Computing the Jordan structure of an eigenvalue

Nicola Mastronardi*, IAC-CNR, Bari, Italy
Paul Van Dooren, UCL, Louvain-la-Neuve, Belgium

Numerical Linear Algebra with Applications (NL2A)
CIRM Luminy, France, October 24–28, 2016

* partially supported by NL2A organizers
and by Mastronardi funds Res. gr. n. 0004/16.

Goal

- Given $A \in \mathbb{C}^{n \times n}$ and let λ (given) be an eigenvalue of A.

Goal

- Given $A \in \mathbb{C}^{n \times n}$ and let λ (given) be an eigenvalue of A.
- Aim: Compute the Jordan structure of λ in A.

Table of contents

Jordan canonical form theorem

Let $A \in \mathbb{C}^{n\times n}$. There is a nonsingular $S \in \mathbb{C}^{n\times n}$, positive integers q and $n_1, \cdots, n_q$ with $n_1 + n_2 + \cdots + n_q = n$, and scalars $\lambda_i \in \mathbb{C},\ i = 1, \ldots, q$, C such that

$$A = SJ_AS^{-1}, \quad J_A = \begin{bmatrix} J_{n_1}(\lambda_1) & & \\ & \ddots & \\ & & J_{n_q}(\lambda_q) \end{bmatrix}$$

The Jordan matrix J_A is uniquely determined by A up to permutation of its direct summands.
The *index* of λ_i is the maximum size of the Jordan blocks of A with eigenvalue λ_i.
The nonincreasingly ordered list of sizes of Jordan blocks of A with eigenvalue λ_i is called the *Segre characteristic* of A associated with the eigenvalue λ_i.

Weyr characteristic

- Let $A \in \mathbb{C}^{n\times n}$, λ eigenvalue of A and

$$r_k(A,\lambda) = \mathtt{rank}(A-\lambda I)^k, \quad r_0(A,\lambda) = n.$$

Weyr characteristic

- Let $A \in \mathbb{C}^{n \times n}$, λ eigenvalue of A and

$$r_k(A, \lambda) = \mathtt{rank}(A - \lambda I)^k, \quad r_0(A, \lambda) = n.$$

- Define

$$w_k(A, \lambda) = r_{k-1}(A, \lambda) - r_k(A, \lambda), \quad w_1(A, \lambda) = n - r_1(A, \lambda).$$

Weyr characteristic

- Let $A \in \mathbb{C}^{n \times n}$, λ eigenvalue of A and

$$r_k(A, \lambda) = \mathtt{rank}(A - \lambda I)^k, \quad r_0(A, \lambda) = n.$$

- Define

$$w_k(A, \lambda) = r_{k-1}(A, \lambda) - r_k(A, \lambda), \quad w_1(A, \lambda) = n - r_1(A, \lambda).$$

- $w_j(A, \lambda)$ is the number of blocks of size at least j

$$\Downarrow$$

$w_j(A, \lambda) - w_{j+1}(A, \lambda)$ is the number of blocks of size j

Weyr characteristic

- Let $A \in \mathbb{C}^{n\times n}$, λ eigenvalue of A and

$$r_k(A,\lambda) = \mathtt{rank}(A - \lambda I)^k, \;\; r_0(A,\lambda) = n.$$

- Define

$$w_k(A,\lambda) = r_{k-1}(A,\lambda) - r_k(A,\lambda), \;\; w_1(A,\lambda) = n - r_1(A,\lambda).$$

- $w_j(A,\lambda)$ is the number of blocks of size at least j

$$\Downarrow$$

$w_j(A,\lambda) - w_{j+1}(A,\lambda)$ is the number of blocks of size j

- The *Weyr characteristic* of A associated with $\lambda \in \mathbb{C}$ is

$$w(A,\lambda) = (w_1(A,\lambda), \cdots, w_q(A,\lambda))$$

Generalized null–space

Let us suppose that the matrix A has $\lambda = 0$ as an eigenvalue.

