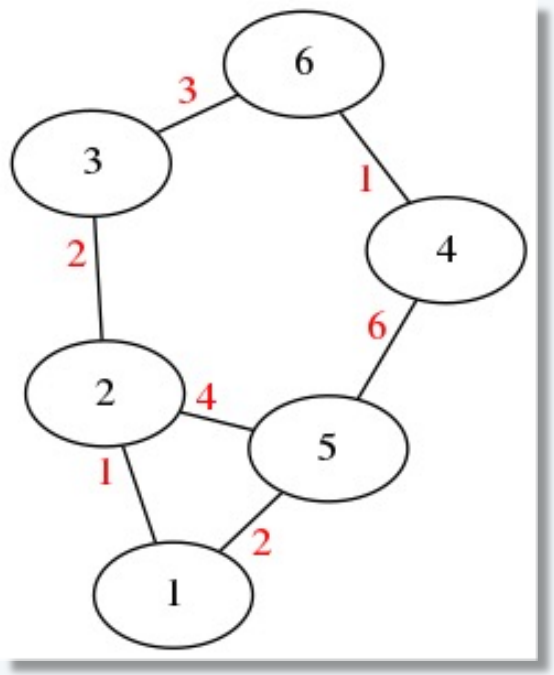
DM587 – Scientific Programming

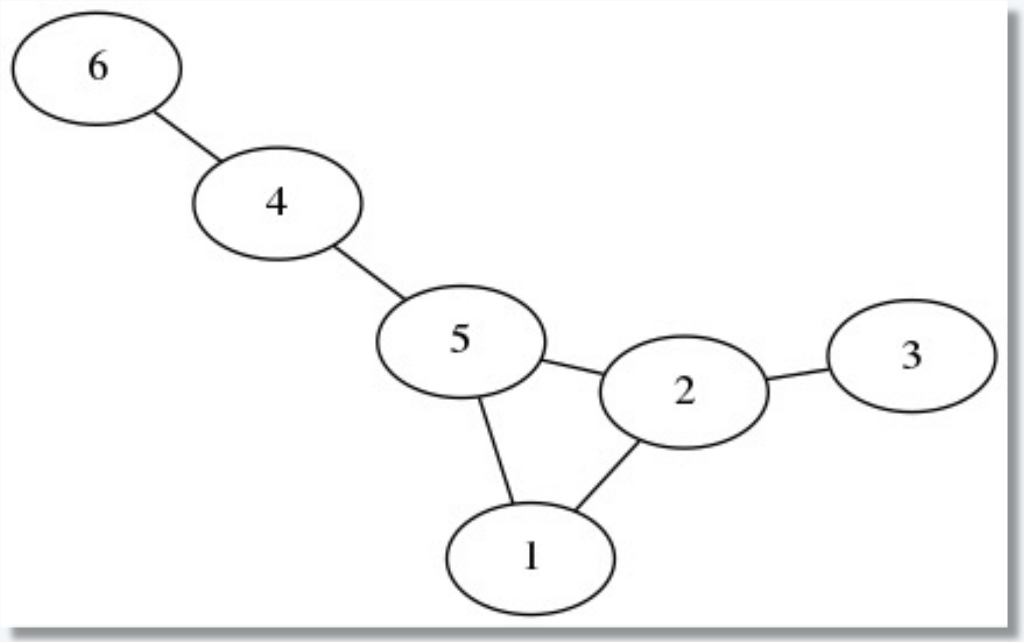
All Pairs Shortest Paths



Slides by Daniel Merkle

daniel@imada.sdu.dk

Powers of the Adjacency Matrix



$$A = \begin{array}{c c} & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

$$A^k = \underbrace{A \times A \ldots \times A}_{k \text{ times}} \text{ is called the k-th power of the adjacency matrix}$$

Theorem:

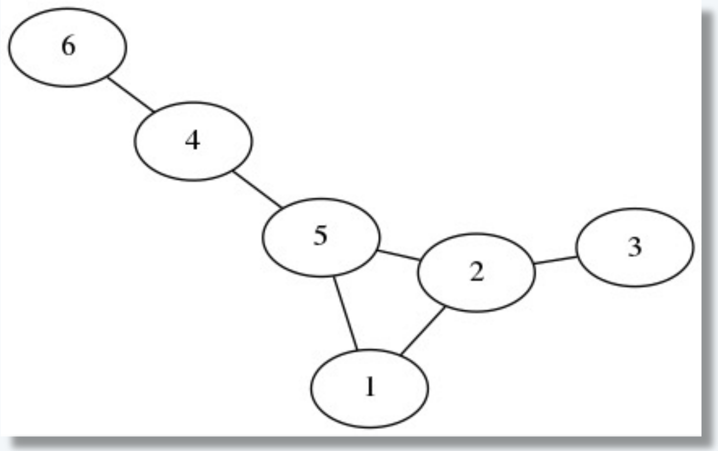
If G is a graph with adjacency matrix A, and vertices with indices $1, \ldots, n$ then for each positive integer k

the ij-th entry of A^k

is

the number of different walks using exactly k edges from node i to node j

$$A^2 = \begin{array}{c c} & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \begin{pmatrix} 2 & 1 & 1 & 1 & 1 & 0 \\ 1 & 3 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 3 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array} \quad A^3 = \begin{array}{c c} & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \begin{pmatrix} 2 & 4 & 1 & 1 & 4 & 1 \\ 4 & 2 & 3 & 1 & 5 & 1 \\ 1 & 3 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 4 & 2 \\ 4 & 5 & 1 & 4 & 2 & 0 \\ 1 & 1 & 0 & 2 & 0 & 0 \end{pmatrix} \end{array} \quad A^4 = \begin{array}{c c} & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \begin{pmatrix} 8 & 7 & 4 & 5 & 7 & 1 \\ 7 & 12 & 2 & 6 & 7 & 1 \\ 4 & 2 & 3 & 1 & 5 & 1 \\ 5 & 6 & 1 & 6 & 2 & 0 \\ 7 & 7 & 5 & 2 & 13 & 4 \\ 1 & 1 & 1 & 0 & 4 & 2 \end{pmatrix} \end{array}$$



$$A = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

<u>Example :</u>

Consider the two vertices with index 4 and 5 in A^4

Length 4 walks:

1) 4 -> 5 -> 1 -> 2 -> 5
2) 4 -> 5 -> 2 -> 1 -> 5

There are 2 walks of length 4.

Furthermore, A^4_{45}=2.

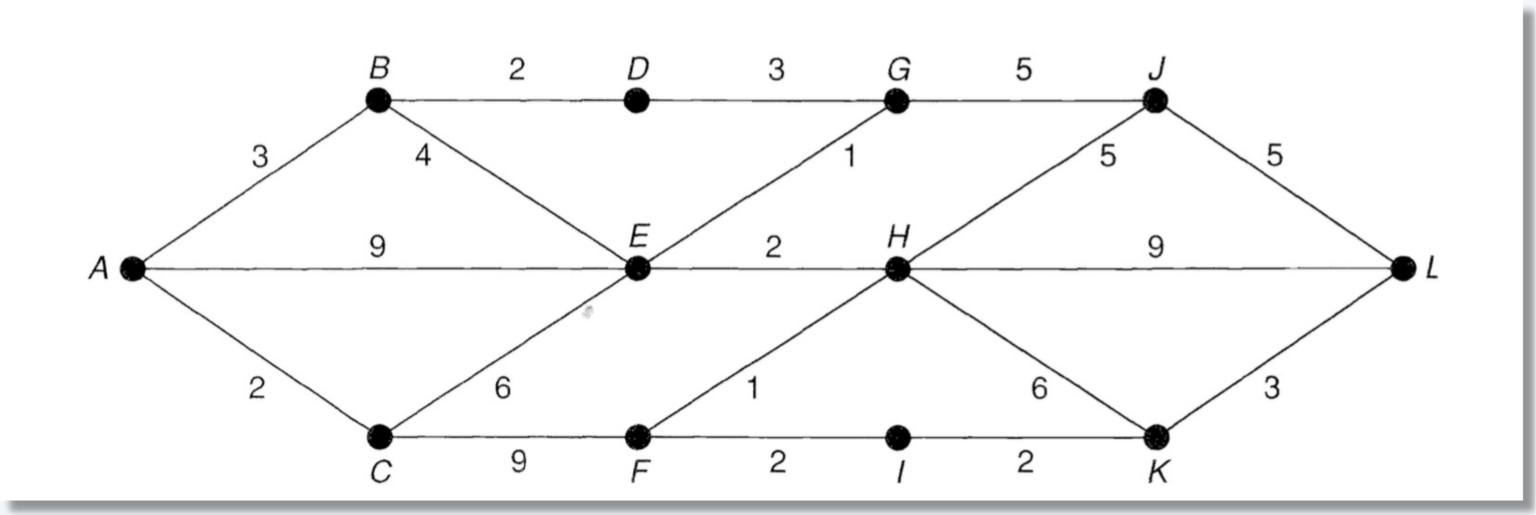
$$A^2 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 2 & 1 & 1 & 1 & 1 & 0 \\ 1 & 3 & 0 & 1 & \boxed{1} & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 2 & 0 & 0 \\ 1 & 1 & 1 & 0 & 3 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix} \end{array}$$

$$A^3 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 2 & 4 & 1 & 1 & 4 & 1 \\ 4 & 2 & 3 & 1 & 5 & 1 \\ 1 & 3 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 4 & 2 \\ 4 & 5 & 1 & 4 & 2 & 0 \\ 1 & 1 & 0 & 2 & 0 & 0 \end{pmatrix} \end{array}$$

$$A^4 = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} \begin{array}{c} \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{pmatrix} 8 & 7 & 4 & 5 & 7 & 1 \\ 7 & 12 & 2 & 6 & 7 & 1 \\ 4 & 2 & 3 & 1 & 5 & 1 \\ 5 & 6 & 1 & 6 & \boxed{2} & 0 \\ 7 & 7 & 5 & 2 & 13 & 4 \\ 1 & 1 & 1 & 0 & 4 & 2 \end{pmatrix} \end{array}$$

Algorithm for All-Pairs Shortest Path

Weighted Graph G with weights on edges:



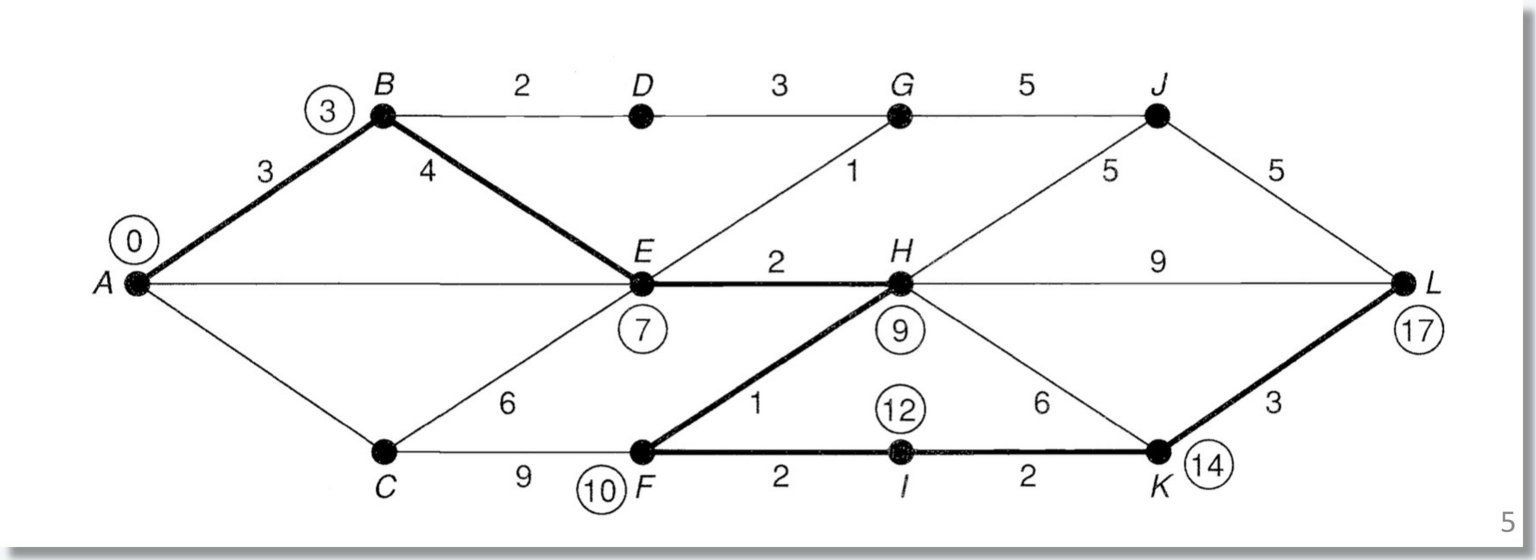
- What is the distance (=length of **the** shortest path) between A and L ?

