The First Algorithms to Compute the SVD

Walter Gander, ETH

The Third International Workshop on Matrix Computations
Lanzhou University

April 15 – April 19, 2022

The Singular Value Decomposition

- Given $A \in \mathbb{R}^{m \times n}$, $m \geq n$
- There exist
  - singular vectors:
    $U = [\mathbf{u}_1, \ldots, \mathbf{u}_n] \in \mathbb{R}^{m \times n}$ and $V = [\mathbf{v}_1, \ldots, \mathbf{v}_n] \in \mathbb{R}^{n \times n}$
    $U^\top U = V^\top V = I_n$ orthogonal
  - singular values:
    $\Sigma = \mathrm{diag}\,(\sigma_1, \ldots, \sigma_n) \in \mathbb{R}^{n \times n}$ diagonal
    with $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0$ such that

$$A = U \Sigma V^\top = \sigma_1 \mathbf{u}_1 \mathbf{v}_1^\top + \sigma_2 \mathbf{u}_2 \mathbf{v}_2^\top + \cdots + \sigma_n \mathbf{u}_n \mathbf{v}_n^\top$$

The Three Heros

Christian Reinsch *1934

TU München

Gene H. Golub (1932–2007)

Stanford

William M. Kahan *1933

Berkeley

The Famous Golub-Reinsch Algorithm

Handbook Series Linear Algebra

Singular Value Decomposition and Least Squares Solutions*

Contributed by

G. H. Golub** and C. Reinsch

1. Theoretical Background

1.1. Introduction

Let A be a real $m \times n$ matrix with $m \geqq n$. It is well known (cf. [4]) that

$$A = U \Sigma V^T \tag{1}$$

where

$$U^T U = V^T V = V V^T = I_n \quad \text{and} \quad \Sigma = \operatorname{diag}(\sigma_1, \ldots, \sigma_n).$$

- Developed 1967 by Reinsch, published 1970 in Numerische Mathematik
- Cited 4093 times as of February 28, 2022
- How was the collaboration? Did Golub and Reinsch meet in 1967?
- Comments in "Milestones in Matrix Computations, 2007"

Why not Compute the SVD via $A^\top A$?

- Eigenvalues of $A^\top A$ are $\lambda_k = \sigma_k^2 \implies \sigma_k = \sqrt{\lambda_k}$
- Numerically bad algorithm. Golub-Kahan (1965):

> *But the calculation of $A^\top A$ using ordinary floating point arithmetic does serious violence to the smaller singular values ...*

Example: 10×7 section of Hilbert matrix

```
m=10; format long
A=hilb(m); A=A(:,1:7);
E=eig(A'*A); E=sqrt(E(7:-1:1));
[E svd(A)]
format short e
RelError= (E-svd(A))./svd(A)
```

```
ans =                                                   RelError =
   1.703422789369242   1.703422789369242                  1.3035e-16
   0.303861884355195   0.303861884355195                  9.1343e-16
   0.027332449735276   0.027332449735275                  2.7291e-14
   0.001576339549570   0.001576339549570                  2.5517e-13
   0.000060439432051   0.000060439432540                 -8.0919e-09
   0.000001483441024   0.000001483519405                 -5.2835e-05
   0.000000020984384   0.000000020211193                  3.8256e-02
```

The Golub-Kahan Algorithm, 1965

- $A \in \mathbb{R}^{m \times n}$ with $m \geq n$
- Consider the augmented matrix $\tilde{A} = \begin{pmatrix} 0 & A \\ A^\top & 0 \end{pmatrix} \in \mathbb{R}^{(m+n)\times(m+n)}$
- Eigenvalues of $\tilde{A}$ are $\lambda_k = \pm\sigma_k$ and $m - n$ zeros.
- Bidiagonalize $A = P \begin{pmatrix} J \\ 0 \end{pmatrix} Q^\top$ with P, Q orthogonal, J bidiagonal

$$\tilde{A} = \begin{pmatrix} P & 0 \\ 0 & Q \end{pmatrix} \begin{pmatrix} 0 & \begin{pmatrix} J \\ 0 \end{pmatrix} \\ (J^\top, 0) & 0 \end{pmatrix} \begin{pmatrix} P^\top & 0 \\ 0 & Q^\top \end{pmatrix}$$

Similarity transformation

Further Simplifications

- Consider $J \in \mathbb{R}^{n \times n}$ (eliminate zero EV)

$$\Longrightarrow \quad \bar{A} = \begin{pmatrix} 0 & J \\ J^\top & 0 \end{pmatrix} \in \mathbb{R}^{2n \times 2n}, \qquad \lambda_k(\bar{A}) = \pm\sigma_k$$

- Adjust signs in J (similarity transformation)

$$J = \begin{pmatrix} a_1 & b_1 & & \\ & \ddots & \ddots & \\ & & a_{n-1} & b_{n-1} \\ & & & a_n \end{pmatrix} \quad \text{may assume } a_k, b_k \geq 0$$

Transform to Tridiagonal Matrix

- Let P be the permutation matrix for

$$P\begin{pmatrix} 1 \\ 2 \\ 3 \\ 4 \\ \vdots \\ 2n-1 \\ 2n \end{pmatrix} = \begin{pmatrix} n+1 \\ 1 \\ n+2 \\ 2 \\ \vdots \\ n+n \\ n \end{pmatrix} \Longrightarrow S = P\begin{pmatrix} 0 & J \\ J^\top & 0 \end{pmatrix}P^\top = \begin{pmatrix} 0 & a_1 & & & & & \\ a_1 & 0 & b_1 & & & & \\ & b_1 & 0 & a_2 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & a_{n-1} & 0 & b_{n-1} & \\ & & & & b_{n-1} & 0 & a_n \\ & & & & & a_n & 0 \end{pmatrix}$$

