there is no effort in going to or from it (in relation to any other city).

(And no, the TSP does *not* refer to the jokes told by boys in Junior High.)

The Ubiquity of the TSP

The TSP is the most studied combinatorial optimization problem in Operations Research (or elsewhere, I might add).

It has spawned an enormous literature on general solution techniques in combinatorial optimization.

It is the optimization "engine" for a huge number of creative applications.

Two pertinent references:

The Traveling Salesman Problem: A Computational Study
David Applegate; Robert Bixby; Vasek Chvatel; William Cook
Princeton University Press, 2007

Applications of Combinatorial Programming to Data Analysis: The Traveling Salesman and Related Problems
Lawrence Hubert; Frank Baker
Psychometrika, 1978, *43*, 81–91.

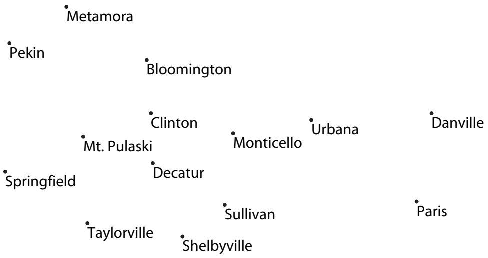
MapQuest Times for Travel on the Eighth Circuit

	1	2	3	4	5	6	7	8	9	10
1	00	35	76	93	72	62	97	94	103	100
2	35	00	57	119	98	87	123	119	129	125
3	76	57	00	87	71	94	136	135	166	162
4	93	119	87	00	30	49	49	86	142	145
5	72	98	71	30	00	41	59	78	135	135
6	62	87	94	49	41	00	42	44	97	98
7	97	123	136	49	59	42	00	41	103	118
8	94	119	135	86	78	44	41	00	76	91
9	103	129	166	142	135	97	103	76	00	38
10	100	125	162	145	135	98	118	91	38	00
11	59	85	122	104	94	57	93	72	50	48
12	76	102	122	76	68	32	48	37	65	81
13	56	82	114	78	68	31	66	60	71	70
14	33	59	83	68	46	35	71	67	100	98

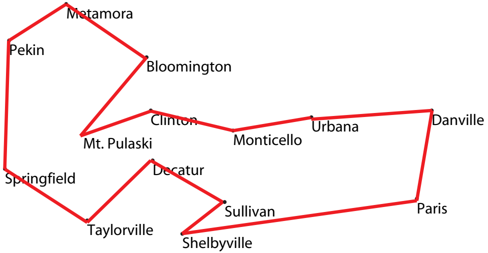
MapQuest Times for Travel on the Eighth Circuit

	11	12	13	14
1	59	76	56	33
2	85	102	82	59
3	122	122	114	83
4	104	76	78	68
5	94	68	68	46
6	57	32	31	35
7	93	48	66	71
8	72	37	60	67
9	50	65	71	100
10	48	81	70	98
11	00	54	31	60
12	54	00	27	50
13	31	27	00	36
14	60	50	36	00

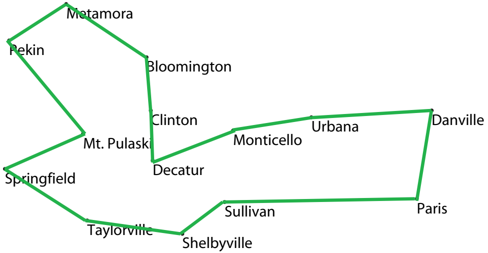
Placement of the Eighth Circuit County Seats



Lincoln's route on the Eighth Circuit



The Optimal Route on the Eighth Circuit



The Quality of Lincoln's Tour

The optimal tour based on total time is 601; Lincoln's total tour length was 645, so he could have saved 44 minutes (in today's MapQuest time) using the optimal tour.

Still, Lincoln's tour is pretty good, and does satisfy two properties that an optimal tour must have for a Euclidean embedded TSP.

(a) The cities in the convex hull are visited in order (for the idea of a convex hull, think about a rubber band pulled around pegs standing at each city location).

(b) The paths in Lincoln's tour do not cross.

The improvements made to Lincoln's tour to get to one that is optimal are suggested by the acute angles at Shelbyville and Mt. Pulaski.

Even though the MapQuest times of travel and the MapQuest distances are *not* linear (due to the possible use of freeways or slower county road travel), the exact same tour would be optimal and Lincoln's would not (but would still be pretty good).

The map representations given earlier were generated from the actual longitude and latitude values for the county seats (also, given by MapQuest); in other words, the MapQuest distances or times were not used to generate the maps.