- Define the spaces $\mathcal{N}_i$ as the null spaces of the powers A^i.

Generalized null–space

Let us suppose that the matrix A has $\lambda = 0$ as an eigenvalue.

- Define the spaces $\mathcal{N}_i$ as the null spaces of the powers A^i.
- These null spaces are nested.

Generalized null–space

Let us suppose that the matrix A has $\lambda = 0$ as an eigenvalue.

- Define the spaces $\mathcal{N}_i$ as the null spaces of the powers A^i.
- These null spaces are nested.
- The *index* q of this eigenvalue is the smallest integer i for which the dimensions $n_i := \dim(\mathcal{N}_i)$ of these spaces do not change anymore :

$$\begin{aligned} \mathcal{N}_1 \subset \mathcal{N}_2 \subset \cdots \subset \mathcal{N}_q &= \mathcal{N}_{q+1} \\ n_1 < n_2 < \cdots < n_q &= n_{q+1}. \end{aligned}$$

Given $A \in \mathbb{C}^{n \times n}$, we can construct a unitary matrix V partitioned as

$$V = \left[\begin{array}{c|c|c|c|c} V_1 & V_2 & \cdots & V_q & V_{q+1} \end{array}\right]$$

where

$$\mathcal{N}_i = \mathrm{Im}(\left[\begin{array}{c|c|c|c} V_1 & V_2 & \cdots & V_i \end{array}\right]), \quad i = 1, \ldots, q$$

i.e. V_i completes the orthogonal basis of $\mathcal{N}_{i-1}$ to an orthogonal basis for the larger space $\mathcal{N}_i$.

Generalized null–space

V transforms A to the following staircase form

$$\tilde{A} = V^* A V = \left[\begin{array}{cccccc|c} 0_{w_1} & \tilde{A}_{1,2} & \cdots & \tilde{A}_{1,q-1} & \tilde{A}_{1,q} & \tilde{A}_{1,q+1} \\ 0 & 0_{w_2} & \tilde{A}_{2,3} & \cdots & \tilde{A}_{2,k} & \tilde{A}_{2,q+1} \\ \vdots & \vdots & \ddots & \ddots & \vdots & \vdots \\ \vdots & \vdots & & 0_{w_{q-1}} & \tilde{A}_{q-1,q} & \tilde{A}_{q-1,q+1} \\ 0 & 0 & \dots & \cdots & 0_{w_q} & \tilde{A}_{q,q+1} \\ \hline 0 & 0 & 0 & \cdots & 0 & \tilde{A}_{q+1,q+1} \end{array}\right]$$

where

- the diagonal blocks 0_{w_i} of $\tilde{A}$ are square and of dimension w_i, $i = 1, \ldots, q$,
- the blocks $\tilde{A}_{i-1,i}$ are of full column rank w_i, for $i = 2, \ldots, q$,
- the block $\tilde{A}_{q+1,q+1}$ is nonsingular (provided it is not empty),
- $w(A, 0) = (w_1(A, 0), \cdots, w_q(A, 0))$ is the Weyr characteristic of A at the eigenvalue 0.

Implicit QR algorithm

Implicit Q theorem
Let $A, H, Q \in \mathbb{C}^{n\times n}$, Q unitary and H upper Hessenberg with positive subdiagonal.
If

$$H = Q^H A Q,$$

then both H and Q are uniquely determined by the first column of Q.