- The eigenvaues of the tridiagonal matrix S are $\lambda_k(S) = \pm\sigma_k(A)$
- Problem is reduced to compute the non-negative eigenvalues of a special symmetric tridiagonal matrix

augmented

Computing the Eigenvalues of S

Golub-Kahan (1965):

> *There are a number of methods for obtaining the eigenvalues of a tridiagonal symmetric matrix. One of the most accurate and effective methods is to use Sturm sequences; an ALGOL program is given by Wilkinson.*[a]

procedure *tridibisection1(c,b,n,gu,go,t,gamma) result: (w,norm,m1) ;*
value *n,gu,go,t,gamma;*
integer *n,t,m1 ;*
real *gu,go,gamma,norm;*
array *c,b,w;*
comment *c* is the diagonal and *b* the sub-diagonal of a symmetric tridiagonal matrix of order *n*. The number *m1* of eigenvalues lying between *gu* and *go* is determined and these eigenvalues are then computed in decreasing order by the method of bisection, and stored as the vector *w* of order *m1*. *t* is the number of bisection steps and *norm* is the infinity norm of the tridiagonal matrix, *gamma* the square of the relative machine precision;

---

[a] J.H. Wilkinson, Calculation of the eigenvalues of a symmetric tridiagonal matrix by the method of bisection, Numerische Mathematik 4, 362–367 (1962).

Translation of tridibisection1 to MATLAB

```
function [w,NormInf,m1]=tribisection(c,b,gu,go)
% TRIBISECTION is a translation from Wilkinson's tridibisection1 from
% ALGOL to MATLAB by W. Gander, May 2019.
% Changes in ALGOL tridibisection:
              % eliminate t=number of bisection steps,
              % instead compute EV to machine precision
gamma=eps^2;  % square of machine precision
n=length(c);  % no need to be in parameter list.
              % function [q1,a1]=sturmssequence(c,p,lambda)
              % with parameters, no global variables
b(n)=0;       % add zero for same length as c
NormInf=abs(c(1))+abs(b(1));
for i=2:n,  l=abs(b(i-1))+abs(c(i))+abs(b(i));  if l>NormInf, NormInf=l; end, end
if nargin==2                    % added by W. Gander:
   go=1.5*NormInf; gu=-go;      % if no interval specified
end                             % compute all eingenvalues
if nargin==3                    % compute all lambda>=gu
   go=1.5*NormInf;
end
```

Algorithm of Golub-Kahan (1965)

```
function q=SVDGolubKahan1(A)
% SVDGOLUBKAHAN1 singular values by the first Golub Kahan algorithm.
%   Applying Wilkinson's TRIBISECTION to matrix S (2n x 2n)
[m,n]=size(A); if n>m, A=A'; [m,n]=size(A);end
[a,b]=Bidiagonalize(A);               % Householder bidiagonalization
sk=zeros(2*n-1,1);                    % form tridiagonal matrix (3.3)
k=1:n; sk(2*k-1)=abs(a(k));           % S on page 213
k=1:n-1; sk(2*k)=abs(b(k+1));
q=tribisection(zeros(2*n,1),sk,0); % compute all nonnegative EV
```

Example

- Let $A = \begin{pmatrix} B & 2B \\ 3B & -B \end{pmatrix}$

  where $B = \begin{pmatrix} 5 & -1 & -1 & 6 & 4 & 0 \\ -3 & 1 & 4 & -7 & -2 & -3 \\ 1 & 3 & -4 & 5 & 4 & 7 \\ 0 & 4 & -1 & 1 & 4 & 5 \\ 4 & 2 & 3 & 1 & 6 & -1 \\ 3 & -3 & -5 & 8 & 0 & 2 \\ 0 & -1 & -4 & 4 & -1 & 3 \\ -5 & 4 & -3 & -2 & -1 & 7 \\ 3 & 4 & -3 & 6 & 7 & 7 \end{pmatrix} \in \mathbb{R}^{9\times 6}$

- A is a (18×12) matrix with rank=6

Results: we obtain the 12 singular values: `testRankDeficient`

With Wilkinson's `tribisection`	Matlab's SVD
72.265903120085341	72.265903120085312
49.630339183086065	49.630339183086065
44.288698552845858	44.288698552845858
36.427417335191990	36.427417335192004
30.416324106579545	30.416324106579548
25.017401012828763	25.017401012828763
0.000000000000008	0.000000000000006
0.000000000000006	0.000000000000005
0.000000000000005	0.000000000000003
0.000000000000003	0.000000000000002
0.000000000000002	0.000000000000002
0.000000000000001	0.000000000000001

Results are good, but large computational effort with bisection

Lost CS-Report

- CS 73 is missing in Stanford's on-line collection
- I got a paper copy from Åke Björck
- We reconstructed the ALGOL procedure
- We found an ALGOL compiler by Jan van Katwijk
- We can test the ALGOL Businger procedure!

