GOLUB-REINSCH ALGORITHM

$$[A]_{mxn}\ m > n$$

- Bidiagonalization of matrix A
  $A = QBP^T \qquad B \rightarrow upper\ Bidiagonal$
- $B = \begin{bmatrix} \tilde{B} \\ 0 \end{bmatrix}$
- $B = \tilde{B}$
- Diagonalization of B $\qquad B = X\Sigma Y^T$
- $U = QX$
- $V^T = (PY)^T$
- $A = U\Sigma V^T$ SVD OF A

Implicitly shifted QR algorithm

- Computes a sequence of upper Biadiagonal matrices B_i
- Uses the Wilkinson shift μ
- i increaces $B \rightarrow Diagonal$

4X4 EXAMPLE

$$B = \begin{pmatrix} d_1 & f_2 & & \\ & d_2 & f_3 & \\ & & d_3 & f_4 \\ & & & d_4 \end{pmatrix}$$

- Determine the Wilkinson shift

$$\begin{pmatrix} {d^2}_3 + {f^2}_3 & d_3 f_4 \\ d_3 f_4 & {d^2}_4 + {f^2}_4 \end{pmatrix}$$

- Givens Matrix $G_1 = G(1,2;\theta_1)$

$$tan\theta_1 = \frac{({B^T}_i B_i)_{12}}{\mu - ({B^T}_i B_i)_{11}} = \frac{d_1 f_2}{\mu - {d^2}_1}$$

$$\begin{pmatrix} c & s \\ -s & c \end{pmatrix} \cdot \begin{pmatrix} d_1 f_2 \\ \mu - {d^2}_1 \end{pmatrix} = \begin{pmatrix} * \\ 0 \end{pmatrix}$$

- Compute BG_1

$$BG_1 = \begin{pmatrix} * & * & & \\ {\color{red}*} & * & * & \\ & & * & * \\ & & & * \end{pmatrix}$$

- Zero out ${\color{red}*}$ and find P_1

$$P_1 B G_1 = \begin{pmatrix} * & * & {\color{red}*} & \\ & * & * & \\ & & * & * \\ & & & * \end{pmatrix}$$

- Zero out $\color{red}{*}$ and find G_2

$$P_1 B G_1 G_2 = \begin{pmatrix} * & * & & \\ & * & * & \\ & \color{red}{*} & * & * \\ & & & * \end{pmatrix}$$

- Zero out $\color{red}{*}$ and find P_2

$$P_2 P_1 B G_1 G_2 = \begin{pmatrix} * & * & & \\ & * & * & \color{red}{*} \\ & & * & * \\ & & & * \end{pmatrix}$$

- Zero out * and find P_3

$$P_3 P_2 P_1 B G_1 G_2 = \begin{pmatrix} * & * & & \\ & * & * & \\ & & * & * \\ & & * & * \end{pmatrix}$$

Finally:

- Zero out * and find G_3

$$P_3 P_2 P_1 B G_1 G_2 G_3 = \begin{pmatrix} * & * & & \\ & * & * & \\ & & * & * \\ & & & * \end{pmatrix}$$

- Itterate

$$f_i \rightarrow 0$$
$$d_i \rightarrow singular\ values$$

Implicit zero shift QR algorithm

- $\mu = 0$
- Entry (1,2) will be zero
- This zero will propagate through the rest of the algorithm
- Increase precision

PSEUDO CODE

$oldc = 1$

$g = d_1$

$p = f_1$

$for\ i = 1\ to\ n - 1$

$\quad [c, s, r] = ROT(g, p)$

$\quad if\ (i \neq 1)\ then$

$\quad\quad f_{i-1} = olds * r$

$\quad end\ if$

$\quad g = oldc * r$

$\quad p = d_{i+1} * s$

$\quad h = d_{i+1} * c$

$\quad [c, s, r] = ROT(g, p)$

$\quad d_i = r$

$\quad if\ (i \neq n - 1)\ then$

$\quad\quad p = f_{i+1}$

$end\ if$

$oldc = c$

$olds = s$

$end\ for$

$f_{n-1} = h * s$

$d_{n-1} = h * c$