MAA704, Numerical methods for computing eigenvalues and eigenvectors.

Christopher Engström

December 17, 2013

Lecture 10: Overview

Today we will look at:

- The Power method.
- Revisiting the QR-factorization and Gram-Schmidt process
- QR-method.

Why do we need numerical methods?

I think everyone have seen how time consuming it is to find the eigenvalues and eigenvectors analytically by hand.

- In the same way using Gauss-elimination or the characteristic polynomial is computationally slow and not very useful in many real-world systems where we often work with very large matrices.
- We will look at some different methods for different kinds of matrices or depending on if you need all eigenvalues or if it's enough with one.

Why do we need numerical methods?

If we need to find eigenvalues or eigenvectors of a matrix we should first ask ourselves:

- Do we already know something about the eigenvalues or eigenvectors we can use? Maybe we know some eigenvalues or have a good approximation already.
- Do we seek only eigenvalues or eigenvectors? Do we need all or only one or a few eigenvalues/eigenvectors?
- Do we have a small or moderately sized matrix or do we have a large or very large matrix requiring specialized methods?
- Do we have a full matrix (few zero elements) or sparse matrix with mostly zeros.
- Do our matrix have any useful property or structure we could use? such as being positive definite or a bandmatrix.

Power method

The Power method is a very simple algorithm for finding the eigenvalue λ_1 with the largest absolute value and it's corresponding eigenvector.

- The method works by iterating $\mathbf{x}_{k+1} = \frac{A\mathbf{x}_k}{||A\mathbf{x}_k||}$.
- Where we assume there is only 1 eigenvalue on the spectral radius and that A is diagonalizable.
- If $\lambda_1 \neq 0$ then $\frac{A\mathbf{x}_k}{||A\mathbf{x}_k||}$ approaches a vector parallel to the eigenvector of λ_1.
- We get the eigenvalue λ_1 from the relation $A^k\mathbf{x} \approx \lambda_1 A^{k-1}\mathbf{x}$ when k is large.

Power method

For the Power method:

- $$\mathbf{x}_{k+1} = \frac{\mathrm{A}\mathbf{x}_k}{||\mathrm{A}\mathbf{x}_k||}$$
- It's easy to see that if the Matrix $A^\top$ is a stochastic matrix this can be seen as iterating a Markov chain.
- This also motivates why we need there to be only one eigenvalue on the spectral radius.
- The relative error can be shown to be in the order of $|\lambda_2/\lambda_1|^k$.

Power method

We give a short example looking at the following matrix:

- $$A = \begin{bmatrix} -2 & 1 & 1 \\ 3 & -2 & 0 \\ 1 & 3 & 1 \end{bmatrix}$$
- We start with $\mathbf{x} = [1\ 1\ 1]^\top$ and iterate:

Power method

We give a short example looking at the following matrix:

- $$\mathbf{A}\mathbf{x} = \begin{bmatrix} -2 & 1 & 1 \\ 3 & -2 & 0 \\ 1 & 3 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 5 \end{bmatrix}$$
- $||\mathbf{A}\mathbf{x}|| = \sqrt{26}$
- $\mathbf{x}_1 = \dfrac{\mathbf{A}\mathbf{x}}{||\mathbf{A}\mathbf{x}||} = [0, 1, 5]/\sqrt{26}$

Power method

Continuing for the next iteration we get:

- $$A\mathbf{x}_1 = \begin{bmatrix} -2 & 1 & 1 \\ 3 & -2 & 0 \\ 1 & 3 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 5 \end{bmatrix} \frac{1}{\sqrt{26}} = \begin{bmatrix} 6 \\ -2 \\ 8 \end{bmatrix} \frac{1}{\sqrt{26}}$$
- $\mathbf{x}_2 = \dfrac{A\mathbf{x}_1}{||A\mathbf{x}_1||} = [6, -2, 8]/\sqrt{104}$

Power method

If we continue for a couple of more iterations we get:

- $\mathbf{x}_2 = [0.59, -0.17, 0.78]^\top$.
- $\mathbf{x}_3 = [-0.25, 0.91, 0.33]^\top$.
- $\mathbf{x}_4 = [0.41, -0.61, 0.67]^\top$.
- $\mathbf{x}_7 = [-0.28, 0.85, -0.44]^\top$.
- $\mathbf{x}_{10} = [0.28, -0.79, 0.55]^\top$.
- We see that it seems to converge, but note that it could take some time.
- For reference the true eigenvector is $\mathbf{x}_{true} = [0.267, -0.802, 0.525]$.