[8]

With best wishes, Gene

CS 73

LEAST SQUARES, SINGULAR VALUES AND MATRIX APPROXIMATIONS

BY

GENE H. GOLUB

AN ALGOL PROCEDURE FOR COMPUTING THE

SINGULAR VALUE DECOMPOSITION

BY

PETER BUSINGER

TECHNICAL REPORT NO. CS73
JULY 31, 1967

COMPUTER SCIENCE DEPARTMENT
School of Humanities and Sciences
STANFORD UNIVERSITY

Algorithm of Golub-Businger 1967

- Stanford Report CS73: Part 1: Theory by Gene Golub.
  Part2: ALGOL procedure by Peter Businger

- Start as before with tridiagonal matrix

$$K_0 = \begin{pmatrix} 0 & a_1 & & & & & \\ a_1 & 0 & b_1 & & & & \\ & b_1 & 0 & a_2 & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & a_{n-1} & 0 & b_{n-1} & \\ & & & & b_{n-1} & 0 & a_n \\ & & & & & a_n & 0 \end{pmatrix}$$

- Define $(\gamma_1, \gamma_2, \ldots, \gamma_{2n-1}) := (a_1, b_1, a_2, \ldots, b_{n-1}, a_n)$

ALGOL Procedure by Businger

```
procedure singular values decomposition
   (a, m, n, u desired, vt desired, eta) results: (sigma, u, vt) ;
value m, n, u desired, vt desired, eta ;
real array a, sigma, u, vt ;
integer m, n ;
boolean u desired, vt desired ;
real eta ;
comment Householder's and the QR method are used to find all singular
   values sigma[i], (i=1, 2, ..., n) of the given matrix a[1:m, 1:n],
   (m≥n). The orthogonal matrices u[1:m, 1:m] and vt[1:n, 1:n] which
   effect the singular values decomposition a=u sigma vt are computed
   individually depending on whether u desired or vt desired. The input
   parameter eta is the relative machine precision ;
```

Theory by Golub

- Compute $\lambda(K_0)$ by QR-Algorithm with implicit shifts
- Since $\lambda(K_0)$ occur in pairs, consider QR-decomposition of $(K_i - s_i I)(K_i + s_i) = K_i^2 - s_i^2 I = M_i R_i$
- Note that if

$$K = \begin{bmatrix} 0 & \gamma_1 & 0 & 0 & 0 \\ \gamma_1 & 0 & \gamma_2 & 0 & 0 \\ 0 & \gamma_2 & 0 & \gamma_3 & 0 \\ 0 & 0 & \gamma_3 & 0 & \gamma_4 \\ 0 & 0 & 0 & \gamma_4 & 0 \end{bmatrix}$$

- then

$$K^2 - s^2 I = \begin{bmatrix} {\gamma_1}^2 - s^2 & 0 & \gamma_1\gamma_2 & 0 & 0 \\ 0 & {\gamma_1}^2 + {\gamma_2}^2 - s^2 & 0 & \gamma_2\gamma_3 & 0 \\ \gamma_1\gamma_2 & 0 & {\gamma_2}^2 + {\gamma_3}^2 - s^2 & 0 & \gamma_3\gamma_4 \\ 0 & \gamma_2\gamma_3 & 0 & {\gamma_3}^2 + {\gamma_4}^2 - s^2 & 0 \\ 0 & 0 & \gamma_3\gamma_4 & 0 & {\gamma_4}^2 - s^2 \end{bmatrix}$$

is pentadiagonal with 3 nonzero diagonals

- Define first Givens-reflection Z_1 to annihilate the (3,1)-element $\gamma_1\gamma_2$

$$Z_1(K^2 - s^2 I)$$

We need a Givens reflection Z_1 of the form

$$Z_p = \begin{bmatrix} 1 & & & & & & & & \\ & \ddots & & & & & & & \\ & & 1 & & & & & & \\ & & & \cos\Theta_p & 0 & \sin\Theta_p & & & \\ & & & 0 & 1 & 0 & & & \\ & & & \sin\Theta_p & 0 & -\cos\Theta_p & & & \\ & & & & & & 1 & & \\ & & & & & & & \ddots & \\ & & & & & & & & 1 \end{bmatrix} \begin{matrix} \\ \\ \\ \leftarrow p \\ \leftarrow p+1 \\ \leftarrow p+2 \\ \\ \\ \\ \end{matrix} \quad (1)$$

First Transformation

- Apply first Givens-reflection Z_1 (defined to annihilate the (3,1)-element $\gamma_1\gamma_2$ in $Z_1(K_0^2 - s^2 I)$ to K_0.

- $K_1 = Z_1 K_0 Z_1 = Z_1 \begin{pmatrix} 0 & a_1 & & & \\ a_1 & 0 & b_1 & & \\ & b_1 & 0 & a_2 & \\ & & \ddots & \ddots & \ddots \end{pmatrix} Z_1$

- Generates bulge X in $K_1 = \begin{pmatrix} 0 & \hat{a}_1 & 0 & X & \\ \hat{a}_1 & 0 & \hat{b}_1 & & \\ 0 & \hat{b}_1 & 0 & \hat{a}_2 & \\ X & & \ddots & \ddots & \ddots \end{pmatrix}$

- Bulge X is chased down by subsequent Givens-reflections $K_i = Z_i K_{i-1} Z_i, \quad i = 2, \ldots, n$, preserving zero diagonal