17

<ins>Generalization:</ins>

- What are the distances of ALL paths (=lengths of ALL shortest paths) between all pairs of nodes?

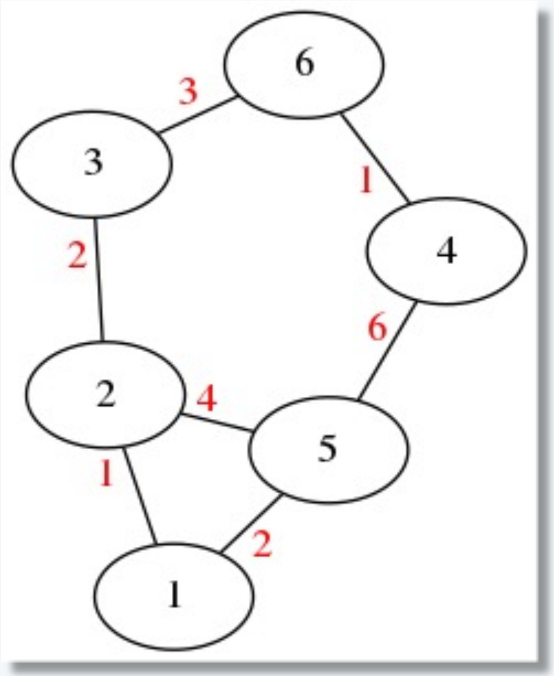
  … and how can we find all these distances?



The Edge Weight Matrix W

Example:

$$W = \begin{array}{c c} & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \begin{pmatrix} 0 & 1 & \infty & \infty & 2 & \infty \\ 1 & 0 & 2 & \infty & 4 & \infty \\ \infty & 2 & 0 & \infty & \infty & 3 \\ \infty & \infty & \infty & 0 & 6 & 1 \\ 2 & 4 & \infty & 6 & 0 & \infty \\ \infty & \infty & 3 & 1 & \infty & 0 \end{pmatrix} \end{array}$$



weights are depicted in red

Definition:

$$W_{ij} = \begin{cases} \text{the weight of the edge } (i,j) & \text{if the edge } (i,j) \text{ exists} \\ 0 & \text{if } i = j \\ \infty & \text{else} \end{cases}$$

Interpretation:

W_{ij} is the distance from vertex i to vertex j using maximally 1 edge

Note: Matrix W has entries corresponding to infinity, as it might be impossible to reach vertex j from vertex i via **1** edge.

We assume all weights are not negative, i.e., larger or equal to 0.

A modified Matrix-Matrix Multiplication

$$\begin{pmatrix} 1 & 0 & 2 \\ 1 & 2 & 4 \\ 3 & 1 & 2 \end{pmatrix} \odot \begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 2 \\ 1 & 2 & 5 \end{pmatrix} = \begin{pmatrix} 2 & 3 & 2 \\ 2 & 3 & 4 \\ 3 & 4 & 3 \end{pmatrix}$$

$$M \odot N = R$$

<u>Definition:</u>

$r_{ij} = \min_k\{m_{ik} + n_{kj}\}$

<u>Example:</u>

$r_{33} = \min\{3+3, 1+2, 2+5\} = 3$

Note: this operation is very similar to the standard matrix-matrix multiplication: however, for computation of the ij-th entry the multiplication is replaced by addition, and addition is replaced by the minimum operation.

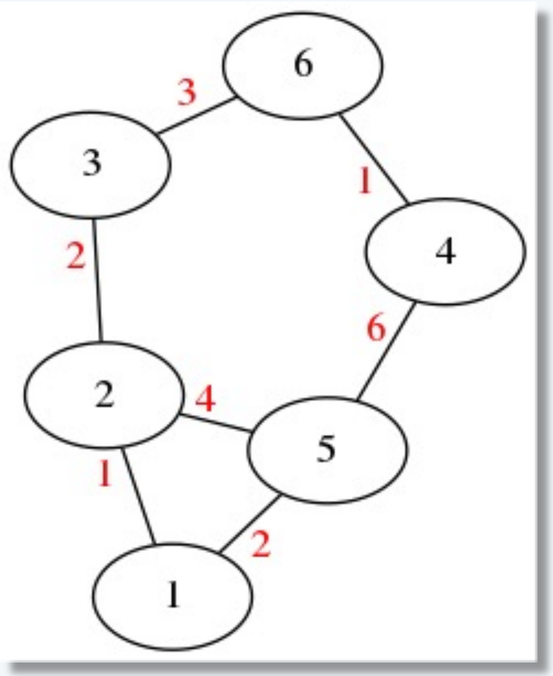
Theorem:

If G is a weighted graph with edge weight matrix W, and vertices with indices $1, \ldots, n$ then for each positive integer k

$$\text{the } ij\text{-th entry of } W^k = \underbrace{W \odot W \odot \ldots \odot W}_{k \text{ times}}$$

is

the length of the shortest path from i to j using maximally k edges



$$W = \begin{array}{cc} & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \begin{pmatrix} 0 & 1 & \infty & \infty & 2 & \infty \\ 1 & 0 & 2 & \infty & 4 & \infty \\ \infty & 2 & 0 & \infty & \infty & 3 \\ \infty & \infty & \infty & 0 & 6 & 1 \\ 2 & 4 & \infty & 6 & 0 & \infty \\ \infty & \infty & 3 & 1 & \infty & 0 \end{pmatrix} \end{array}$$

<u>Examples :</u>

Consider the two vertices with index 4 and 1 in W^4

Shortest Path using maximally 4 edges:

4 -> 6 -> 3 -> 2 -> 1 (distance 7)

Consider the two vertices with index 5 and 3 in W^4

Shortest Path using maximally 4 edges:

5 -> 1 -> 2 -> 3 (distance 5)

$$W^2 = \begin{array}{cc} & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \begin{pmatrix} 0 & 1 & 3 & 8 & 2 & \infty \\ 1 & 0 & 2 & 10 & 3 & 5 \\ 3 & 2 & 0 & 4 & \boxed{6} & 3 \\ 8 & 10 & 4 & 0 & 6 & 1 \\ 2 & 3 & 6 & 6 & 0 & 7 \\ \infty & 5 & 3 & 1 & 7 & 0 \end{pmatrix} \end{array}$$

$$W^3 = \begin{array}{cc} & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \begin{pmatrix} 0 & 1 & 3 & 8 & 2 & 6 \\ 1 & 0 & 2 & 6 & 3 & 5 \\ 3 & 2 & 0 & 4 & 5 & 3 \\ 8 & 6 & 4 & 0 & 6 & 1 \\ 2 & 3 & 5 & 6 & 0 & 7 \\ 6 & 5 & 3 & 1 & 7 & 0 \end{pmatrix} \end{array}$$

$$W^4 = \begin{array}{cc} & \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\ \begin{array}{c} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array} & \begin{pmatrix} 0 & 1 & 3 & 7 & 2 & 6 \\ 1 & 0 & 2 & 6 & 3 & 5 \\ 3 & 2 & 0 & 4 & 5 & 3 \\ \boxed{7} & 6 & 4 & 0 & 6 & 1 \\ 2 & 3 & \boxed{5} & 6 & 0 & 7 \\ 6 & 5 & 3 & 1 & 7 & 0 \end{pmatrix} \end{array}$$

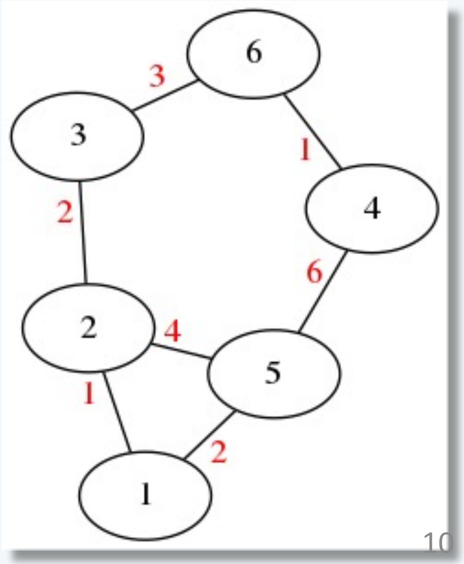
$$W = \begin{array}{cc} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix} & \begin{pmatrix} 0 & 1 & \infty & \infty & 2 & \infty \\ 1 & 0 & 2 & \infty & 4 & \infty \\ \infty & 2 & 0 & \infty & \infty & 3 \\ \infty & \infty & \infty & 0 & 6 & 1 \\ 2 & 4 & \infty & 6 & 0 & \infty \\ \infty & \infty & 3 & 1 & \infty & 0 \end{pmatrix} \end{array} \quad W^2 = \begin{array}{cc} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix} & \begin{pmatrix} 0 & 1 & 3 & 8 & 2 & \infty \\ 1 & 0 & 2 & 10 & 3 & 5 \\ 3 & 2 & 0 & 4 & 6 & 3 \\ 8 & 10 & 4 & 0 & 6 & 1 \\ 2 & 3 & 6 & 6 & 0 & 7 \\ \infty & 5 & 3 & 1 & 7 & 0 \end{pmatrix} \end{array} \quad W^3 = \begin{array}{cc} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix} & \begin{pmatrix} 0 & 1 & 3 & 8 & 2 & 6 \\ 1 & 0 & 2 & 6 & 3 & 5 \\ 3 & 2 & 0 & 4 & 5 & 3 \\ 8 & 6 & 4 & 0 & 6 & 1 \\ 2 & 3 & 5 & 6 & 0 & 7 \\ 6 & 5 & 3 & 1 & 7 & 0 \end{pmatrix} \end{array}$$

$$W^4 = \begin{array}{cc} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix} & \begin{pmatrix} 0 & 1 & 3 & 7 & 2 & 6 \\ 1 & 0 & 2 & 6 & 3 & 5 \\ 3 & 2 & 0 & 4 & 5 & 3 \\ 7 & 6 & 4 & 0 & 6 & 1 \\ 2 & 3 & 5 & 6 & 0 & 7 \\ 6 & 5 & 3 & 1 & 7 & 0 \end{pmatrix} \end{array} \quad W^5 = \begin{array}{cc} & \begin{matrix} 1 & 2 & 3 & 4 & 5 & 6 \end{matrix} \\ \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{matrix} & \begin{pmatrix} 0 & 1 & 3 & 7 & 2 & 6 \\ 1 & 0 & 2 & 6 & 3 & 5 \\ 3 & 2 & 0 & 4 & 5 & 3 \\ 7 & 6 & 4 & 0 & 6 & 1 \\ 2 & 3 & 5 & 6 & 0 & 7 \\ 6 & 5 & 3 & 1 & 7 & 0 \end{pmatrix} \end{array}$$

$$W \neq W^2 \neq W^3 \neq W^4 = W^5 = W^6 = \ldots$$

Which value of k is necessary, in order to have W^k contain all the pairwise distances of all vertexes?