Power method

There are a couple of things to consider when using the Power method:

- If there is more than one eigenvalue on the spectral radius or if it's multiplicity is larger than one, the method won't converge.
- Since the error is of the order $|\lambda_2/\lambda_1|^k$, if λ_2 is close to λ_1 the convergence can be very slow.
- We only get the (in absolute value) largest eigenvalue and corresponding eigenvector.
- There are however ways to handle some of these weaknesses as we will see later.

Power method

While the method does have a couple of negatives, it does have a couple of very strong points as well.

- In every step we make only a single vector-matrix multiplication.
- If A is a sparse matrix the iterations will be fast even for extremely large matrices.
- Since we do no matrix decomposition (as we do in most other methods) we do not "destroy" the sparcity of the matrix when calculating the matrix decomposition.
- Otherwise common methods using matrix decomposition used on a sparse matrix generally destroys the sparcity making them unsuitable for these kind of matrices.

Power method

But how can we handle the problems found earlier?

- We will start by looking at something called the inverse power method.

Inverse power method

We consider the matrix invertible matrix A with one eigenpair λ, x.

- We then have $\mathsf{A}\mathbf{x} = \lambda\mathbf{x}$ from the definition. This gives:
-

$$\mathsf{A}\mathbf{x} = \lambda\mathbf{x} \Leftrightarrow (\mathsf{A}-\mu\mathsf{I})\mathbf{x} = (\lambda-\mu)\mathbf{x} \Leftrightarrow \mathbf{x} = (\mathsf{A}-\mu\mathsf{I})^{-1}(\lambda-\mu)\mathbf{x}$$

$$\Leftrightarrow (\lambda-\mu)^{-1}\mathbf{x} = (\mathsf{A}-\mu\mathsf{I})^{-1}\mathbf{x}$$

- We see that $(\lambda-\mu)^{-1}, \mathbf{x}$ is an eigenpair of $(\mathsf{A}-\mu\mathsf{I})^{-1}$.

Inverse power method

If we instead iterate using this new matrix $(\mathsf{A} - \mu\mathsf{I})^{-1}$.

- The method becomes:

$$\mathbf{x}_{k+1} = \frac{(\mathsf{A} - \mu\mathsf{I})^{-1}\mathbf{x}_k}{||(\mathsf{A} - \mu\mathsf{I})^{-1}\mathbf{x}_k||}$$

  .

- This will converge to the eigenvector of the dominant eigenvalue of $(\mathsf{A} - \mu\mathsf{I})^{-1}$.
- We get the dominant eigenvalue $(\lambda - \mu)^{-1}$ for the λ closest to μ.

Inverse power method

If we look at the error we get:

- $$\left| \frac{(\lambda_{(\text{second closest to } \mu)} - \mu)^{-1}}{(\lambda_{(\text{closest to } \mu)} - \mu)^{-1}} \right|^k$$
- $$\Leftrightarrow \left| \frac{(\lambda_{(\text{closest to } \mu)} - \mu)}{(\lambda_{(\text{second closest to } \mu)} - \mu)} \right|^k$$

Inverse power method

We can easy see that if μ is a good approximation of an eigenvalue of A the error:

$$\left| \frac{(\lambda_{(\text{closest to } \mu)} - \mu)}{(\lambda_{(\text{second closest to } \mu)} - \mu)} \right|^k$$

will be very close to zero and the method converge very fast. Often in only one or a couple iteration.

