Jacobi's rotation method

- We want to diagonalize A using similarity transf., or in other words, find S such that $S^T A S = D$
- Will do it by a series of sim. transf $S_1, S_2, \dots, S_M$

$$A$$
$$S_1^T A S_1$$
$$S_2^T S_1^T A S_1 S_2$$
$$\vdots$$
until we get
$$S_M^T \dots S_2^T S_1^T A S_1 S_2 \dots S_M \approx D$$

- Then we have $S \approx S_1 S_2 \dots S_M$
  - S contains eigenvectors of A
  - D contains eigenvalues of A

- Notation in our algorithm: $\boxed{A^{(m+1)} = S_m^T A^{(m)} S_m}$
  - Start from $A^{(1)}$

$$A^{(2)} = S_1^T A^{(1)} S_1$$
$$A^{(3)} = S_2^T A^{(2)} S_2$$
$$\vdots$$
until $A^{(M)} \approx D$ ← eigenvalues

$$A^{(m)} = \begin{bmatrix} a_{11}^{(m)} & a_{12}^{(m)} & \dots \\ a_{21}^{(m)} & a_{22}^{(m)} & \dots \\ \vdots & & \end{bmatrix}$$

- $\boxed{R^{(m+1)} = S_m R^{(m)}}$
  - Start from $R^{(1)} = I$

$$R^{(2)} = S_1 R^{(1)}$$
$$R^{(3)} = S_2 R^{(2)}$$
$$\vdots$$
until $R^{(M)} = S_{M-1} R^{(M-1)} = S_{M-1} S_{M-2} \dots S_1 \approx S$ ← eigenvectors!

- S_m matrices are rotation matrices

$$S_m = \begin{array}{cc} & \begin{array}{ccc} & k & \ell \\ & \downarrow & \downarrow \end{array} \\ \begin{array}{c} \\ k \rightarrow \\ \\ \ell \rightarrow \\ \\ \end{array} & \begin{bmatrix} 1 & & & & & \\ & \ddots & & & & \\ & & \cos\theta & \cdots & \sin\theta & \\ & & \vdots & \ddots & \vdots & \\ & & -\sin\theta & \cdots & \cos\theta & \\ & & & & & \ddots \; 1 \end{bmatrix} \end{array}$$

4x4 examples

$$S_m = \begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$ Rotation in (x_1, x_2) plane

$$S_m = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & 0 & \sin\theta \\ 0 & 0 & 1 & 0 \\ 0 & -\sin\theta & 0 & \cos\theta \end{bmatrix}$$ Rotation in (x_2, x_4) plane

- Here S_m is defined as clockwise rotation to match Morten's lecture notes

  Then S_m^T is a counterclockwise rotation

- <u>Note</u>: S_m is fully specified by (k, ℓ, θ)

  (Never need to construct the full S_m)

Algorithm

1) • Choose tolerance ε, e.g. $\varepsilon = 10^{-8}$

• Let $A^{(1)} = A$

• Let $R^{(1)} = I$

2) Find indices (k,l) of max off-diagonal element in A

3) While $|a_{kl}^{(m)}| > \varepsilon$

3.1) Compute $$\tau = \frac{a_{ll}^{(m)} - a_{kk}^{(m)}}{2a_{kl}^{(m)}}$$

Notation:
$\sin\theta \equiv s$
$\cos\theta \equiv c$
$\tan\theta \equiv t$

3.2) Compute $\tan\theta$, $\cos\theta$, $\sin\theta$ (t,c,s):

• $t = -\tau \pm \sqrt{1+\tau^2}$

Choose solution that gives smallest t, i.e. smallest angle θ

So: if $\tau > 0$, use $t = -\tau + \sqrt{1+\tau^2} = \dfrac{1}{\tau + \sqrt{1+\tau^2}}$

if $\tau < 0$, use $t = -\tau - \sqrt{1+\tau^2} = \dfrac{-1}{-\tau + \sqrt{1+\tau^2}}$

• $c = \dfrac{1}{\sqrt{1+t^2}}$

• $s = ct$

If $a_{kl}^{(m)} = 0$:
$c = 1$
$s = 0$
$t = 0$

[Now we know S_m, since we have $k, l, \cos\theta, \sin\theta$]

3.3) Transform current A matrix,

$$A^{(m)} \to A^{(m+1)} = S_m^T A^{(m)} S_m$$

by updating elements:

- $a_{kk}^{(m+1)} = a_{kk}^{(m)} c^2 - 2 a_{kl}^{(m)} c s + a_{ll}^{(m)} s^2$
- $a_{ll}^{(m+1)} = a_{ll}^{(m)} c^2 + 2 a_{kl}^{(m)} c s + a_{kk}^{(m)} s^2$
- $a_{kl}^{(m+1)} = 0$
- $a_{lk}^{(m+1)} = 0$

$\Big\}$ This was the requirement we used to determine $\tan\theta$, i.e. that these elements should be transf. to 0.