Answer: $n-1$ (which is identical to $|V|-1$)



Assume all edge weights are not negative. The number of edges needed for a shortest path can maximally be n-1, where n is the number of vertices in the graph. If the path would go via n edges, then you would have to visit at least one vertex twice, but then the path cannot be a shortest path anymore. Obviously $W^k = W^{n-1}$ for all k>n-1.

Lemma:

If G is a weighted graph with edge weight matrix W, and vertices with indices $1, \ldots, n$ then

$$\text{the } ij\text{-th entry of } W^{n-1} = \underbrace{W \odot W \odot \ldots \odot W}_{n-1 \text{ times}}$$

is

the distance from i to j

$D := W^{n-1}$ is called the distance matrix of the graph G.

Computation of the Distance Matrix by Repeated Squaring

$$W^{n-1} = \underbrace{\left(\underbrace{\left(\underbrace{\left(\underbrace{\left(\underbrace{(W \odot W)}_{W^2} \odot W\right)}_{W^3} \odot W\right)}_{W^4} \odot W\right)}_{W^5} \odot \ldots \odot W\right)}_{W^{n-1}}$$

n-2 matrix-matrix multiplication are needed in order to compute the distance matrix $D = W^{n-1}$

$$W^{(2^k)} = \underbrace{\left(\left(\underbrace{\left(\underbrace{\left(\underbrace{(W \odot W)}_{W^2}\right)^2}_{W^4}\right)^2}_{W^8}\right)^2 \cdots\right)^2}_{W^{(2^k)}}$$

k matrix-matrix multiplication are needed (namely squaring a matrix k times) in order to compute the matrix $W^{(2^k)}$

2^k has to be larger or equal to n-1, or equivalently, k has to be larger or equal to $\log_2(n-1)$

<u>Example:</u> Consider a graph G with 101 vertices. In order to compute the distance matrix $\mathrm{D} = W^{100}$, the left approach needs to make 99 matrix-matrix multiplications. The right approach (called repeated squaring) requires only 7 matrix-matrix multiplications, as $2^7 = 128$, and $\mathrm{D} = W^{128} = W^{100}$

Another Application of the Distance Matrix: Predicting Boiling Points of Paraffins

In 1947 Harry Wiener defined the **Wiener-Index** of a graph G in order to predict the boiling point of different paraffins. He used the graph representation G of the carbon backbone of a molecule with *n* carbon atoms and calculated the Wiener-Index the sum of all distances between all pairs of vertexes, i.e.

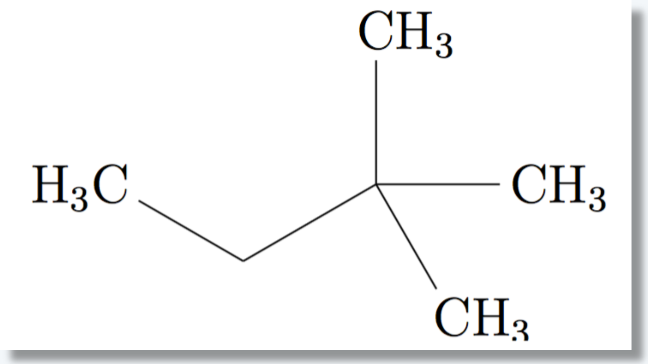
$$\mathcal{W}(G) = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} D_{ij}$$

He predicted the boiling point t_B to be

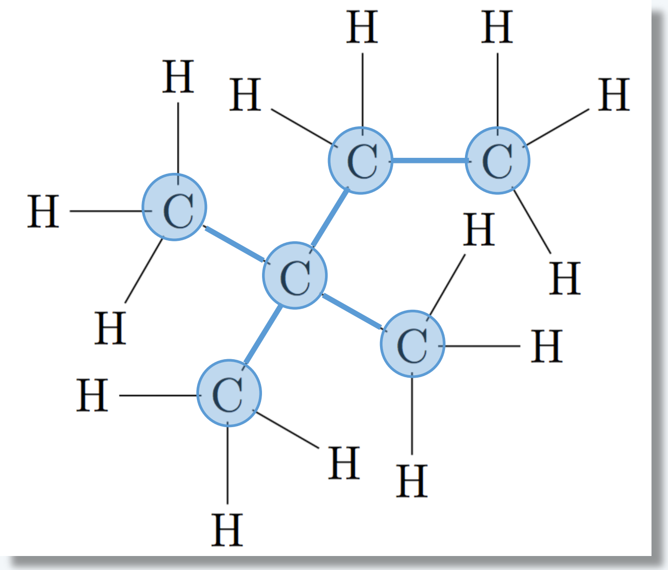
$$t_B = t_0 - \left(\frac{98}{n^2}(w_0 - \mathcal{W}(G)) + 5.5 \cdot (p_0 - p)\right)$$

with
$$t_0 = 745.42 \cdot \log_{10}(n + 4.4) - 689.4$$
$$w_0 = \frac{1}{6} \cdot (n+1) \cdot n \cdot (n-1)$$
$$p_0 = n - 3$$
$$p = \text{the number of shortest paths } i \to \ldots \to j \text{ of length 3 in } G \text{ with } i < j$$
$$= \text{half of the number of entries "3" in the distance matrix } D$$

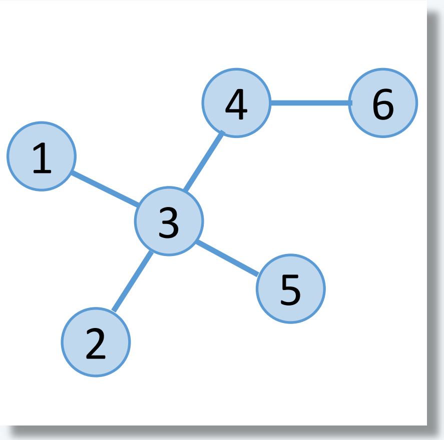
Wiener Index : Boiling Point Prediction, Example (2,2-dimethylbutan)



The chemical compound



The carbon backbone



Graph G

$$
W = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\
\begin{pmatrix}
0 & \infty & 1 & \infty & \infty & \infty \\
\infty & 0 & 1 & \infty & \infty & \infty \\
1 & 1 & 0 & 1 & 1 & \infty \\
\infty & \infty & 1 & 0 & \infty & 1 \\
\infty & \infty & 1 & \infty & 0 & \infty \\
\infty & \infty & \infty & 1 & \infty & 0
\end{pmatrix}
\end{array}
$$

Edge Weight Matrix

Note: Depending on how you chose to label your graph, the edge weight matrix might look different. This won't matter for the subsequent calculations.

$$
D = W^{n-1} = \begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{array}
\begin{array}{c}
\begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \end{array} \\
\begin{pmatrix}
0 & 2 & 1 & 2 & 2 & 3 \\
2 & 0 & 1 & 2 & 2 & 3 \\
1 & 1 & 0 & 1 & 1 & 2 \\
2 & 2 & 1 & 0 & 2 & 1 \\
2 & 2 & 1 & 2 & 0 & 3 \\
3 & 3 & 2 & 1 & 3 & 0
\end{pmatrix}
\end{array}
$$

Distance Matrix

$$\mathcal{W}(G) = \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} D_{ij} = 28$$

$$t_0 = 68.72$$

$$w_0 = \frac{1}{6} \cdot 5 \cdot 6 \cdot 7 = 35$$

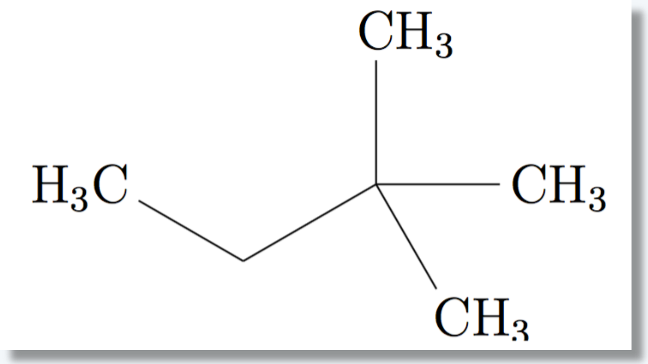
$$p_0 = 6 - 3 = 3$$

$$p = 3$$

$$
\begin{aligned}
t_B &= t_0 - \left(\frac{98}{n^2} (w_0 - \mathcal{W}(G)) + 5.5 \cdot (p_0 - p) \right) \\
&= 68.72 - \frac{98}{36}(35 - 28) - 5.5 \cdot (3 - 3) \\
&= 49.66
\end{aligned}
$$

Calculation of Wiener Index and other parameters, as well as the resulting boiling point prediction.

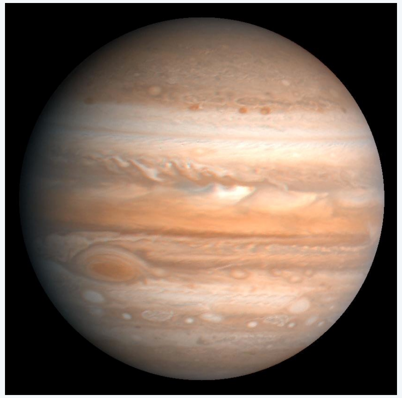
Wiener Index : Boiling Point Prediction, Example (2,2-dimethylbutan)



$$\text{Predicted Boiling Point:} \quad t_B = 49.66$$

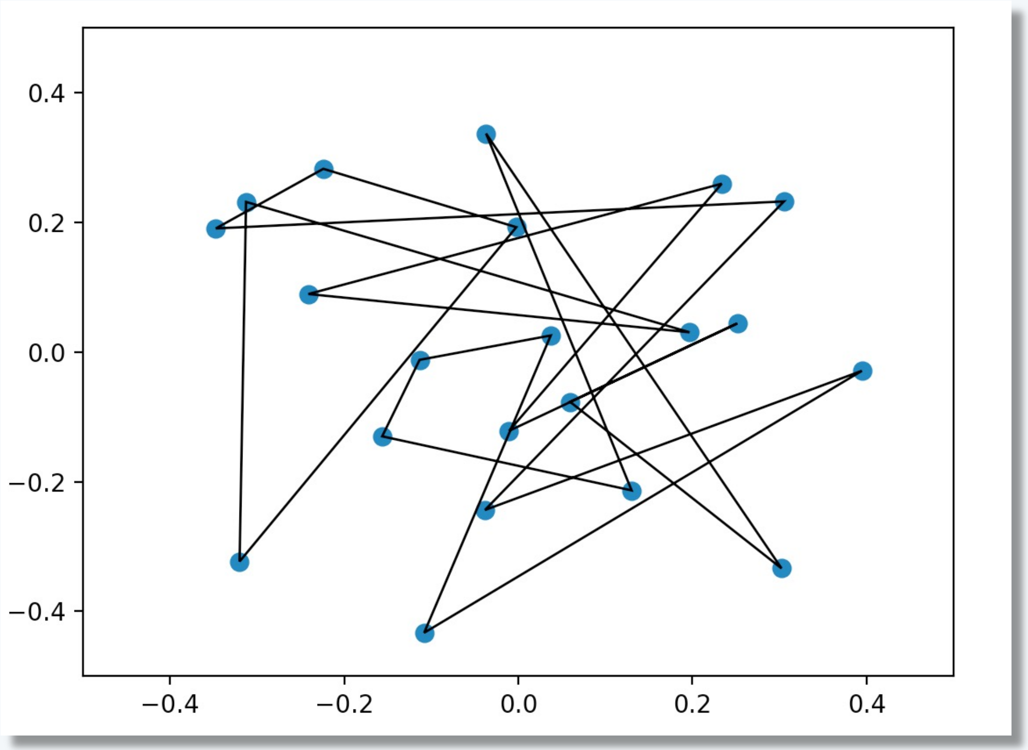
$$\text{Real Boiling Point:} \quad t_B^{\text{real}} \approx 49.7 - 50.0$$

The prediction of boiling points of paraffins based on the Wiener-Index of the corresponding molecular graph is amazingly accurate. Try it yourself (see exercises)! Intuitively, the Wiener-Index quantifies the “compactness” of a graph (or molecule). Long single chained molecules with n carbons have a larger Wiener-Index than molecules that contain many branches. Long molecules tend to align nicely, and have usually a higher boiling point.