Inverse power method

By using different μ we can find different eigenvectors of A, not only the dominant one.

- However we still have the problem with eigenvalues with multiplicity larger than one, or more than one with the same absolute value.
- The Inverse power method is often used when you have a good approximation of an eigenvalue and need to find an approximation of the corresponding eigenvector.
- The inverse power method requires us to either solve a linear system or calculate the inverse of the matrix $(\mathrm{A} - \mu)$. This makes it unsuitable for huge sparse matrices where the basic power algorithm works best.

Application PageRank.

Pagerank is the method originally designed by Google to rank homepages in their search engine in order to show the most relevant pages first.

The principle of PageRank

Nodes who are linked to by many other nodes and/or other important nodes are more important than others.

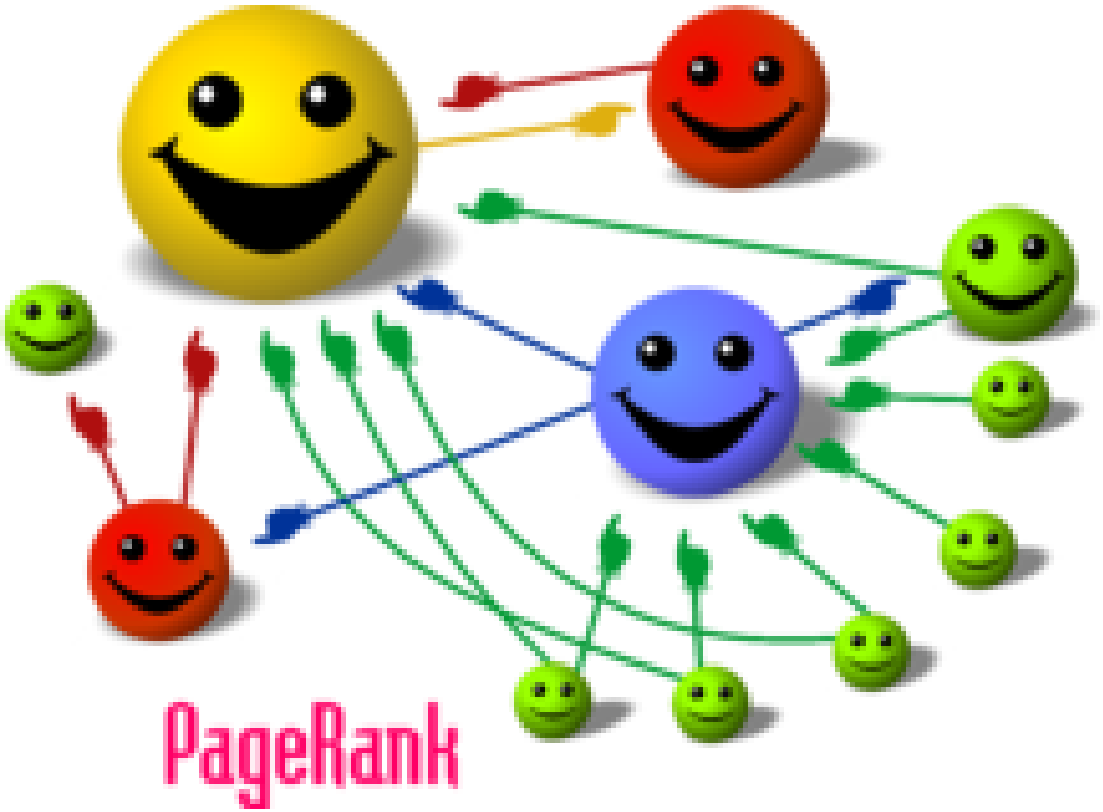


image by Felipe Micaroni Lalli

Application PageRank.