- For all $i \neq k, l$:
  - $a_{ik}^{(m+1)} = a_{ik}^{(m)} c - a_{il}^{(m)} s$
  - $a_{ki}^{(m+1)} = a_{ik}^{(m+1)}$
  - $a_{il}^{(m+1)} = a_{il}^{(m)} c + a_{ik}^{(m)} s$
  - $a_{li}^{(m+1)} = a_{il}^{(m+1)}$

Must keep separate copies of some elements, i.e. don't use the new $a_{ik}^{(m+1)}$ here

3.4) Update the overall rotation matrix,

$$R^{(m)} \to R^{(m+1)} = S_m R^{(m)}$$

by updating elements:

- <u>For all i:</u>
  - $r_{ik}^{(m+1)} = r_{ik}^{(m)} c - r_{il}^{(m)} s$
  - $r_{il}^{(m+1)} = r_{il}^{(m)} c + r_{ik}^{(m)} s$

Again, make sure to use the correct value here!

3.5) Find the (k, l) indices of the <u>new</u> max off-diag. element

[While loop returns to 3)]

- Look at 2x2 case to see where the algorithm comes from

  - Symmetric A $(a_{21} = a_{12})$

$$A^{(1)} = \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} \\ a_{21}^{(1)} & a_{22}^{(1)} \end{bmatrix}$$

  - We want $A^{(2)}$ $\begin{bmatrix} a_{11}^{(2)} & 0 \\ 0 & a_{22}^{(2)} \end{bmatrix}$

  - So need transf. S_1 that gives $a_{12}^{(2)} = 0$ $(k=1, l=2)$

- Let $S_1 = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \equiv \begin{bmatrix} c & s \\ -s & c \end{bmatrix}$

- Write out $A^{(2)} = S^T A^{(1)} S$

$$\begin{bmatrix} a_{11}^{(2)} & a_{12}^{(2)} \\ a_{12}^{(2)} & a_{22}^{(2)} \end{bmatrix} = \begin{bmatrix} c & -s \\ s & c \end{bmatrix} \begin{bmatrix} a_{11}^{(1)} & a_{12}^{(1)} \\ a_{12}^{(1)} & a_{22}^{(1)} \end{bmatrix} \begin{bmatrix} c & s \\ -s & c \end{bmatrix}$$

- We get:

  - $a_{11}^{(2)} = a_{11}^{(1)} c^2 - 2a_{12}^{(1)} cs + a_{22}^{(1)} s^2$
  - $a_{22}^{(2)} = a_{22}^{(1)} c^2 + 2a_{12}^{(1)} cs + a_{11}^{(1)} s^2$
  - $a_{12}^{(2)} = \left(a_{11}^{(1)} - a_{22}^{(1)}\right) cs + a_{12}^{(1)} (c^2 - s^2)$

- <u>Require</u> $a_{12}^{(2)} = 0$

$$\Rightarrow \left(\frac{a_{11}^{(1)} - a_{22}^{(1)}}{a_{12}^{(1)}}\right) cs + c^2 - s^2 = 0 \qquad (*)$$

- Notation:
$$\tan\theta \equiv t = \frac{s}{c}$$
$$\tau \equiv \frac{a_{22} - a_{11}}{2a_{12}}$$

- Then (*) $\Rightarrow$
$$s^2 + 2\tau cs - c^2 = 0$$

- Divide by c^2:
$$\boxed{t^2 + 2\tau t - 1 = 0}$$
2nd ord. eq. for t

- Solutions:
$$t = -\tau \pm \sqrt{1 + \tau^2}$$
[These choices of $\tan\theta$ (so angle θ) ensures that $a_{12}^{(2)} = 0$]

- Compute:
$$c = \frac{1}{\sqrt{1+t^2}}$$
$$s = ct$$

- Can now compute $a_{11}^{(2)}$ and $a_{22}^{(2)}$

[No need to compute $a_{12}^{(2)}$ or $a_{21}^{(2)}$ — they are 0 by our choice of θ]

$\Rightarrow$ Have the new
$$A^{(2)} = \begin{bmatrix} a_{11}^{(2)} & 0 \\ 0 & a_{22}^{(2)} \end{bmatrix}$$
Eigenvalues of A

- Eigenvectors:
$$R^{(2)} = S_1 R^{(1)} = S_1 I = \left[\begin{pmatrix} c \\ -s \end{pmatrix} \begin{pmatrix} s \\ c \end{pmatrix} \right]$$
Eigenvectors of A

Problems

- For $N \times N$ matrix we need many iterations, because S_m also affect other elements along rows/colums k and l
- Can end in situation where element a_{ij} that have prev. been set to $a_{ij} = 0$ is transformed back to $a_{ij} \neq 0$ at later iteration