Shift Strategy

- Shift s_i^2 used for $(K_i - s_i I)(K_i + s_i) = K_i^2 - s_i^2 I$ is square of eigenvalue of bottom 4×4 matrix closer to γ_{t-1}^2

$$M = \begin{bmatrix} 0 & \gamma_{t-3} & 0 & 0 \\ \gamma_{t-3} & 0 & \gamma_{t-2} & 0 \\ 0 & \gamma_{t-2} & 0 & \gamma_{t-1} \\ 0 & 0 & \gamma_{t-1} & 0 \end{bmatrix}$$

- Characteristic polynomial

$$\det(M - \lambda I) = \lambda^4 - \left({\gamma_{t-3}}^2 + {\gamma_{t-2}}^2 + {\gamma_{t-1}}^2\right)\lambda^2 + {\gamma_{t-3}}^2{\gamma_{t-1}}^2$$

Solution of $\det(M - \lambda I) = 0$

$$\lambda^2 = \frac{({\gamma_{t-3}}^2 + {\gamma_{t-2}}^2 + {\gamma_{t-1}}^2) \pm \sqrt{({\gamma_{t-3}}^2 + {\gamma_{t-2}}^2 + {\gamma_{t-1}}^2)^2 - 4{\gamma_{t-3}}^2{\gamma_{t-1}}^2}}{2}$$

- Compare with the ALGOL statements:

```
g0:=gamma[t-1]^2+gamma[t-2]^2+gamma[t-3]^2 ;
g1:=gamma[t-1]^2*gamma[t-3]^2 ;
g2:=0.5*(g0+sqrt(g0^2-4.0*g1)) ;
g3:=g1/g2 ;
kappa:=if abs(gamma[t-1]^2-g2)<abs(gamma[t-1]^2-g3) then g2 else g3 ;
```

- Carefully solved for larger solution g2, then smaller g3 by relation of Vieta
- Shift `kappa` is solution closer to γ_{t-1}^2
- But no check whether the discriminant is negative

Observations

- Symmetry and zero-diagonal of K_i is preserved
- The algorithm works only on the subdiagonal $(\gamma_1, \gamma_2, \ldots, \gamma_{2n-1}) = (a_1, b_1, a_2, \ldots, b_{n-1}, a_n)$
- If $b_{n-1} \to 0$ then $\lambda = a_n$ is eigenvalue, deflate $n := n - 2$

Example

DemosALGOL/Bsp7p1.alg

$$J = \begin{pmatrix} 1 & 2 & 0 & 0 \\ 0 & 1 & 4 & 0 \\ 0 & 0 & 1 & 6 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \qquad \Longrightarrow \qquad K_0 = \begin{pmatrix} 0 & 1 & & & & & & \\ 1 & 0 & 2 & & & & & \\ & 2 & 0 & 1 & & & & \\ & & 1 & 0 & 4 & & & \\ & & & 4 & 0 & 1 & & \\ & & & & 1 & 0 & 6 & \\ & & & & & 6 & 0 & 1 \\ & & & & & & 1 & 0 \end{pmatrix}$$

Example cont.

- Main loop in ALGOL procedure starts with label `inspect`
- `s`, `t` define current block in subdiagonal γ
- `sigma` contains computed eigenvalues
- we added a variable to `count` iterations

```
    ......
inspect:
   ; comment scan for lower block limit t ;
   count:=count+1;
   outstring(1,"\n\ninspect "); outinteger(1,count); outstring(1,"\n");
   outvec("sigma",sigma,n);
   gamma[s]:=gamma[t]:=0.0 ;
   for i:=t-2 while abs(gamma [i])<= epsilon do
   begin comment pick up computed value ;
    ......
```

Example Results

Bsp7p1.alg

```
inspect 1
gamma
-1.0000e0  2.0000e0  1.0000e0  4.0000e0  1.0000e0  -6.0000e0 -1.0000e0  0
s and t:  0 8 , zero shift

inspect 2
gamma
2.4083e0  1.4948e0  3.8669e0  1.5422e0  5.8850e0  -3.1456e-3  1.8246e-2  0
s and t:  0 8 , zero shift

inspect 3
gamma
3.4918e0  2.1196e0  3.8644e0  3.4473e0  4.0614e0  4.8402e-8  1.8246e-2  0
determine shift  kappa=3.329227607e-4
s and t:  0 8

inspect 4
gamma
4.5505e0  2.1040e0  5.1932e0  1.5866e0  2.3191e0  -1.0213e-23  1.8246e-2  0
```

- since $|\gamma_6| \leq \text{eta}\|K_0\|_\infty$, $\Longrightarrow$ $\sigma_4 = 1.8246e{-2}$, deflate $t := t - 2$

The next iterations give:

```
s and t:  0 6 , zero shift
inspect 5
gamma
5.4666e0  1.9829e0  4.6208e0  3.5192e-1  2.1696e0  0
determine shift , kappa=4.672568312
s and t:  0 6
inspect 6
gamma
6.0855e0  9.5698e-1  4.1695e0  -1.9043e-4  2.1599e0  0
determine shift , kappa=4.665413228
s and t:  0 6
inspect 7
gamma
6.2029e0  3.5366e-1  4.0906e0  8.1443e-15  2.1599e0  0
determine shift , kappa=4.665413228
s and t:  0 6
inspect 8
gamma
6.2184e0  1.2520e-1  4.0803e0  2.3665e-30  2.1599e0  0
```