- In our definition of PageRank we look at webpages as nodes in a graph, and a link from one page to another as a link in the graph between the corresponding pages.
- We do not allow any webpage to link to itself.
- Given the adjacency matrix for this system we then make one modification to it. We weight the matrix by normalizing the sum of every non-zero row such that it sums to one. This is our matrix A.
- We call webpages that link to no other page dangling pages (or dangling nodes), we assume these webpages to link to all pages(even themselves) rather than none.

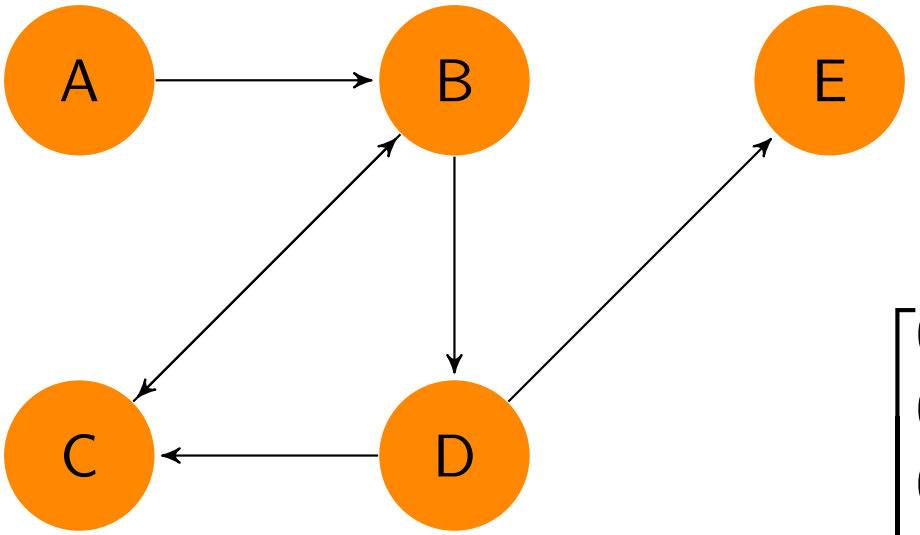
PageRank: definition

PageRank is the eigenvector to the dominant eigenvalue with value one to the matrix:

- $$\mathrm{M} = c(\mathrm{A} + \mathbf{g}\mathbf{v}^\top)^\top + (1 - c)\mathbf{v}\mathbf{e}^\top$$
- Where the $0 < c < 1$ is a scalar, usually $c = 0.85$.
- A is a non negative square matrix whose rows all sum to one.
- $\mathbf{v}$ is a non negative weightvector with the sum of all elements equal to one.
- $\mathbf{g}$ is a vector with elements equal to zero for all elements except those corresponding to dangling nodes which are equal to one.
- $\mathbf{e}$ is the vector with all elements wqual to one.

PageRank

Let us consider the following small system with corresponding adjacency matrix:



$$\begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

PageRank

We assume all elements of v equal such that we give no node a higher weight.

- Using the matrix for the graph we get the matrix $(A + \mathbf{g}\mathbf{v}^\top)$ by normalizing every row to one and add the change on the last row we get:
- $$(A + \mathbf{g}\mathbf{v}^\top) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 1/2 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1/2 & 0 & 1/2 \\ 1/5 & 1/5 & 1/5 & 1/5 & 1/5 \end{bmatrix}$$

PageRank

Using this and we get our matrix
$\mathrm{M} = c(\mathrm{A} + \mathbf{g}\mathbf{v}^\top)^\top + (1-c)\mathbf{v}\mathbf{e}^\top$ as:

- $$\begin{bmatrix} 0 & c & 0 & 0 & 0 \\ 0 & 0 & c/2 & c/2 & 0 \\ 0 & c & 0 & 0 & 0 \\ 0 & 0 & c/2 & 0 & c/2 \\ c/5 & c/5 & c/5 & c/5 & c/5 \end{bmatrix}^\top + \frac{1-c}{5} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$
- Using $c = 0.85$ and calculating the eigenvector with eigenvalue one of M gives pagerank $p = [0.0532, 0.3356, 0.2790, 0.1958, 0.1364]^\top$.