Instructional Uses of the Lincoln MapQuest Example

There are at least four areas in Psychology that might use this example:

A) Direct application of the TSP (or optimal path) optimization task

cda.psych.uiuc.edu/lincoln_mapquest_presentation.pdf

B) Use of the example itself for multidimensional scaling and visualization

C) Comparisons between alternative proximity matrix representations

D) To give illustrations from the field of computational geometry

Six TSP (or Optimal Path) Applications

A) As a canonical experimental task in the problem solving literature:
Some tours are more equal than others: The convex-hull model revisited with lessons for testing models of the traveling salesperson problem
Susanne Tak, Marco Plaisier, and Iris van Rooij
The Journal of Problem Solving, 2008, *2*, 4–25 (Online journal)

B) As a (developmental) task in distinguishing perceptual versus cognitive skills:
Perceptual or analytical processing? Evidence from children's and adult's performance on the Euclidean traveling salesperson problem
Iris van Rooij, U. Stege, and A. Schactman
The Journal of Problem Solving, 2006, *1*, 44–73.

C) As part of the continuing discussion of what constitutes "figural goodness":

The generally direct relation between the quality of a tour and its perceived goodness-of-figure

The discussion of tour aesthetics with high quality geometric tours "feeling right" — particularly to graphic design professionals

D) As part of various neuropsychological assessment procedures:

The Trail Making Test from the Halstead-Reitan Battery

E) As a model for Data Array Clustering and helping interpret the patterning in an n object by p attribute data matrix:

$$\mathbf{X} = \{x_{ij}\}_{n \times p}$$

where x_{ij} is the value for object i on attribute j.

Define two separate proximity matrices between the objects and between the attributes:

$$\mathbf{R}_{row} = \{r_{ii'}\} = \{\sum_{j=1}^{p} x_{ij} x_{i'j}\}_{n \times n}$$

$$\mathbf{C}_{column} = \{c_{jj'}\} = \{\sum_{i=1}^{n} x_{ij} x_{ij'}\}_{p \times p}$$

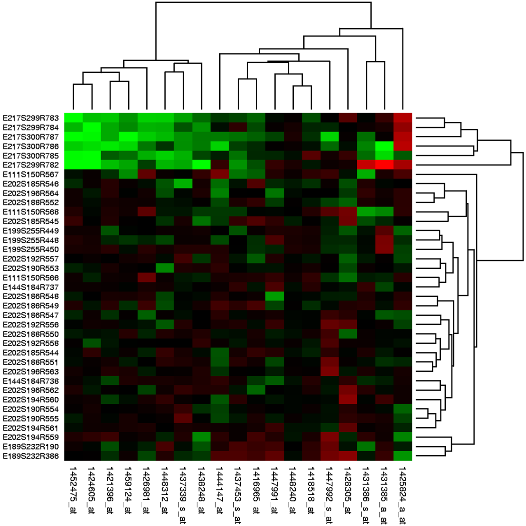
Reorder the rows (and separately the columns) to place the numerically larger elements of the array together by finding maximum weight paths using $\mathbf{R}_{row}$ and $\mathbf{C}_{column}$.

Could lead to prettier heat maps for enormously-sized matrices given that we now can solve the TSP optimally for tens of thousands of cities (can we all say, "data mining"):

A very nice reference for some of the historical background:

The history of the cluster heat map
Leland Wilkinson and Michael Friendly
The American Statistician, in press.

Also see:
The bond energy algorithm revisted
Phipps Arabie; Lawrence Hubert
IEEE Transactions on Systems, Man, and Cybernetics, *20*, 1990, 268–274.


E217S299R783
E217S299R784
E217S300R787
E217S300R786
E217S300R785
E217S299R782
E111S150R567
E202S185R546
E202S196R564
E202S188R552
E111S150R568
E202S185R545
E199S255R449
E199S255R448
E199S255R450
E202S192R557
E202S190R553
E111S150R566
E144S184R737
E202S186R548
E202S186R549
E202S186R547
E202S192R556
E202S188R550
E202S192R558
E202S185R544
E202S188R551
E202S196R563
E144S184R738
E202S196R562
E202S194R560
E202S190R554
E202S190R555
E202S194R561
E202S194R559
E189S232R190
E189S232R386
1452475_at
1424605_at
1421396_at
1459124_at
1426981_at
1448312_at
1437339_s_at
1438248_at
1444147_at
1437453_s_at
1416965_at
1447991_at
1448240_at
1418518_at
1447992_s_at
1428305_at
1431386_s_at
1431385_a_at
1425824_a_at

E) As a model for Unidimensional Unfolding, using an n subject by p object preference matrix:

$$\mathbf{P} = \{p_{ij}\}_{n \times p}$$

where p_{ij} is a measure of preference (or dissimilarity) that subject i has for object j (i.e., a low value implies preference for the object).