- $\sigma_3 = 2.1599$, $t := t - 2$

Finally

```
gamma
6.2184e0  1.2520e-1  4.0803e0   0
s and t:  0 4 , zero shift
inspect 9
gamma
6.2202e0  -5.3871e-2  4.0792e0  0
determine shift , kappa=16.63775489
s and t:  0 4
inspect 10
gamma
6.2206e0  -2.0816e-17  4.0789e0  0
```

Thus one step with zero shift followed by one with shift 16.63775489 yields $\gamma_2 = -2.0816e - 17 \approx 0 \implies \sigma_1$ and σ_2

```
sigma
6.220651007345815e0
4.078940413139840e0
2.159956765331501e0
1.824617112552942e-2
```

Check with MATLAB

```
>> J=[1     2     0     0
      0     1     4     0
      0     0     1     6
      0     0     0     1];
```

```
>> svd(J)
ans =
     6.220651007345817e+00
     4.078940413139843e+00
     2.159956765331501e+00
     1.824617112552942e-02
```

Rank Deficient Example

`beispiel1B.alg`

$$B = \begin{pmatrix} 5 & -1 & -1 & 6 & 4 & 0 \\ -3 & 1 & 4 & -7 & -2 & -3 \\ 1 & 3 & -4 & 5 & 4 & 7 \\ 0 & 4 & -1 & 1 & 4 & 5 \\ 4 & 2 & 3 & 1 & 6 & -1 \\ 3 & -3 & -5 & 8 & 0 & 2 \\ 0 & -1 & -4 & 4 & -1 & 3 \\ -5 & 4 & -3 & -2 & -1 & 7 \\ 3 & 4 & -3 & 6 & 7 & 7 \end{pmatrix}$$

- $A = \begin{pmatrix} B & 2B \\ 3B & -B \end{pmatrix}$
- Golub-Businger needs 18 steps for 12 σ_k
- good result!

```
sigma
7.226590312008532e1
4.963033918308604e1
4.428869855284583e1
3.642741733519196e1
3.041632410657953e1
2.501740101282876e1
3.445807702749013e-15
5.861665712052792e-15
2.351807372845660e-15
4.488399943021106e-15
2.802195408089914e-15
2.226465746728873e-15
```

Close Singular Values

- Bidiagonal matrix with 2 clusters, gap $1e{-}7$

b_{kk}	$b_{k,k+1}$	σ_k
1.614874172816116	9.264623902779769e-01	2.000000100000000
1.238486644745703	2.131595816650056e-07	2.000000000000000
1.926281858121494	4.598199463754764e-01	1.000000100000000
1.038269760777829		1.000000000000000

- Start:

$$\gamma = [-1.6148,\ 9.2646e{-}1,\ 1.2384e0,\ 2.1315e{-}7,\ 1.9262,\ -4.5981e{-}1,\ -1.0382,\ 0]$$

step	shift	γ_4	γ_6	step	shift	γ_2
0		2.1315e-7	4.5981e-1	6	zero	-8.5471e-8
1	zero	7.4224e-7	-1.2203e-1	7	4.000000011	-3.3341e-9
2	1.000000179	3.3742e-1	-7.4612e-9	8	4.000000011	-9.8840e-11
3	1.000000199	2.3202e-8	-1.0219e-11	9	3.999999993	1.5748e-12
4	1.000000198	1.5378e-15	-5.9455e-14	10	3.999999993	-2.5092e-14
5	1.000000199	1.0250e-22	-1.1886e-17	11	3.999999993	3.9981e-16

- deflate $t := t - 4$

linear convergence

Results

The computed σ_k are correct:

```
sigma
2.000000099999999e0
2.000000000000000e0
1.000000099999999e0
9.999999999999994e-1
```

Narrowing the Gap

clusterB.alg

- Consider bidiagonal matrix with 2 clusters, gap $1e{-}8$

b_{kk}	$b_{k,k+1}$	σ_k
1.614874124853175	9.264623389167206e-01	2.000000010000000
1.238486628039565	2.131595964078222e-08	2.000000000000000
1.926281841828408	4.598199397802367e-01	1.000000010000000
1.038269674236179		1.000000000000000

- The Golub-Businger program produces an infinite loop

Reason: after QR-step #9, shift is `NaN`, discriminant `g0^2-4.0*g1` is $-8.881784197e{-}16$ negative due to rounding errors

Multiple Singular Values

- Matrix with $\sigma_k = [2, 2, 1, 1]$. Golub-Businger needs 11 QR-steps for
.

b_{kk}	$b_{k,k+1}$	σ_k
1.546667895215945	9.673182260019585e-01	1.999999999999999e0
1.293102421137901	1.845276169487005e-15	1.999999999999999e0
1.984647769311140	2.136408344093210e-01	1.000000000000001e0
1.007735493887760		1.000000000000000e0

- Augment the multiplicity: $\sigma_k = [1, 1, 1, 2, 2, 2]$

diagonal	secondary diagonal
1.664426845302032	8.846508172580001e-01
1.200172696232285	2.323234527365937e-15
1.927953055087120	4.548199770714277e-01
1.037369657276030	1.265849009056839e-15
1.994732361709430	1.255160648047072e-01
1.002640774467638	

- Golub-Businger produces an infinite loop because when scanning

```
for i:=t-2 while abs(gamma [i])<= epsilon do
```

we have: `epsilon=5.664412841e-16` $<$ `abs(gamma[2])=7.7715e-16`.