PageRank

- However we note that the matrix M has been constructed such that it's a non-negative matrix.
- And it also seems reasonable to assume it's irreducible through our vector $\mathbf{v}$. In fact unless some nodes have a zero-weight in $\mathbf{v}$ we will have a positive matrix.
- We also have the sum of every row equal to one, so we can say M is actually a stochastic matrix.
- This gives us a more intuitive way to describe pagerank looking at is from a Markov chain perspective.

PageRank

Consider a internet surfer surfing the web.

- The internet surfer starts surfing at a random page.
- Then with probability c it follows any of the links from the current page and with probability $1 - c$ it starts over at a new random page.
- The internet surfer then continue repeatedly either follows a link from the current page or starts over at a new random page.
- The PageRank of a node (page) can then be seen as the probability that the internet surfer is at that page after a long time.

PageRank

But how do we calculate this eigenvector? To put in perspective $2008 - 2009$ both Google and bling reported indexing over 1.000.000.000.000 webpages.

- That's about 10^{24} elements in our matrix M.
- However the matrix A is extremely sparse, since a page usually only link to a couple of other pages.
- If we have a sparse matrix, we could use the Power method, while other methods such as the QR-method or gauss-elimination would prove impossible.
- Even if M is technically a full matrix, the multiplication $\mathbf{x}_{n+1} = \mathrm{M}\mathbf{x}_n$ can be done fast by calculating $\mathrm{A}\mathbf{x}_n$ first.

Part 2: QR-method

QR-factorization and the QR-method in part two.

QR-method

Often it's not enough to find only one eigenvalue or eigenvector but you need all of them.

- In generall, when you use a program to find eigenvalues and eigenvectors it's nearly always the QR-method you use.
- But before we look at the method we will take another look at the QR-factorization and why this is relevant.

QR-decomposition

Theorem

Every $n \times m$ matrix A have a matrix decomposition

$$A = QR$$

. *where*

- R *is a* $n \times m$ *upper triangular matrix..*
- Q *is a* $n \times n$ *unitary matrix.*

QR-decomposition

A unitary matrix is a square matrix satisfying:

- $\mathrm{U}^H\mathrm{U} = \mathrm{U}\mathrm{U}^H = \mathrm{I}$
- In other words $\mathrm{U}^{-1} = \mathrm{U}^H$.
- Where U^H is the conjugate transpose $(\mathrm{A}^H)_{ij} = \overline{(\mathrm{A})_{ji}}$.

QR-decomposition

We let U be a unitary matrix, some important properties are:

- The condition number $\kappa(\mathrm{U}) = 1$, making for very well-conditioned and robust systems or algorithms.
- Since $\mathrm{U}^{-1} = \mathrm{U}^{H}$ we can very easily find the inverse of U.
- This means similarity transforms $\mathrm{U}^{H}\mathrm{AU}$ involving U is both easy to compute and robust.

QR-decomposition

If A is a square real matrix then Q is an orthogonal matrix $Q^\top = Q^{-1}$.

- QR-factorization plays an important role in the QR-method commonly used to calculate eigenvalues and eigenvectors.
- The QR-factorization can be computed using for example Householder transformations or the Gram-Schmidt process.

Gram-Schmidt for finding a QR-decomposition

We remember the Gram-Schmidt process for finding a orthonormal basis, this can be used to find a QR-decomposition as well.

- Given a $n \times m$ matrix A with columns $A_{.1}, A_{.2}, \ldots, A_{.n}$.
- We form the columns of our matrix Q one at a time:
- We let $Q_{,1} = A_{.1}/||A_{.1}||$ and create

$$A'_{.i} = A_{.i} - Q_{,1}^H A_{.i} Q_{,1}, \;\; i = 2, \ldots, n$$

- We then create $Q_{,2}$ from $A'_{.2}$ and repeat untill we have a full basis or we run out of columns.
- Q is then our matrix Q in the QR-decomposition and we can easily get R by multiplication $\mathrm{R} = \mathrm{Q}^H \mathrm{A}$.
- We recognize this as the modified Gram-Schmidt process with normalization during the algorithm.