Implicit QR algorithm for Eigenvalue problems

Let $A \in \mathbb{C}^{n \times n}$

- **Implicit QR algorithm**

  % Computation of an upper triangular matrix T
  % and a unitary matrix U such that $A = UTU^*$
  1) Set $A_0 = A$ and $U_0 = I$;
  2) for $k = 1, 2, \ldots,$ do
  3) Determine Q_k;
  4) $A_k = Q_k^H A_{k-1} Q_k$;
  5) $U_k = U_{k-1} Q_k$;
  6) end
  7) Set $T = A_\infty$ and $U = U_\infty$

Implicit QR algorithm for Eigenvalue problems

Let $A \in \mathbb{C}^{n \times n}$

- **Implicit QR algorithm**
  % Computation of an upper triangular matrix T
  % and a unitary matrix U such that $A = UTU^*$
  1) Set $A_0 = A$ and $U_0 = I$;
  2) for $k = 1, 2, \ldots,$ do
  3) Determine Q_k;
  4) $A_k = Q_k^H A_{k-1} Q_k$;
  5) $U_k = U_{k-1} Q_k$;
  6) end
  7) Set $T = A_\infty$ and $U = U_\infty$

- Each iteration requires $O(n^3)$ flops.

Implicit QR algorithm for Eigenvalue problems

Let $A \in \mathbb{C}^{n \times n}$

- **Implicit QR algorithm**
  % Computation of an upper triangular matrix T
  % and a unitary matrix U such that $A = UTU^*$
  1) Set $A_0 = A$ and $U_0 = I$;
  2) for $k = 1, 2, \ldots,$ do
  3) Determine Q_k;
  4) $A_k = Q_k^H A_{k-1} Q_k$;
  5) $U_k = U_{k-1} Q_k$;
  6) end
  7) Set $T = A_\infty$ and $U = U_\infty$

- Each iteration requires $O(n^3)$ flops.
- if A upper Hessenberg, each iteration requires $O(n^2)$ flops. In this case Q_k are Hessenberg matrices, too (sequence of $n-1$ Givens rotations).

Implicit QR algorithm

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & \times & \times \end{bmatrix} \quad \text{multiply by } G_1$$

Implicit QR algorithm

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & \times & \times \end{bmatrix} \quad \text{multiply by } G_1$$

Implicit QR algorithm

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & \times & \times \end{bmatrix} \qquad G_1^H H G_1$$

Implicit QR algorithm

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \otimes & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & \times & \times \end{bmatrix}$$

chase the bulge $\otimes$

Implicit QR algorithm

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & \times & \times \end{bmatrix} \quad G_2^H G_1^H H G_1 G_2$$

Implicit QR algorithm

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & \otimes & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & \times & \times \end{bmatrix} \quad \text{chase the bulge } \otimes$$

Implicit QR algorithm

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & & \times & \times \end{bmatrix} \quad G_3^H G_2^H G_1^H H G_1 G_2 G_3$$

Implicit QR algorithm

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & \otimes & \times & \times & \times \\ & & & & \times & \times \end{bmatrix}$$ chase the bulge $\otimes$

Implicit QR algorithm

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & \times & \times & \times \end{bmatrix} \quad G_4^H G_3^H G_2^H G_1^H H G_1 G_2 G_3 G_4$$

Implicit QR algorithm

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & \otimes & \times & \times \end{bmatrix} \quad \text{chase the bulge } \otimes$$

Implicit QR algorithm

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & \times & \times \end{bmatrix} \quad G_5^H G_4^H G_3^H G_2^H G_1^H H G_1 G_2 G_3 G_4 G_5$$

Implicit QR algorithm for Eigenvalue problems

- If the shift $\sigma = \lambda_i$, (perfect shift) in theory, then one step of the Implicit QR algorithm is needed to compute λ_i. Indeed, it turns out that

$$Q^H A Q = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & 0 & \lambda_i \end{bmatrix}$$

Implicit QR algorithm for Eigenvalue problems

- If the shift $\sigma = \lambda_i$, (perfect shift) in theory, then one step of the Implicit QR algorithm is needed to compute λ_i. Indeed, it turns out that

$$Q^H A Q = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ & \times & \times & \times & \times & \times \\ & & \times & \times & \times & \times \\ & & & \times & \times & \times \\ & & & & 0 & \lambda_i \end{bmatrix}$$

- Moreover, the last column of Q is the corresponding eigenvector.