Since `gamma[2]` remains constant the iteration never ends.

Wilkinson matrix

$$\begin{pmatrix} 100 & 1 & & & & & \\ 1 & 90 & 1 & & & & \\ & 1 & \ddots & \ddots & & & \\ & & \ddots & 80 & 1 & & \\ & & & 1 & 90 & 1 \\ & & & & 1 & 100 \end{pmatrix} \qquad \begin{array}{c} \text{computed } \sigma_k \\ \hline 3.000000082849189e1 \\ 2.999999917290396e1 \\ 2.000049662325264e1 \\ 1.999950657441164e1 \\ 1.009659543859793e1 \\ 9.900494253375482e0 \\ 1.970928910340472e-1 \end{array}.$$

- $n = 21$, augmented system $2n = 42$.
- After 27 QR-steps Golub-Businger computed 7 correct singular values
- However, next step discriminant is $-1.862645149e{-9}$

$\implies$ infinite loop

- **Conclusion**: The Golub-Businger program is not foolproof. If it works, it is very effective. But sometimes it fails.

The Springer Handbook Project in the Sixties:

- First attempt for building a software library
- Golub-Businger submitted their SVD procedure – was not accepted.
- Chr. Reinsch was in charge of testing the submissions:

  > *It was easy to find sample matrices with unsatisfactory convergence rates and in some cases the algorithm would not converge at all.*

- Reinsch developed his own SVD algorithm at the same time in 1967, independently of Golub-Businger
- F. L. Bauer decided to accept the algoritm of Reinsch in the Handbook under joint authorship Golub-Reinsch .

Handbook for
Automatic Computation

Edited by
F. L. Bauer · A. S. Householder · F. W. J. Olver
H. Rutishauser † · K. Samelson · E. Stiefel

Volume II

J. H. Wilkinson · C. Reinsch

Linear Algebra

Chief editor
F. L. Bauer

The Algorithm of Reinsch

- developed 1967, published 1970 in *Numerische Mathematik*

Numer. Math. 14, 403—420 (1970)

Handbook Series Linear Algebra

Singular Value Decomposition and Least Squares Solutions*

Contributed by

G. H. Golub** and C. Reinsch

1. Theoretical Background

1.1. Introduction

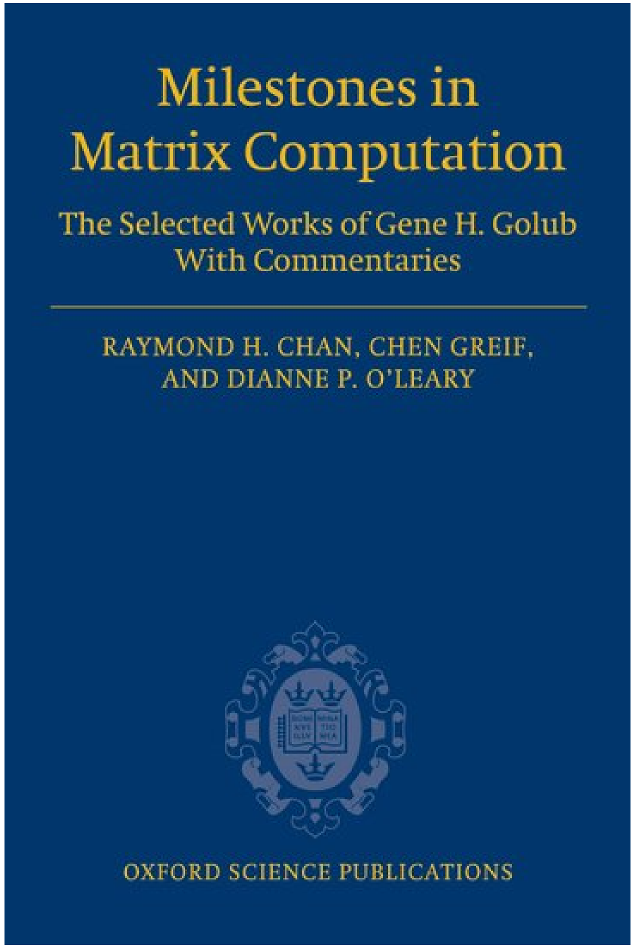
Let A be a real $m \times n$ matrix with $m \geqq n$. It is well known (cf. [4]) that

$$A = U \Sigma V^T \qquad (1)$$

Gene Golub's 75th Birthday Celebration 2007 in Stanford

Raymond H. Chan,
Chen Greif, and
Dianne P. O'Leary

- Compiled a book with selected papers of Gene Golub
- with commentaries of his coauthors
- This gave Reinsch opportunity to explain



The Algorithm of Reinsch

- Bidiagonalize $A = PBQ^\top$, $\lambda_k(B^\top B) = \sigma_k(B)^2$

$$B = \begin{pmatrix} q_1 & e_2 & & & \\ & q_2 & e_3 & & \\ & & \ddots & \ddots & \\ & & & \ddots & e_n \\ & & & & q_n \end{pmatrix}$$

- Consider applying the QR-Algorithm with implicit shift σ to

$$T = B^\top B = \begin{pmatrix} q_1^2 & q_1 e_2 & & & \\ q_1 e_2 & e_2^2 + q_2^2 & q_2 e_3 & & \\ & q_2 e_3 & \ddots & \ddots & \\ & & \ddots & e_{n-1}^2 + q_{n-1}^2 & q_{n-1} e_n \\ & & & q_{n-1} e_n & e_n^2 + q_n^2 \end{pmatrix}$$