Gram-Schmidt for finding a QR-decomposition

We consider the matrix

$$A = \begin{bmatrix} 1 & -1 & 2 & 3 \\ 2 & 0 & 2 & 1 \\ 2 & 2 & 3 & 2 \end{bmatrix}$$

- This gives:

$$Q_{.1} = \frac{A_{.1}}{||A_{.1}||} = \frac{1}{3}\begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix}, \quad A'_{.2} = A_{.2} - Q_{.1}^H A_{.2} Q_{.1} = \frac{1}{3}\begin{bmatrix} -4 \\ -2 \\ 4 \end{bmatrix}$$

- $$A'_{.3} = \frac{1}{3}\begin{bmatrix} 2 \\ -2 \\ 1 \end{bmatrix}, \quad A'_{.4} = \frac{1}{3}\begin{bmatrix} 6 \\ -3 \\ 0 \end{bmatrix}$$

Gram-Schmidt for finding a QR-decomposition

- Continuing with the next column we get:

$$Q_{.2} = \frac{A'_{.2}}{||A'_{.2}||} = \frac{1}{3}\begin{bmatrix} -2 \\ -1 \\ 2 \end{bmatrix}, \quad A''_{.3} = A'_{.3} - Q_{.2}^H A'_{.3} Q_{.2} = \frac{1}{3}\begin{bmatrix} 2 \\ -2 \\ 1 \end{bmatrix}$$

- $$A''_{.4} = \frac{1}{3}\begin{bmatrix} 4 \\ -4 \\ 2 \end{bmatrix}$$

- $$Q_{.3} = \frac{A''_{.3}}{||A''_{.3}||} = \frac{1}{3}\begin{bmatrix} 2 \\ -2 \\ 1 \end{bmatrix}$$

- We now have a full basis and can stop since $Q_{.4}$ must be zero.

Gram-Schmidt for finding a QR-decomposition

For the QR-decomposition $A = QR$ we then get the matrices:

- $$Q = (Q_{.1}, Q_{.2}, Q_{.3}) = \frac{1}{3}\begin{bmatrix} 1 & -2 & 2 \\ 2 & -1 & -2 \\ 2 & 2 & 1 \end{bmatrix}$$
- And
$$R = Q^H A = \frac{1}{3}\begin{bmatrix} 9 & 3 & 12 & 9 \\ 0 & 6 & 0 & -3 \\ 0 & 0 & 3 & 6 \end{bmatrix}$$

QR-method

The QR-method is an iterative algorithm for computing the eigenvalues and eigenvectors of a matrix A using QR-factorization.

- We start by assigning $A_0 := A$
- In every step we then compute the QR-decomposition $A_k = Q_k R_k$.
- We then form $A_{k+1} = R_k Q_k = Q_k^H Q_k R_k Q_k = Q_k^H A_k Q_k = Q_k^{-1} A_k Q_k$
- So all A_k are similar and therefor have the same eigenvalues.
- When the method converge A_k will be a triangular matrix and therefor have it's eigenvalues on the diagonal.

QR-method

The main advantage of the QR-method is that it's not only fast when implemented with some necessary modifications.

- It's also very stable in that a small change in A only gives a small change in the computed eigenvalues.
- This is one of the main reason it's used over other similar methods such as for example methods using the LUP decomposition.

QR-method

The QR-method can be seen as a variation of the Power method but instead of only one vector, we work with a complete basis of vectors, using QR-decomposition to normalize and orthogonalize in every step.

- As with the Power method, the QR-method in it's basic form have a couple of things to consider.
- Computing the QR-decomposition of a matrix is expensive, and we need to do it in every step.
- The convergence can be very slow.
- Like the Power method we have problems with complex and non simple eigenvalues.