DM587 – Scientific Programming

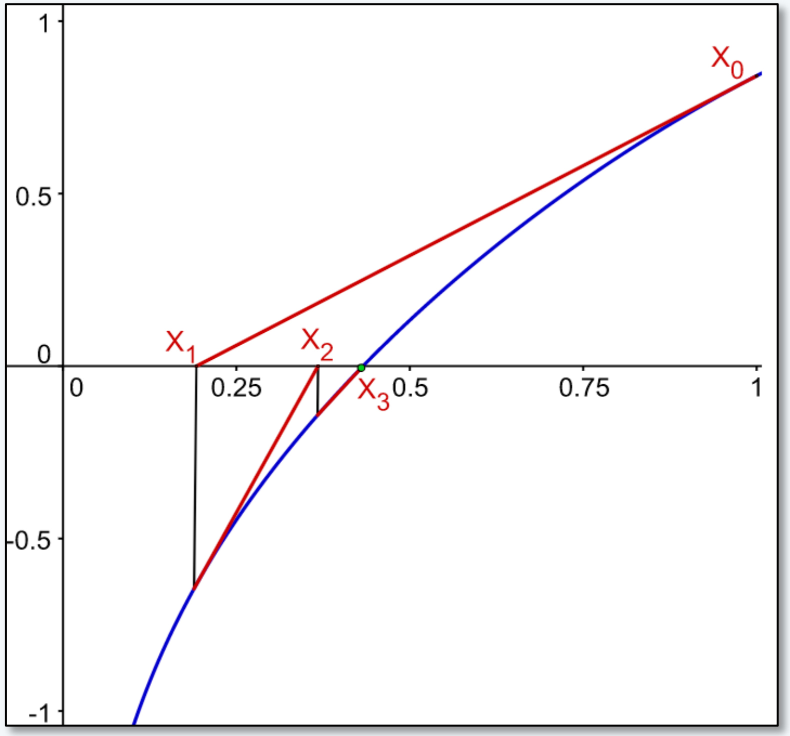
From Random Polygon to Ellipse



Slides by Daniel Merkle

daniel@imada.sdu.dk

Iterative Methods



- Iterative methods for
  - Finding roots
  - Finding matrix inverse
  - Solving systems of linear equations
  - …

John von Neumann Lecture

Established in 1959, the prize honors John von Neumann, a founder of modern computing. The lecture is awarded annually for distinguished contributions to applied mathematics and for the effective communication of these ideas to the community.

Prize Description

The John von Neumann Lecture is awarded annually to one individual for outstanding and distinguished contributions to the field of applied mathematics and for the effective communication of these ideas to the community. It is one of SIAM's most distinguished prizes as well as an important lecture at the SIAM Annual Meeting.

The 2018 winner: Charles Francis Van Loan

The topic of this lecture is based on his speech when receiving the award (Sept 2018).

Linear Algebra Applications

<u>Based on:</u>

A.N. Elmachtoub, C.F. Van Loan (2010), From random polygon to ellipse: an eigenanalysis, SIAM Rev. 52, 151–170.

John von Neumann prize lecture, Sept 2018, for C.F. van Loan

Intro Programming with Matlab (2008)

↓

Intro Matrix Computations (2009)

↓

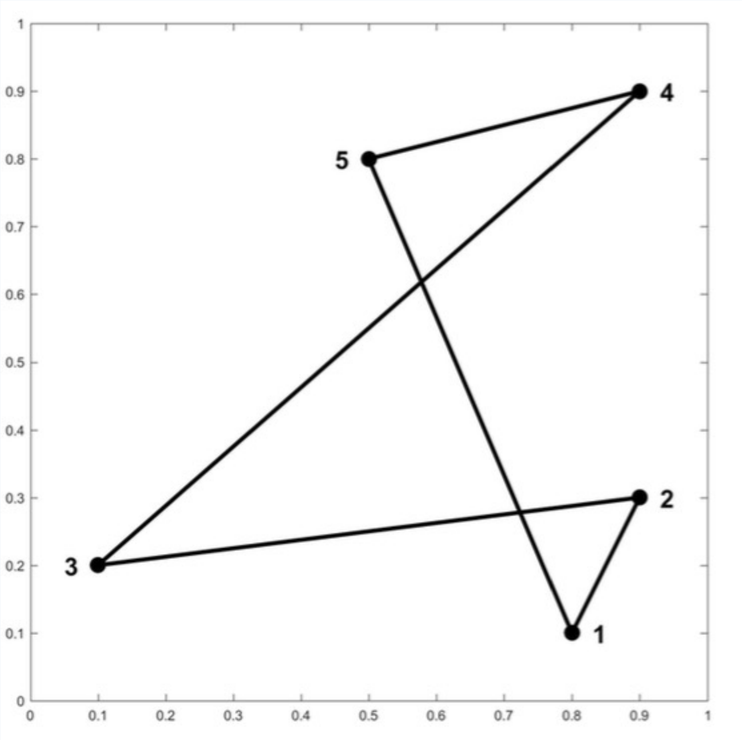
SIAM Review (2010)

↓

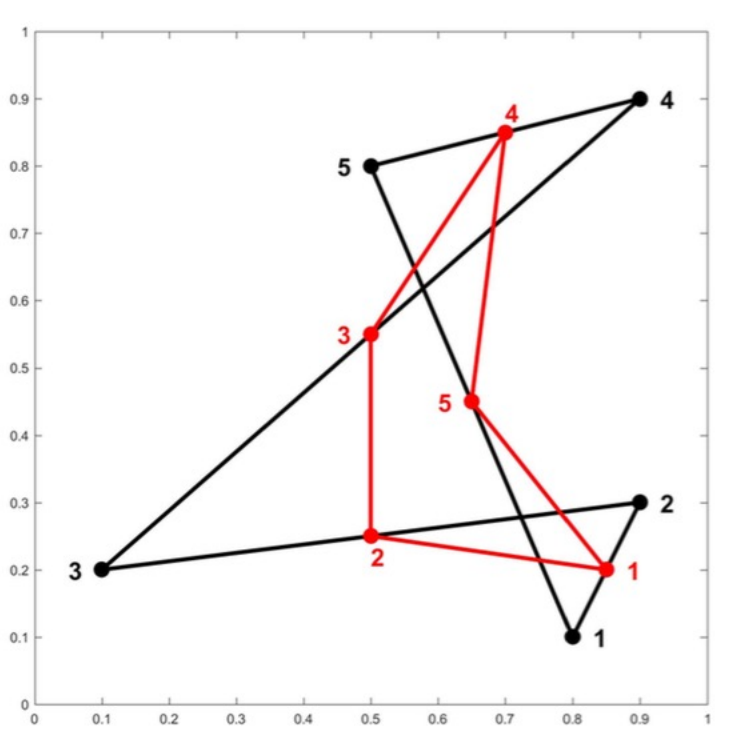
SIAM News (2018)

Display a sequence of polygons where each polygon is obtained from its predecessor by connecting the midpoints of its sides.

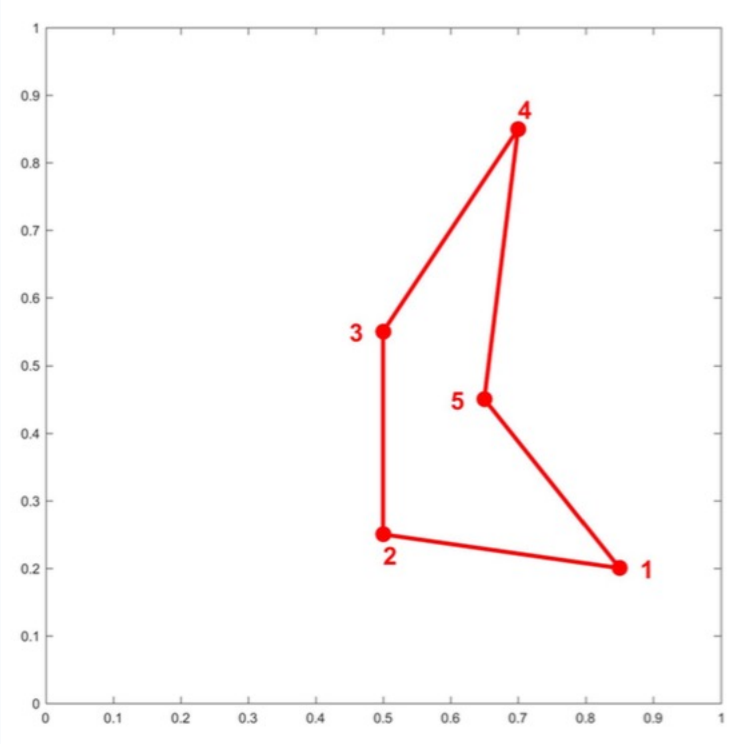
Let the original polygon be random.



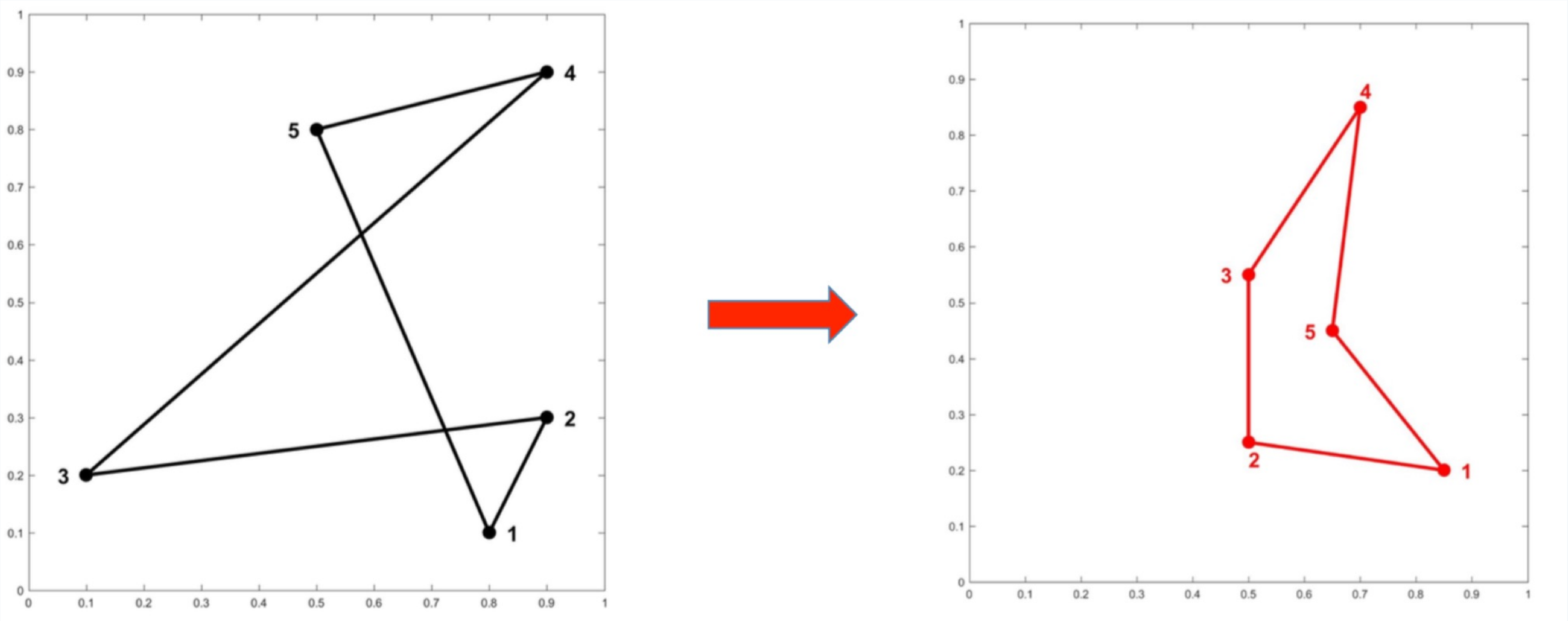
A random polygon (5 points)



Connecting the midpoints



New polygon



One step. This obviously can be repeated.