We look for a reordering of the rows of $\mathbf{P}$ so that within each column the entries generally decrease to a minimum and thereafter increase.

Similarly, we look for a reordering of the columns to identify the same type of gradient within each row.

As suggested by D.G. Kendall, define two separate proximity matrices between the objects and between the attributes:

$$\mathbf{R}_{row} = \{r_{ii'}\} = \{\sum_{j=1}^{p} \min(x_{ij}, x_{i'j})\}_{n \times n}$$

$$\mathbf{C}_{column} = \{c_{jj'}\} = \{\sum_{i=1}^{n} \min(x_{ij}, x_{ij'})\}_{p \times p}$$

And obtain the row and column reorderings by finding maximum weight paths within $\mathbf{R}_{row}$ and $\mathbf{C}_{\mathbf{column}}$

We are, in effect, looking for the Coombsian "parallelograms":

attributes:	a1	a2	a3	a4	a5	a6	a7	a8
subjects								
s1	2	3	5	8	12	14	17	19
s2	1	2	4	7	11	13	16	18
s3	2	1	1	4	8	10	13	15
s4	4	3	1	2	6	8	11	13
s5	5	4	2	1	5	7	10	12
s6	7	6	4	1	3	5	8	10
s7	8	7	5	2	2	4	7	9
s8	9	8	6	3	1	3	6	8
s9	11	10	8	5	1	1	4	6
s10	13	12	10	7	3	1	2	4
s11	14	13	11	8	4	2	1	3
s12	16	15	13	10	6	4	1	1

F) As a model for Profile Smoothing (or what is now known as Parallel Coordinate Analysis), and helping unclutter the plots of n subject profiles over a set of p attributes (the latter are listed along a horizontal axes).

Define a proximity matrix among the attributes as

$$\mathbf{C}_{column} = \{c_{jj'}\} = \{\sum_{i=1}^{n} I(x_{ij} > x_{i'j} \text{ and } x_{ij'} < x_{i'j'})$$

where $I(\cdot)$ is a zero/one indicator of a crossing of two profiles for subjects i and i'

A minimum weight path using $\mathbf{C}_{column}$ is a mechanism for minimizing the number of crossings over the set of profiles (a measure of visual clutter).

Some particularly relevant sources:

Hyperdimensional Data Analysis Using Parallel Coordinates
Edward Wegman
Journal of the American Statistical Association, 1990, *85*, 664-675

Chapter 1: Profiles, in John Hartigan, *Clustering Algorithms*, 1975, Wiley

Parallel Coordinates: Visual Multidimensional Geometry and Its Applications
Alfred Inselberg, 2009, Springer

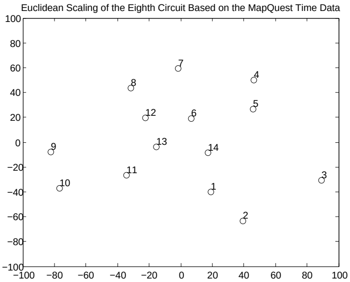
Applications of the MapQuest Time Data for Multidimensional Scaling and Visualization

Using MapQuest times to generate a spatial representation of the county seats, we can compare this scaling to the actual map based on longitude and latitude.

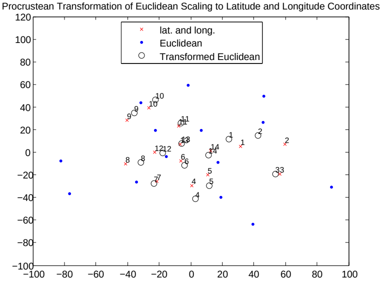
The comparison can be done through a Procrustes transformation of the coordinates from the scaling to the coordinates of the actual map (e.g., through a MATLAB M-function, procrustes.m).

We carried out the (metric) multidimensional scaling with the M-file, mdscale.m, from the Statistics Toolbox in MATLAB.

Euclidean Scaling Based on the Mapquest Times



Comparison to Longitude and Latitude

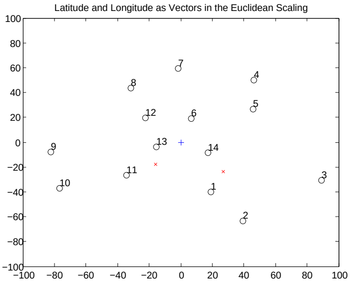


Property Fitting in a Multidimensional Scaling

Given the spatial representation from a scaling based on time, the two properties of longitude and latitude can be fit as vectors into the representation to indicate east-west and north-south directions.