QR-method

- To solve the first problem the matrix A is nearly always first brought to upper Hessenberg form.
- This will both decrease the cost of the QR-factorizations as well as improve the convergence

Hessenberg matrix

A matrix A is a (upper) Hessenberg matrix if all elements $a_{ij} = 0, i > j + 1$. In other words A is of the form:

$$\mathrm{A} = \begin{bmatrix} \bigstar & \bigstar & \bigstar & \cdots & \bigstar & \bigstar & \bigstar \\ \bigstar & \bigstar & \bigstar & \cdots & \bigstar & \bigstar & \bigstar \\ 0 & \bigstar & \bigstar & \cdots & \bigstar & \bigstar & \bigstar \\ 0 & 0 & \bigstar & \cdots & \bigstar & \bigstar & \bigstar \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & \bigstar & \bigstar & \bigstar \\ 0 & 0 & 0 & \cdots & 0 & \bigstar & \bigstar \end{bmatrix}$$

Hessenberg matrix

Theorem

Every square matrix A have a matrix decomposition

$$\mathrm{A} = \mathrm{U}^H \mathrm{H} \mathrm{U}$$

where

- U *is a unitary matrix.*
- H *is a Hessenberg matrix.*

Hessenberg matrix

Since U is a unitary matrix, A and H are similar and we can work with H rather than A in the QR-method.

- This is useful for a couple of reasons.
- Calculating the QR-factorization of a Hessenberg matrix is much faster than for a general matrix.
- Since multiplication of a Hessenberg matrix with a triangular matrix (both in the same direction) is itself a Hessenberg matrix. We need only do the Hessenberg factorization once.
- Since a Hessenberg matrix is nearly triangular, its convergence towards the triangular matrix will be improved.
- Since U is unitary, it's condition number is 1, numerical erros when finding H is not amplified.

QR-method

However the convergece can still be very slow even when working with the Hessenberg form.

- We remember how we could solve this in the Power method by shifting the eigenvalues of the matrix.
- We can do the same in the QR-method using $(A_k - \sigma I)$.
- Rewriting as before we get
  $A_{k+1} = Q_k^H(A_k - \sigma I)Q_k + \sigma I = Q_k^H(A_k)Q_k$.
- We see that A_k, A_{k+1} is still similar and therefor will have the same eigenvalues.
- As in the Power method we want σ to be as close as possible to one of the eigenvalues.
- The diagonal elements of the hessenberg matrix prove to give a good approximation.

QR-method

- Since the diagonal elements of A_k should give increasingly accurate approximation of the eigenvalues we can change our shift to another eigenvalue or improve the current one between iterations as well.
- As fast as a subdiagonal element is close to zero, we have either found an eigenvalue if it's one of the outer elements.
- Or we can divide the problem in two smaller ones since we would have:
- $A = \begin{bmatrix} A_{1,1} & A_{1,2} \\ 0 & A_{2,2} \end{bmatrix}$

QR-method

- In practice an implicit method using multiple shifts at the same time is usually used.
- This also solves the problems with complex eigenvalues in real matrices.
- The Implicit version is also called the Francis algorithm.

QR-method

To find the eigenvectors

- The eigenvectors can either be found by multiplying $Q = Q_0 Q_1 \dots Q_k$ with the initial unitary matrix to bring the problem to Hessenberg from.
- They can also be found through the use of a form of inverse iteration.
- It's also possible to use for example Gauss elimination using the eigenvalues on our original matrix.

QR-method

We give a summary of the modified method using a single shift:

1. Compute the Hessenberg form ($A = U^H H U$) using for example Householder transformations.
2. Assign $A_0 = H$. Then in every step, choose a shift σ as one of the diagonal elements of A_k.
3. Compute the $Q_k R_k$ factorization of $(A_k - \sigma I)$.
4. Compute $A_{k+1} = R_k Q_k + \sigma I$
5. Iterate $2 - 4$ until A_k is a triangular matrix, optionally dividing the problem in two whenever a subdiagonal element becomes zero.