One Step in Vector Terminology

Assume the following five points define a close polygon: $(x_1, y_1), (x_2, y_2), (x_3, y_3), (x_4, y_4), (x_5, y_5)$

$$\begin{bmatrix} \widehat{x}_1 \\ \widehat{x}_2 \\ \widehat{x}_3 \\ \widehat{x}_4 \\ \widehat{x}_5 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} x_1 + x_2 \\ x_2 + x_3 \\ x_3 + x_4 \\ x_4 + x_5 \\ x_5 + x_1 \end{bmatrix} \qquad \begin{bmatrix} \widehat{y}_1 \\ \widehat{y}_2 \\ \widehat{y}_3 \\ \widehat{y}_4 \\ \widehat{y}_5 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} y_1 + y_2 \\ y_2 + y_3 \\ y_3 + y_4 \\ y_4 + y_5 \\ y_5 + y_1 \end{bmatrix}$$

One Step in Matrix Terminology

$$\widehat{x} = \begin{bmatrix} \widehat{x}_1 \\ \widehat{x}_2 \\ \widehat{x}_3 \\ \widehat{x}_4 \\ \widehat{x}_5 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} x_1 + x_2 \\ x_2 + x_3 \\ x_3 + x_4 \\ x_4 + x_5 \\ x_5 + x_1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \equiv M_5 x$$

$$\widehat{y} = \begin{bmatrix} \widehat{y}_1 \\ \widehat{y}_2 \\ \widehat{y}_3 \\ \widehat{y}_4 \\ \widehat{y}_5 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} y_1 + y_2 \\ y_2 + y_3 \\ y_3 + y_4 \\ y_4 + y_5 \\ y_5 + y_1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} \equiv M_5 y.$$

In general

$$\widehat{x} = M_n x$$

$$\widehat{y} = M_n y$$

$$M_n = \frac{1}{2}\begin{bmatrix} 1 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & & \vdots \\ \vdots & \vdots & & \ddots & 1 & \vdots \\ 0 & 0 & \cdots & & 1 & 1 \\ 1 & 0 & \cdots & \cdots & 0 & 1 \end{bmatrix}$$

A First Try

length = 1, i.e., $x^{(0)} = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{pmatrix}$

$$||x^{(0)}||_2 = \sqrt{\sum_{1 \le i \le n} x_i^2} = 1$$

Algorithm 1

Input: Unit 2-norm n-vectors $x^{(0)}$ *and* $y^{(0)}$.

Display $\mathcal{P}_0 = \mathcal{P}(x^{(0)}, y^{(0)})$.

for $k = 1, 2, \ldots$

% Compute $\mathcal{P}_k = \mathcal{P}(x^{(k)}, y^{(k)})$ from $\mathcal{P}_{k-1} = \mathcal{P}(x^{(k-1)}, y^{(k-1)})$

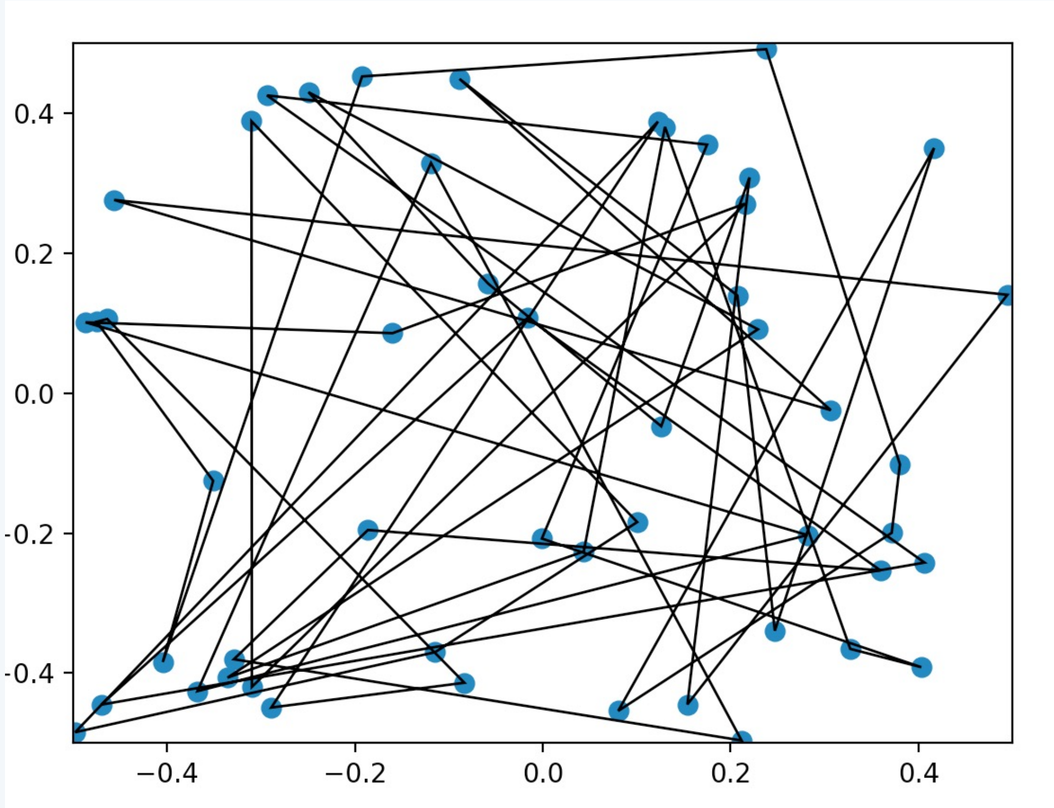
$$x^{(k)} = M_n x^{(k-1)}$$

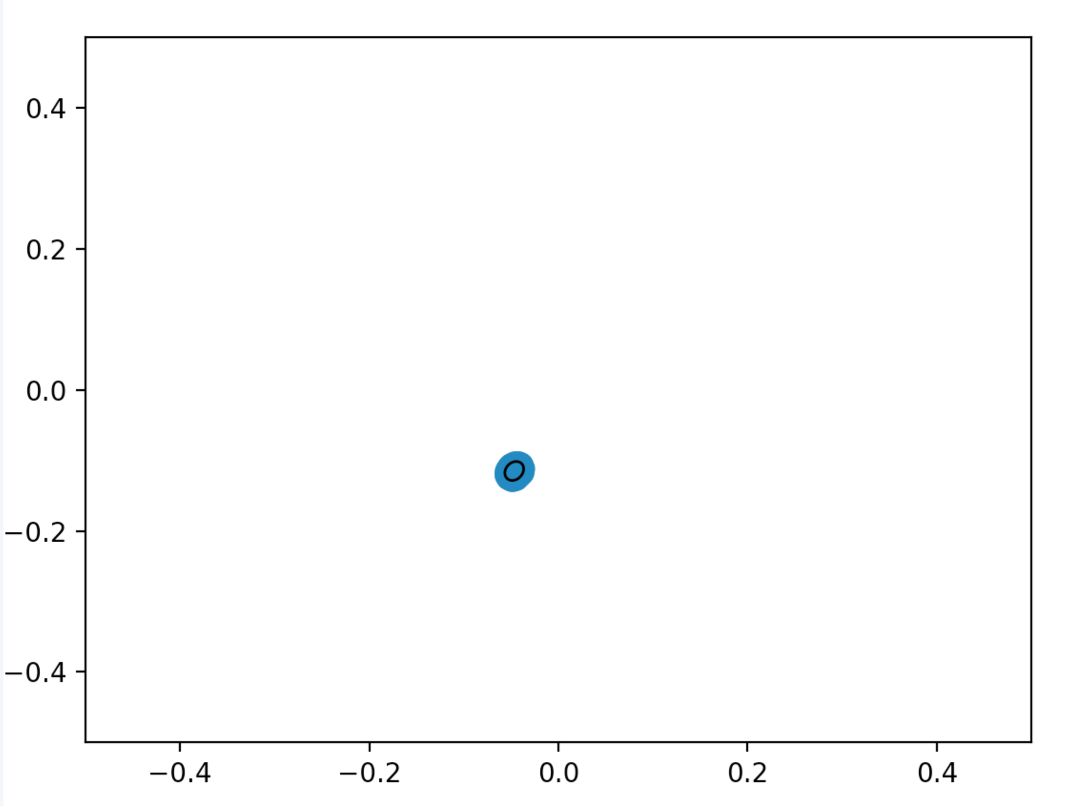
$$y^{(k)} = M_n y^{(k-1)}$$

Display $\mathcal{P}_k$.

end

Not too interesting …





The points seem to converge to a point. Not too surprisingly, it’s the centroid of the input points.

Centroid remains unchanged

Centroid:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{e^T x}{n} \qquad \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i = \frac{e^T x}{n}$$

$$e = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix}$$

As $e^T M_n = e^T$ it holds:

$$\frac{e^T x^{(k)}}{n} = \frac{e^T M_n x^{(k-1)}}{n} = \frac{e^T x^{(k-1)}}{n}$$

$$\frac{e^T y^{(k)}}{n} = \frac{e^T M_n y^{(k-1)}}{n} = \frac{e^T y^{(k-1)}}{n}.$$

A Second Try

$$x^{(0)} = \begin{pmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{pmatrix}$$

$$\sum_{1 \le i \le n} x_i = 0$$

Algorithm 2

Input: Unit 2-norm n-vectors $x^{(0)}$ *and* $y^{(0)}$ *whose components sum to zero.*

Display $\mathcal{P}_0 = \mathcal{P}(x^{(0)}, y^{(0)})$.

for $k = 1, 2, \ldots$

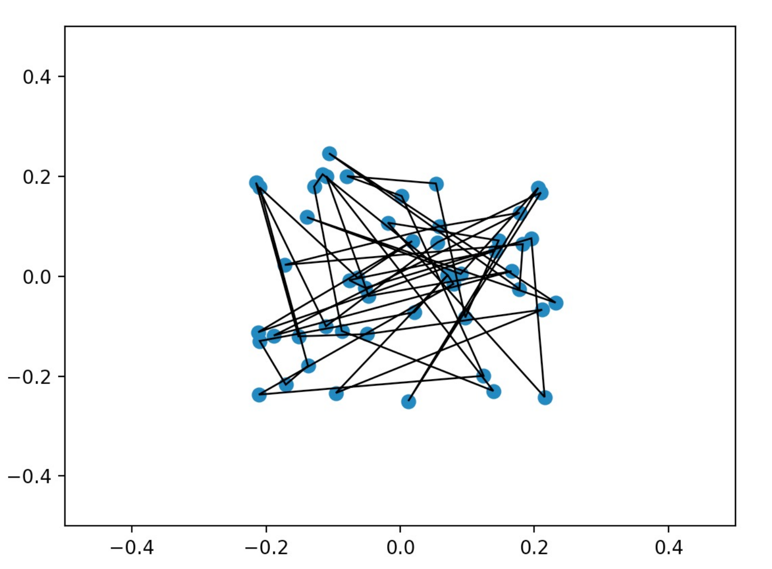
% Compute $\mathcal{P}_k = \mathcal{P}(x^{(k)}, y^{(k)})$ from $\mathcal{P}_{k-1} = \mathcal{P}(x^{(k-1)}, y^{(k-1)})$