These projections (defining two vectors in the scaling) can be seen as an illustration of Tucker's vector model, or alternatively, as one of Doug Carroll's external unfolding options in PREFMAP.

Vector Fitting in a Multidimensional Scaling



Scaling in the Euclidean or City-Block Metrics

Because times between locations do not always reflect straightline (i.e., Euclidean) distances, possibly better fits could be obtained using city-block metric scaling methods.

Also, there is a city-block bias of major highways going East/West and North/South, and in how states were platted in the 1800's.

At one time I lived at the intersection in Champaign County at 1800N and 600E (18 miles north of the county to the south, and 6 miles east of the county to the west).

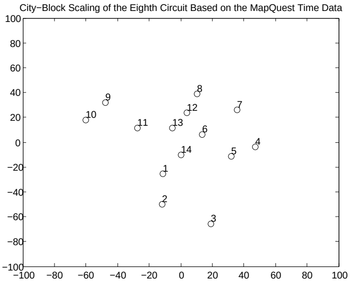
The city-block scaling of the MapQuest travel time data was done with the very robust M-file, biscalqa.m, documented in the monograph:

Structural representation of proximity matrices with MATLAB
Lawrence Hubert; Jacqueline Meulman; Phipps Arabie
SIAM: Philadelphia, 2006

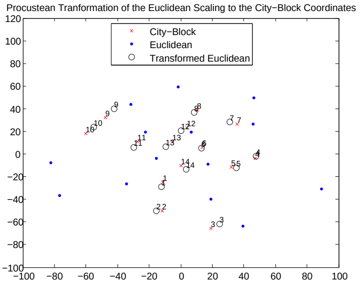
As the illustration to follow shows, it really makes little difference in this case, whether a scaling is done with the city-block or the Euclidean metric.

From my experiences, this latter observation is one that generalizes very widely.

City-Block Scaling of the MapQuest Time Data



Procrustean Transformation of Euclidean To City-Block Coordinates



Comparisons of Alternative Proximity Matrix Representations

We can have meaningful demonstrations of how various alternative methodologies look and compare on the same data set. For example, for the MapQuest time data:

A (best) least-squares fitted ultrametric has a (not so good) variance-accounted-for (vaf) of 49.32% (but using about half of the fitted "parameters" of the next three)

A (best) least-squares fitted additive tree has a vaf of 85.25%

A (best) least-squares city-block scaling has a vaf of 97.48%

A (best) least-squares Euclidean scaling has a vaf of 98.80%

A number of ways could be tried for imbedding an ultrametric or additive tree into one of the scalings.

Computational Geometry Illustrations

Various computationally important (and very cool) geometric structures can be shown nicely with the Eighth Circuit Map:

Voronoi diagram: a subdivision of a Euclidean space that shows the regions of points closest to each of the given points in the space.

Delaunay triangulation: an undirected graph among the given points where an edge exists between two points if a boundary is present in the Vornoi diagram separating the two points.

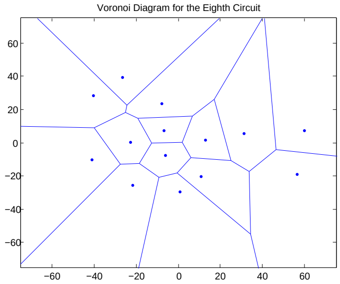
Nearest neighbor graph: an undirected graph among the given points when an edge is present between two points if one is a nearest neighbor of the other.

Minimum spanning tree: an undirected, connected, acyclic graph among the given points that has minimum weight for the sum of edge lengths.

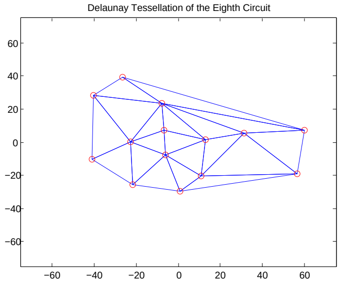
Gabriel graph: an undirected graph among the given points where an edge exists between two points if the disk having that edge as a line segment includes no other points from the given point set.

A Gabriel graph is a subgraph of the Delaunay triangulation and contains the minimum spanning tree and the nearest neighbor as subgraphs.

Voronoi Diagram for the Eighth Circuit



Delaunay Tesselation for the Eighth Circuit



Minimum Spanning Tree for the Eighth Circuit

Minumum Spanning Tree for the Eighth Circuit