- Use Wilkinson's shift σ: EV of lower 2×2 minor closer to T_{nn}

- First QR transformation by Givens rotation G_1 such that

$$\begin{pmatrix} c & s \\ -s & c \end{pmatrix}^\top \begin{pmatrix} q_1^2 - \sigma \\ q_1 e_2 \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix}.$$

- Applying G_1 and $G_1^\top$ to $T = B^\top B$ yields

$$G_1^\top B^\top B G_1 = \begin{pmatrix} x & x & x & & & \\ x & x & x & & & \\ x & x & x & x & & \\ & & x & x & x & \\ & & & \ddots & \ddots & \ddots \end{pmatrix},$$

- and subsequent transformations chase the bulge x till the tridiagonal form is restored

Clever Idea of Reinsch: Work with B alone!

- consider $BG_1 = \begin{pmatrix} x & x & & & & \\ x & x & x & & & \\ & & x & x & & \\ & & & x & x & \\ & & & & \ddots & \ddots \end{pmatrix}$

- chasing the bulge x by Givensrotations to restore bidiagonal form:

$$P_1^\top BG_1 = \begin{pmatrix} x & x & x & & & \\ & x & x & & & \\ & & x & x & & \\ & & & x & x & \\ & & & & \ddots & \ddots \end{pmatrix}, \; P_1^\top BG_1G_2 = \begin{pmatrix} x & x & & & & \\ & x & x & & & \\ & x & x & x & & \\ & & & x & x & \\ & & & & \ddots & \ddots \end{pmatrix}$$

- $\tilde{B}$ is again bidiagonal: $\tilde{B} = P_{n-1}^\top \cdots P_1^\top BG_1 \cdots G_{n-1}$

- Theorem: This transformation $B \to \tilde{B}$ is mathematically the same process as a QR-step for the tridiagonal matrix $T = B^\top B \to \tilde{B}^\top \tilde{B}$

Splitting $B = \begin{pmatrix} q_1 & e_2 & & & \\ & q_2 & e_3 & & \\ & & \ddots & \ddots & \\ & & & \ddots & e_n \\ & & & & q_n \end{pmatrix}$

- If $e_i = 0 \implies B$ splits in two bidiagonal matrices

$$B = \begin{pmatrix} B_1 & 0 \\ 0 & B_2 \end{pmatrix}, \quad \text{svd}(B) = \text{svd}(B_1) \cup \text{svd}(B_2).$$

  The singular values of B_1 and B_2 can be computed independently (even in parallel).

- If split for $i = n$, $e_n = 0 \implies B_2 = q_n$ and q_n is a singular value. Continue computations with B_1.

Cancellation

- If $q_i = 0 \implies$ one of the singular values $= 0$
- Split B using special Givens rotations $\begin{pmatrix} c & -s \\ s & c \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ r \end{pmatrix}$

$$G_{i,i+1}^{\top} \begin{pmatrix} q_1 & e_2 & & & & & \\ & \ddots & \ddots & & & & \\ & & q_{i-1} & e_i & & & \\ & & & 0 & e_{i+1} & & \\ & & & & q_{i+1} & \ddots & \\ & & & & & \ddots & e_n \\ & & & & & & q_n \end{pmatrix} = \begin{pmatrix} q_1 & e_2 & & & & & \\ & \ddots & \ddots & & & & \\ & & q_{i-1} & e_i & & & \\ & & & 0 & 0 & X & \\ & & & & \tilde{q}_{i+1} & \ddots & \\ & & & & & \ddots & e_n \\ & & & & & & q_n \end{pmatrix}.$$

- remove bulge X by further Givens rotations $G_{i,k}$, $k = i+2, \ldots, n$
- Because $e_{i+1} = 0$, the matrix splits again in two submatrices

Convergence

- Cannot expect $e_i = 0$ or $q_i = 0$, need a threshold to decide when zero.
  - Golub-Reinsch recommend (with ε=machine precision) $|e_{i+1}|, |q_i| \leq \varepsilon \max_i(|q_i| + |e_i|) = \varepsilon \|B\|_1,$
- Deflation:
  If e_n is negligible $\implies$ q_n is a singular value.
  Proceed iteration with submatrix of order $n - 1$.

Results for Rank Deficient Matrix A:

`beispiel1R.alg`

ALGOL Program Golub-Reinsch	Matlab's SVD
72.265903120085326	72.265903120085312
49.630339183086043	49.630339183086065
44.288698552845872	44.288698552845858
36.427417335191983	36.427417335192004
30.416324106579530	30.416324106579548
25.017401012828770	25.017401012828763
0.000000000000015	0.000000000000006
0.000000000000005	0.000000000000005
0.000000000000004	0.000000000000003
0.000000000000004	0.000000000000002
0.000000000000003	0.000000000000002
0.000000000000002	0.000000000000001

Golub-Reinsch needs 15 steps for these 12 σ_k

Close and Multiple Singular Values

2 clusters, gap $1e{-}8$

b_{kk}	$b_{k,k+1}$
1.614874124853175	9.264623389167206e-01
1.238486628039565	2.131595964078222e-08
1.926281841828408	4.598199397802367e-01
1.038269674236179	