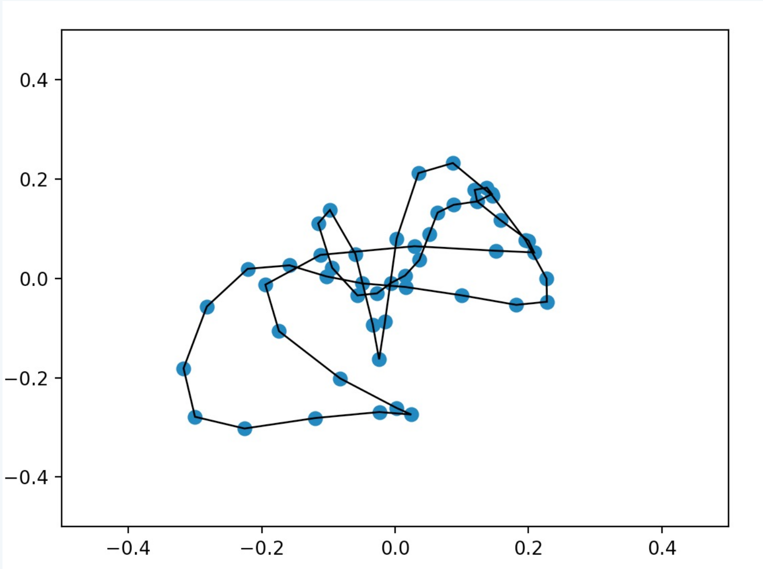
$f = M_n x^{(k-1)}, \quad x^{(k)} = f / \| f \|_2$

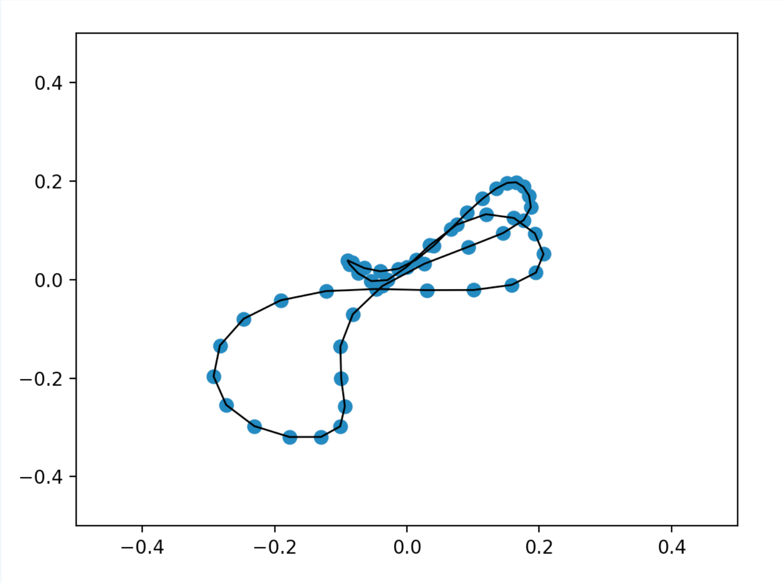
$g = M_n y^{(k-1)}, \quad y^{(k)} = g / \| g \|_2$

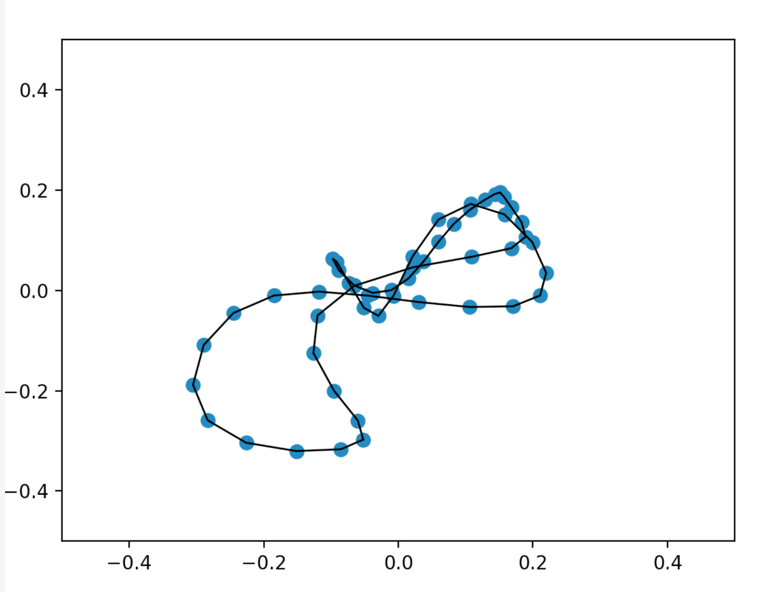
Display $\mathcal{P}_k$.

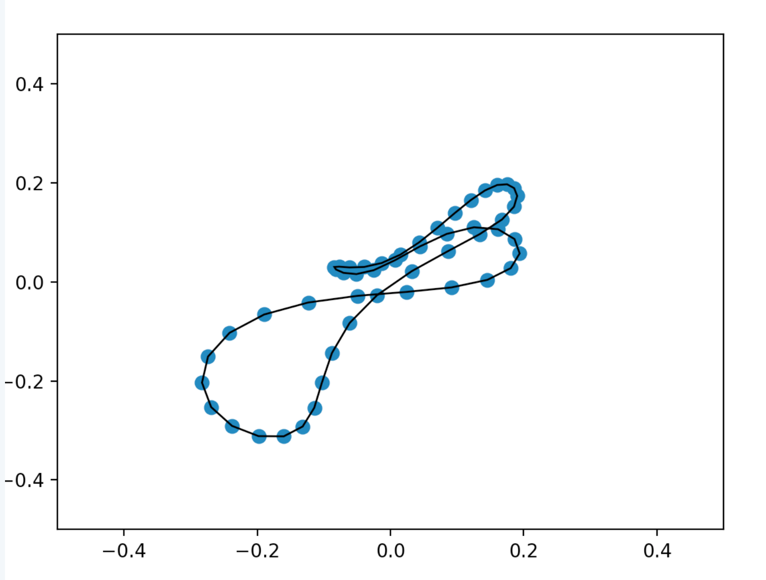
end





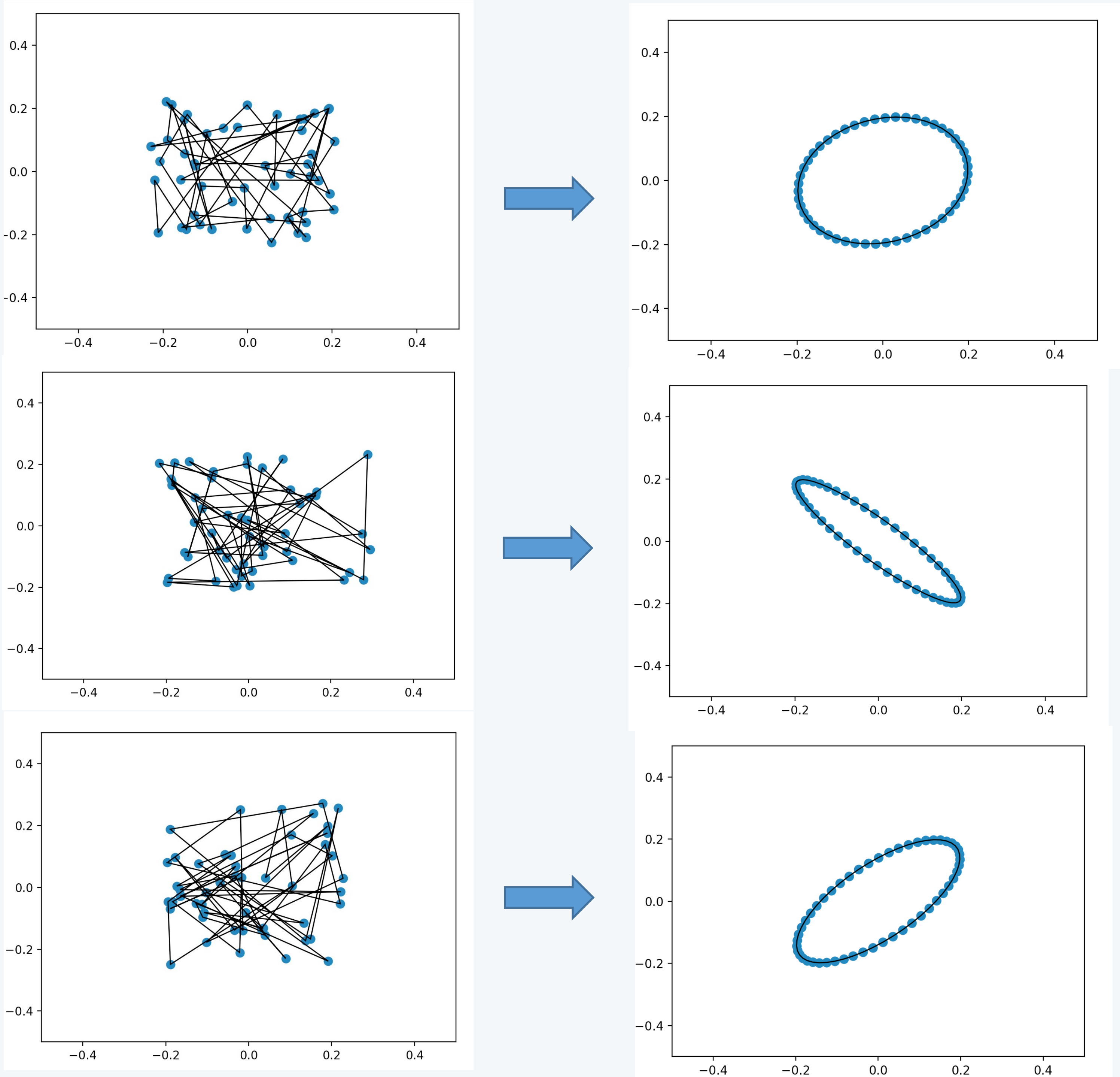


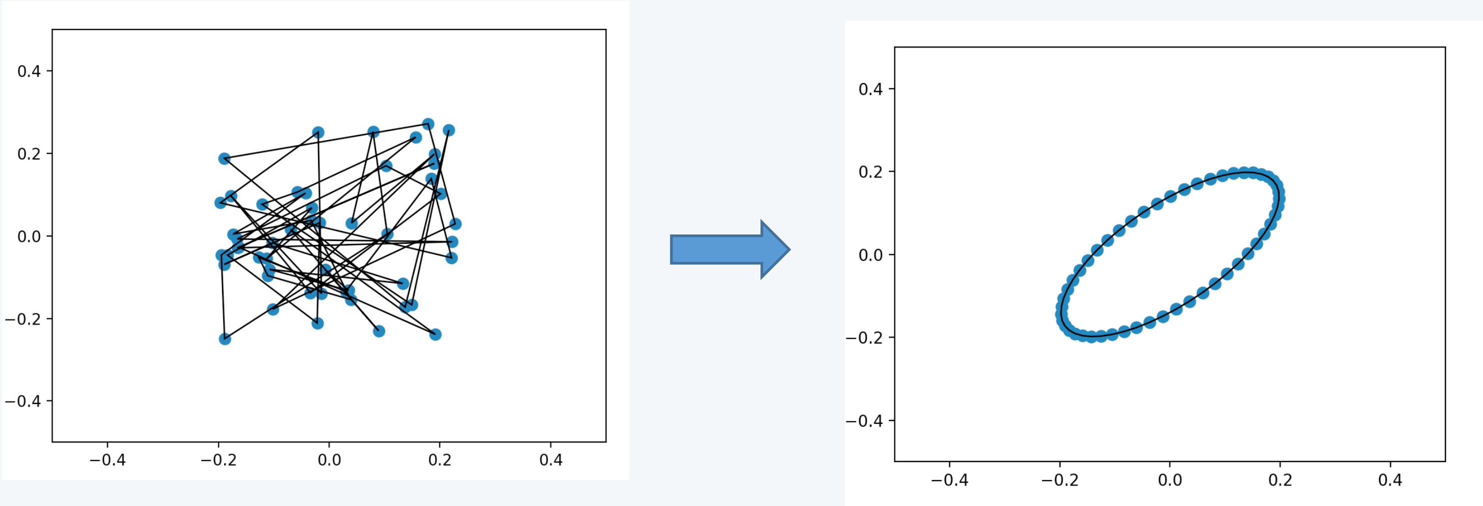




What’s happening???