Golub-Reinsch needs 4 steps

2.000000010000000

2.000000000000000

1.000000000000000

1.000000009999999

$\sigma_k = [1, 1, 1, 2, 2, 2]$

diagonal	secondary diagonal
1.666426845302032	8.846508172580001e-01
1.200172696232285	2.323234527365937e-15
1.927953055087120	4.548199770714277e-01
1.037369657276030	1.265849009056839e-15
1.994732361709430	1.255160648047072e-01
1.002640774467638	

Golub-Reinsch needs 6 steps to get

9.999999999999996e-1

1.000000000000000

1.999999999999999

2.000000000000000

2.000000000000000

1.000000000000001

Wilkinson matrix

$$\begin{pmatrix} 100 & 1 & & & & \\ 1 & 90 & 1 & & & \\ & 1 & \ddots & \ddots & & \\ & & \ddots & 80 & 1 & \\ & & & 1 & 90 & 1 \\ & & & & 1 & 100 \end{pmatrix}$$

computed σ_k	
$1.000995057466245e2$	$4.999999999999966e1$
$1.000995057466244e2$	$4.000000000069121e1$
$9.000049342558835e1$	$3.999999999930925e1$
$9.000049342558833e1$	$3.000000082849191e1$
$8.000000082709604e1$	$2.999999917290397e1$
$8.000000082709600e1$	$2.000049662325266e1$
$7.000000000069067e1$	$1.999950657441164e1$
$7.000000000069064e1$	$1.009659543859792e1$
$6.000000000000036e1$	$9.900494253375477e0$
$6.000000000000031e1$	$1.970928910340454e-1$
$5.000000000000036e1$	

Golub-Reinsch correctly computes all σ_k in 41 steps

Remarks

- Reinsch has created a wonderful foolprof algorithm for computing the SVD
- Golub became coauthor though he did not work on this algorithm
- Is this plagiarism? Why did Bauer made him a coauthor?
- the answer is:

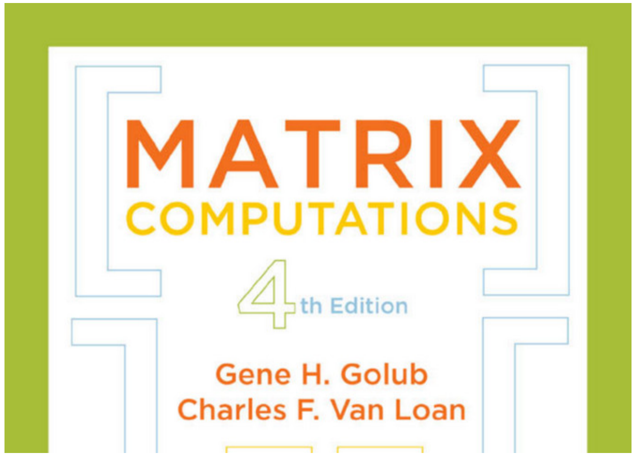
  Golub-Businger and Golub-Reinsch compute the same iterates!

- The data structure is different, but both algorithms work on the same elements of the bidiagonal matrix.
- When using the same shifts, the iterates of Golub-Businger and Golub-Reinsch produces the same numbers

This may explain the decision of F. L. Bauer to make Golub a coauthor

Reinsch is Missing Credit for his SVD-Algorithm

- in the famous textbook Golub-van Loan the Reinsch SVD algorithm is explained in details
- But reference is completely wrong.
  page 489: *A preferable method for computing the SVD is described by Golub and Kahan (1965). Their technique finds U and V simoultaneously by implicitely applying the symmetric QR algorithm to* $A^\top A$
  This is completely wrong!
- Reinsch is not mentioned at all!



4th edition 2013

Acknowledgments

I wish to thank

- Christian Reinsch TUM, for many discussions and many e-mails. He is a real genius.
- Jan van Katwijk, J.vanKatwijk@gmail.com Lazy Chair Computing. Jan did a wonderful work to produce this jff-algol program which compiles the Algol sources to C code. Doing so he revives old ALGOL procedures and made it possible for us to experiment.
- Johann Joss, my old fellow student from ETH. He is a gifted mathematician and computer scientist. With his help we got rid of all problems and finally we had running ALGOL procedures.
- Å. Björck. I am indebted to Åke for sending me a paper copy of the Stanford CS Report #73. This paper by Golub and Businger is not well known and it is hard to get hold of this report

References

- G. Golub and W. Kahan, Calculating the Singular Values and Pseudo-Inverse of a Matrix, SIAM. J. Numer. Anal., Vol. 2, 1965, pp. 202-224
- J. Wilkinson, Calculation of the eigenvalues of a symmetric tridiagonal matrix by the method of bisection, Numerische Mathematik, Vol. 4, 1962, pp. 362-367.
- G. Golub and P. Businger, Least Squares, Singular Values and Matrix Approximations, Stanford Technical Report No. CS73, 1967
- G. H. Golub and C. Reinsch, Singular Value Decomposition and Least Squares Solutions, Numer. Math., Vol. 14, 1970, pp. 403-420.
- J. Wilkinson and Chr. Reinsch, Linear Algebra, Springer, 1971.
- Raymond H. Chan, Chen Greif and Dianne P. O'Leary eds., Milestones in Matrix Computation: Selected Works of Gene H. Golub, with Commentaries, Oxford University Press, 2007.