Three test runs





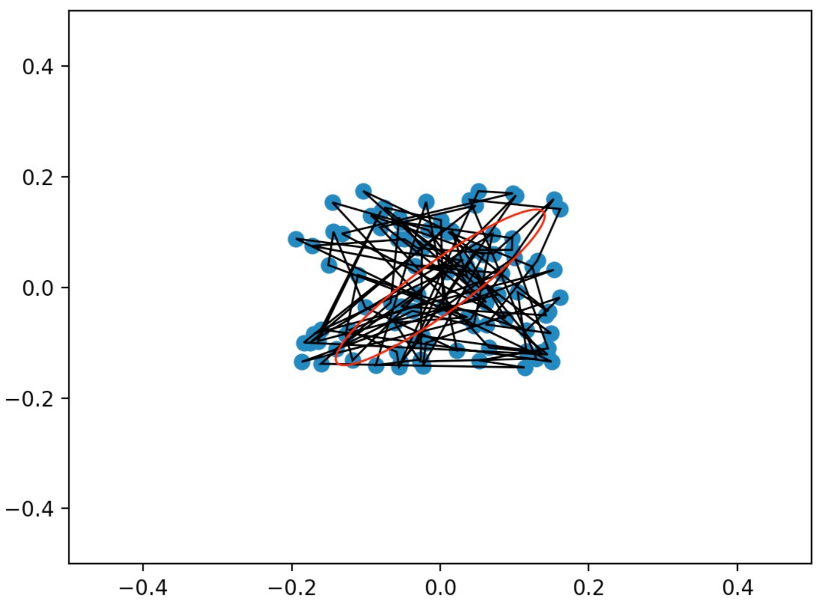
The points seem to converge to an ellipse with a 45-degree tilt

1. What is the limiting ellipse and why the 45-degree tilt?
2. How long does it take to "converge"?
3. Does it always converge?
4. What is the inverse of the repeated polygon averaging process, and does it always exits?

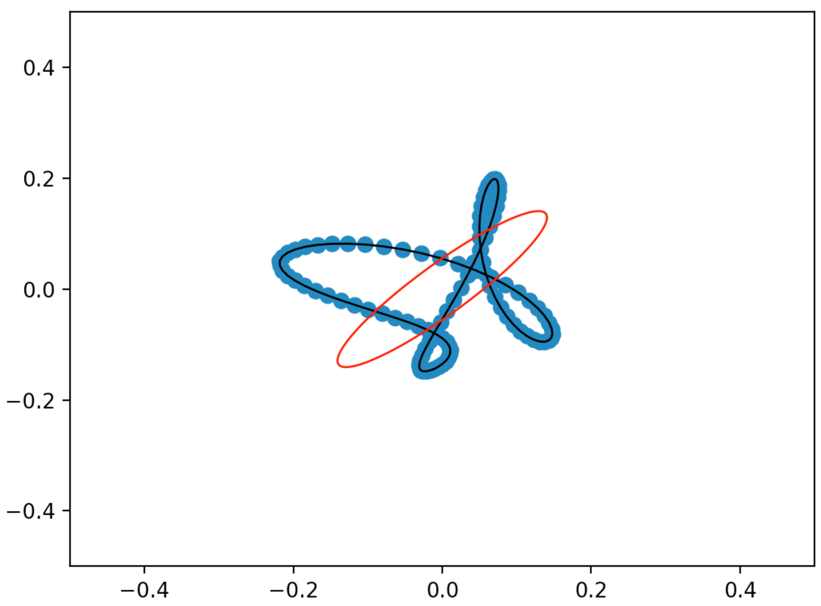
The ellipse can be computed in advance

Example: Polygon with 101 points

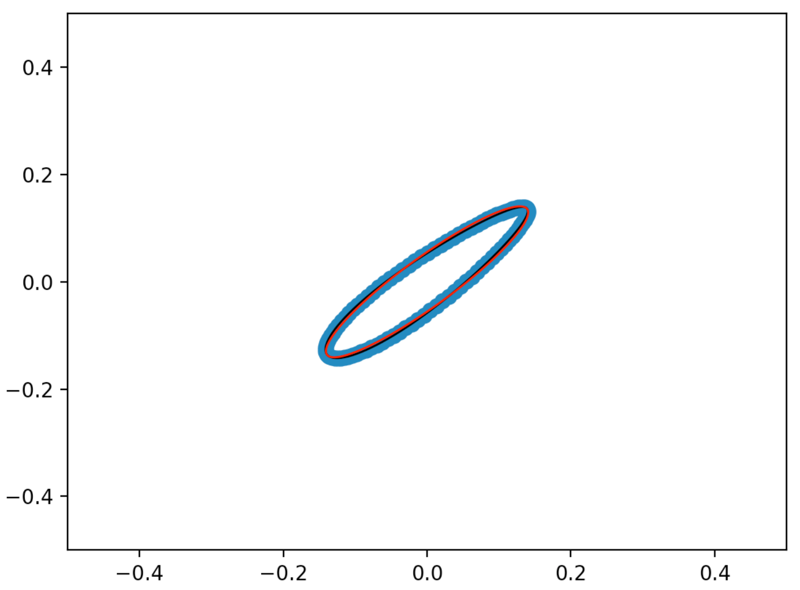
Step 1



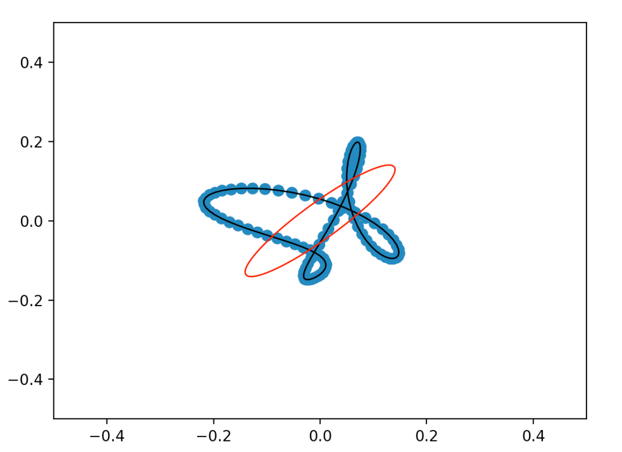
Step 100



Step 500



Pre-Computation of the ellipse



Background:

The analysis is not trivial, and requires

- an understanding of the eigen-system of the matrix M
- decomposition techniques (SVD- and Schur- decomposition)
- Interested?
  - Follow the wonderful von Neumann Price Lecture from **Charles F. Van Loan**
  - Read the (not so easy parts of the) paper
    “**From Random Polygon to Ellipse: An Eigenanalysis”, Adam N. Elmachtoub and Charles F. van Loan**

Pre-Computation of the ellipse

For a polygon with n points, define vectors **c** and **s** as follows (they form a orthonormal basis)

$$\tau = \begin{bmatrix} 0 \\ 2\pi/n \\ 4\pi/n \\ \vdots \\ 2(n-1)\pi/n \end{bmatrix} \qquad c = \sqrt{2/n}\begin{bmatrix} \cos(\tau_1) \\ \cos(\tau_2) \\ \vdots \\ \cos(\tau_n) \end{bmatrix} \qquad s = \sqrt{2/n}\begin{bmatrix} \sin(\tau_1) \\ \sin(\tau_2) \\ \vdots \\ \sin(\tau_n) \end{bmatrix}$$

Pre-Computation of the "ellipse"

The polygon converges (at even even steps) to the closed polygon defined by vector $u^{(0)}$ and $v^{(0)}$

$$u^{(0)} = \cos(\theta_u)c \; + \; \sin(\theta_u)s$$

$$v^{(0)} = \cos(\theta_v)c \; + \; \sin(\theta_v)s$$

where

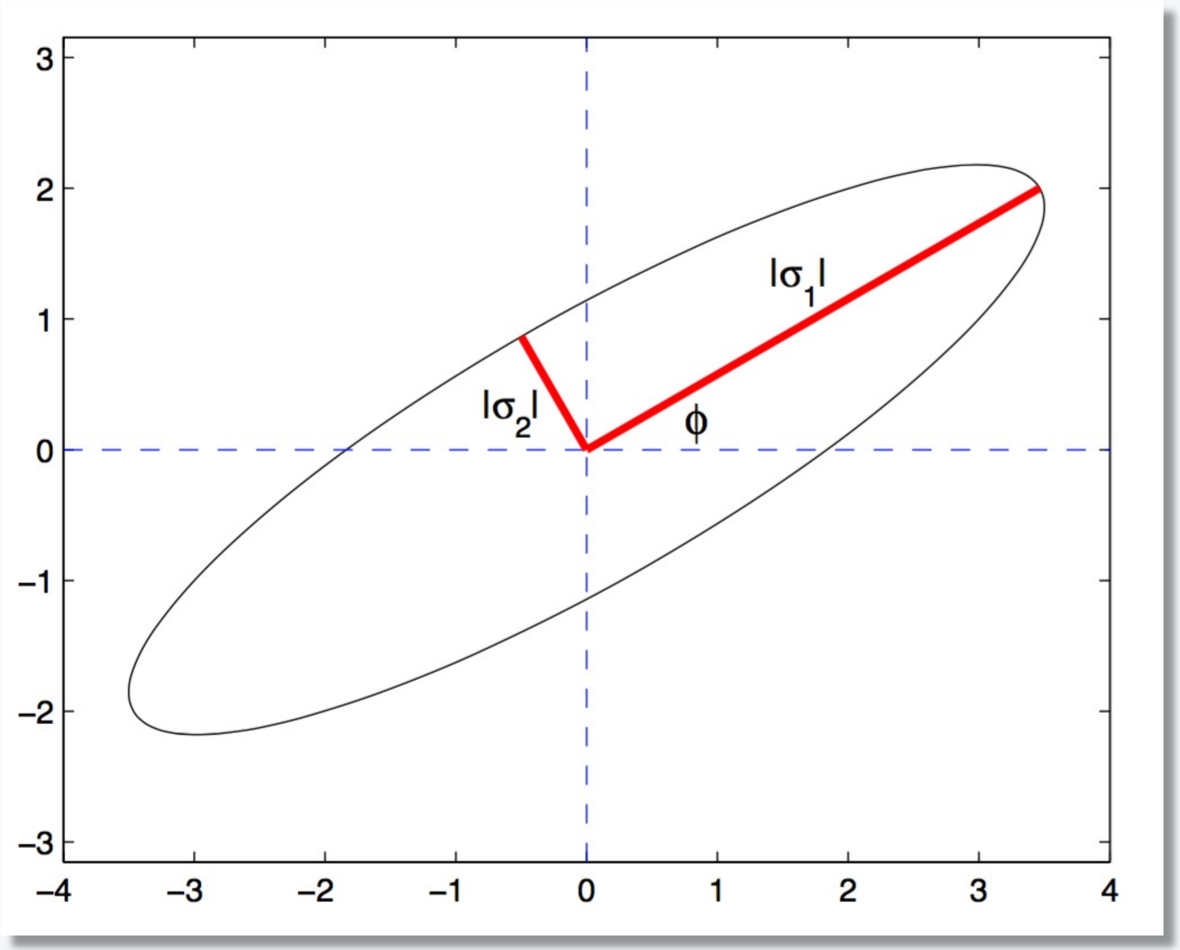
$$\cos(\theta_u) \; = \; \frac{c^T x^{(0)}}{\sqrt{(c^T x^{(0)})^2 + (s^T x^{(0)})^2}} \qquad \sin(\theta_u) \; = \; \frac{s^T x^{(0)}}{\sqrt{(c^T x^{(0)})^2 + (s^T x^{(0)})^2}}$$

$$\cos(\theta_v) = \frac{c^T y^{(0)}}{\sqrt{(c^T y^{(0)})^2 + (s^T y^{(0)})^2}} \qquad \sin(\theta_v) \; = \; \frac{s^T y^{(0)}}{\sqrt{(c^T y^{(0)})^2 + (s^T y^{(0)})^2}}.$$

Proof: paper! (Note, TYPO in paper)

Pre-Computation of the ellipse

Also in the paper: An understanding of the shape of the ellipse in terms of σ_1 and σ_2 :



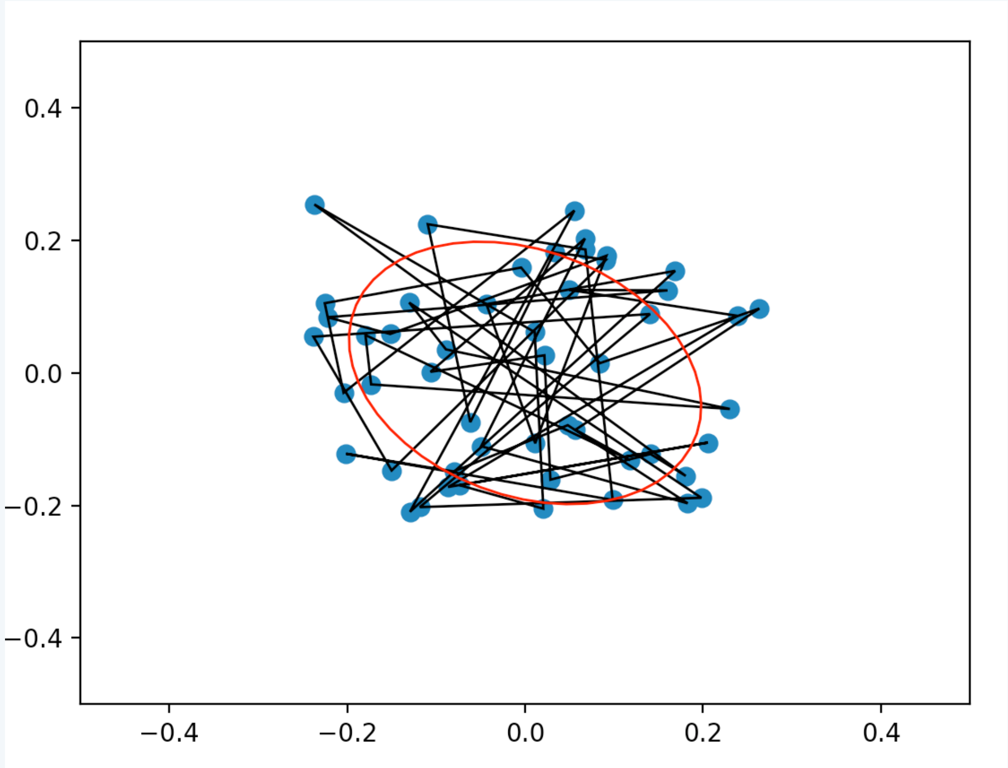
Backwards in Time?

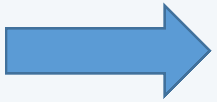
Generating Polygons $P_{-1}, P_{-2}, \ldots$

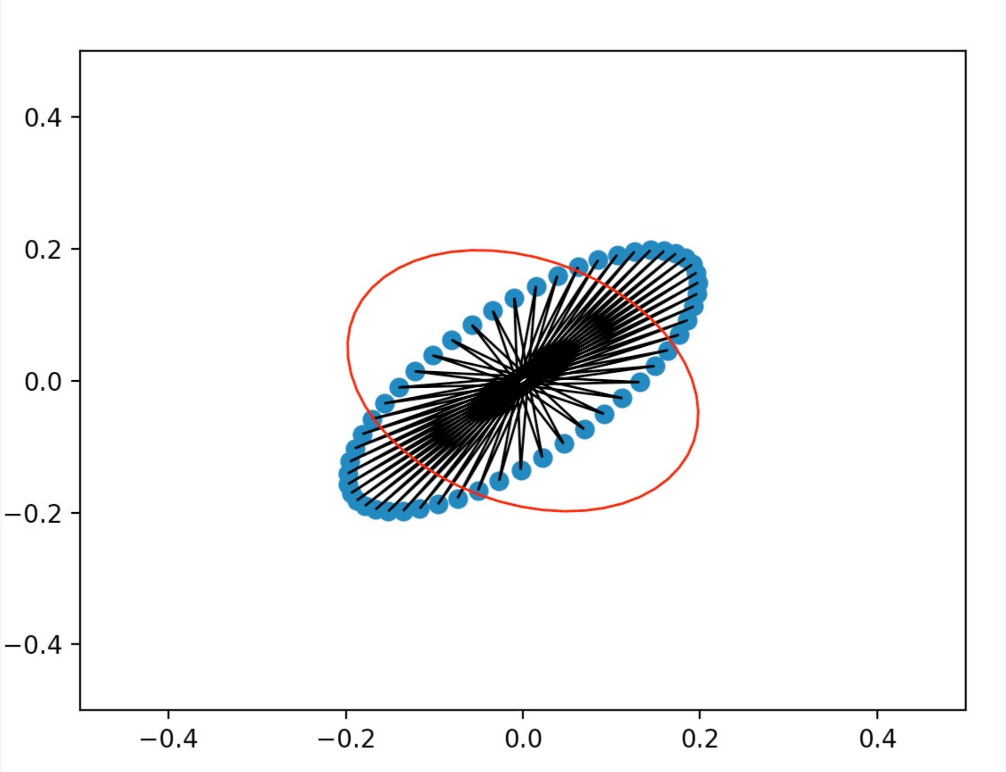
```
x = rand(n,1); x = x - mean(x); x = x/norm(x)
y = rand(n,1); y = y - mean(y); y = y/norm(y)
for k = 1,2,...
    x = inv(M)*x; x = x/norm(x)
    y = inv(M)*y; y = y/norm(y)
end
```

$$M = \frac{1}{2}\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 \end{bmatrix} \qquad M^{-1} = \begin{bmatrix} 1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & 1 & -1 \\ -1 & 1 & 1 & -1 & 1 \\ 1 & -1 & 1 & 1 & -1 \\ -1 & 1 & -1 & 1 & 1 \end{bmatrix}$$

Example: n=51 points. WTH?







Does M^{-1} always exist? No! **Only for odd n.**

[For intuition on the non-existence of the inverse for even n, chose 4 random points (assume time step t) and try to infer 4 points (time t-1) which lead to the 4 points at time t. After being successful, try to find another 4 points which can lead to the same 4 points. After being successful, what does this tell us about the inverse of M?]

An important note of caution

Algorithm 2

Input: Unit 2-norm n-vectors $x^{(0)}$ *and* $y^{(0)}$ *whose components sum to zero.*

Display $\mathcal{P}_0 = \mathcal{P}(x^{(0)}, y^{(0)})$.

for $k = 1, 2, \ldots$

% Compute $\mathcal{P}_k = \mathcal{P}(x^{(k)}, y^{(k)})$ from $\mathcal{P}_{k-1} = \mathcal{P}(x^{(k-1)}, y^{(k-1)})$

$$f = M_n x^{(k-1)}, \quad x^{(k)} = f / \| f \|_2$$

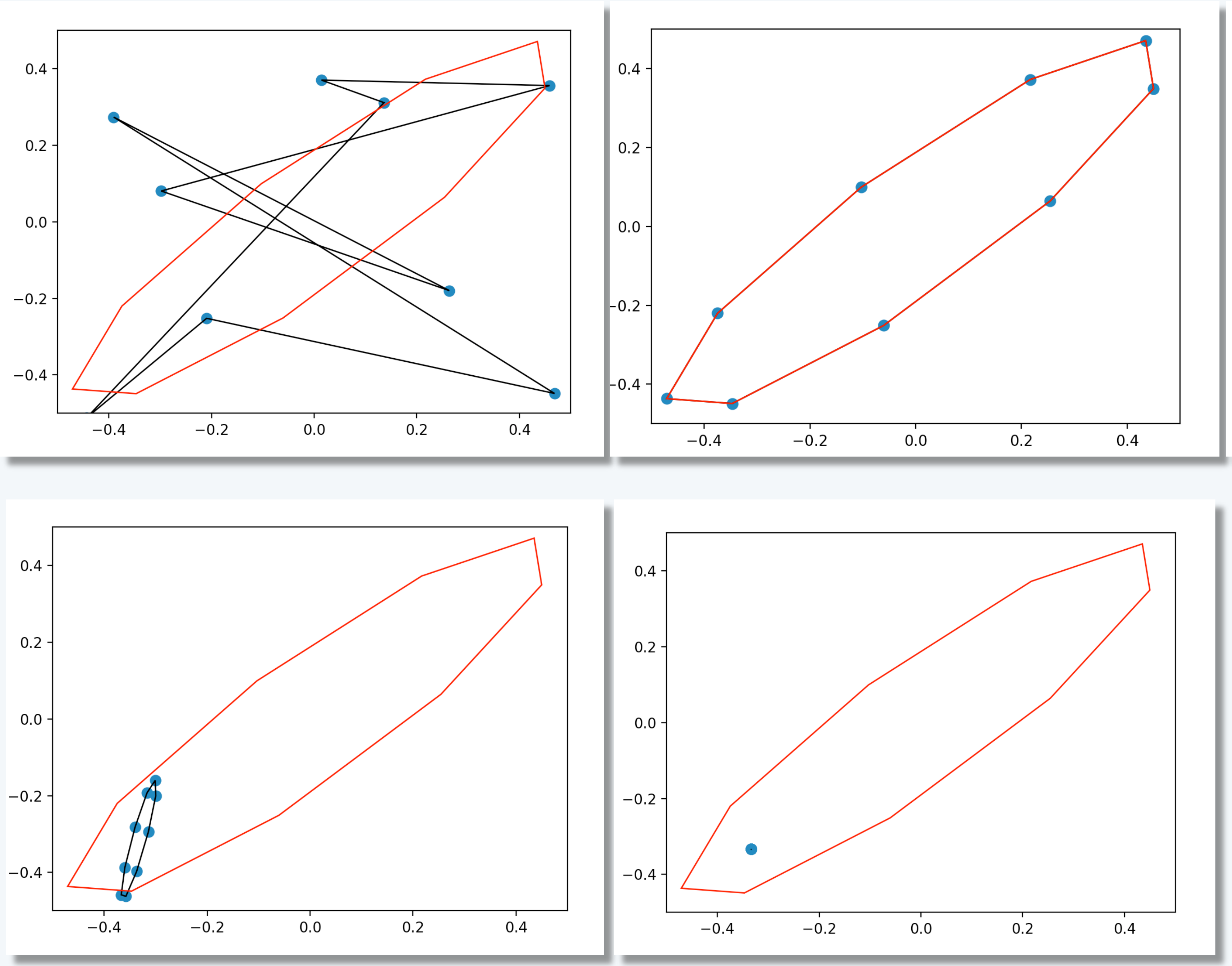
$$g = M_n y^{(k-1)}, \quad y^{(k)} = g / \| g \|_2$$

Display $\mathcal{P}_k$.

end

We learned that the centroids do not change (by mathematical proof!)
When submitting a solution to the lab exercise, please (try to) check: is this true in your simulation?
If not, how to “fix” it? (-> enforce a mean(x)=0 and mean(y)=0 after each step? Is this a fix? No! But at least better.)

Numerical issues can be very complicated i.) to detect and ii.) to resolve!!



Iteration with 9 points at step 0, 400, 500, 600, and 700, without enforcing the mean to be 0. The simulation was done using M^{10} for increased simulation speed.

Numerical Issues!