Example

A_0 random Hessenberg matrix of order 6. Apply the reverse implicit QR method with zero shift.

Iteration 1 :

$$
\begin{array}{cc}
A_1 & Q_1(:,1) \\
\begin{bmatrix}
2.1254e-01 & 5.1355e-01 & -9.8834e-01 & -6.7006e-01 & -9.1117e-01 & 1.9544e+00 \\
1.6444e+00 & -1.4496e+00 & 1.3054e+00 & 1.0498e+00 & -2.2361e+00 & -1.3969e+00 \\
0 & 2.7097e+00 & -1.2429e+00 & -3.1337e-01 & 3.8856e-02 & 2.3230e+00 \\
0 & 0 & 2.2401e+00 & 1.1219e-01 & 2.0416e+00 & 1.4699e+00 \\
0 & 0 & 0 & 1.3407e+00 & -9.2542e-01 & 3.4951e-01 \\
0 & 0 & 0 & 0 & 1.0699e+00 & -7.9406e-01
\end{bmatrix}, &
\begin{bmatrix}
1.2819e-01 \\
-2.4644e-01 \\
-3.2405e-01 \\
-3.7833e-01 \\
-2.0859e-01 \\
7.9447e-01
\end{bmatrix}
\end{array}
$$

Example

A_0 random Hessenberg matrix of order 6. Apply the reverse implicit QR method with zero shift.

Iteration 4 :

$$
\begin{array}{cc}
A_4 & Q_4(:,1) \\
\begin{bmatrix}
6.4386e-01 & 1.0008e+00 & 1.6293e-01 & 3.9208e-01 & -1.6347e-01 & -1.7600e+00 \\
-9.5792e-02 & 7.9231e-01 & 8.3601e-01 & -6.8727e-01 & -4.6446e-01 & -2.2911e+00 \\
0 & 7.8491e-01 & -9.0158e-01 & -1.7201e+00 & 1.6668e+00 & 5.3743e-01 \\
0 & 0 & 1.8797e+00 & 4.5564e-03 & 2.8826e+00 & 1.0044e+00 \\
0 & 0 & 0 & 2.2059e+00 & -2.7218e+00 & -4.4165e-01 \\
0 & 0 & 0 & 0 & 1.6866e+00 & -1.9046e+00
\end{bmatrix}, &
\begin{bmatrix}
-9.8911e-01 \\ 1.1251e-01 \\ 5.2169e-02 \\ 2.5834e-02 \\ -5.5525e-02 \\ -5.0259e-02
\end{bmatrix}
\end{array}
$$

Example

A_0 random Hessenberg matrix of order 6. Apply the reverse implicit QR method with zero shift.

Iteration 8 :

$$\begin{array}{cc} A_8 & Q_8(:,1) \\ \begin{bmatrix} 6.9865e-01 & -1.0905e+00 & 1.0178e+00 & -2.0665e-01 & -1.3190e+00 & 6.7463e-01 \\ -2.1948e-03 & 1.0442e+00 & -1.4087e+00 & 2.1662e-01 & 4.1660e-01 & 1.1266e+00 \\ 0 & 1.1958e+00 & 1.7750e-01 & 2.2040e+00 & -2.1854e-01 & 1.9325e-01 \\ 0 & 0 & 1.8807e+00 & -6.0932e-01 & -1.6119e+00 & 1.5927e+00 \\ 0 & 0 & 0 & 4.0942e-01 & -1.9581e+00 & -3.2083e+00 \\ 0 & 0 & 0 & 0 & 3.8693e-01 & -3.4401e+00 \end{bmatrix}, & \begin{bmatrix} -1.0000e+00 \\ 2.0624e-03 \\ -1.2570e-03 \\ -1.9294e-03 \\ -5.4949e-04 \\ -1.0571e-04 \end{bmatrix} \end{array}$$

Example

A_0 random Hessenberg matrix of order 6. Apply the reverse implicit QR method with zero shift.

Iteration 12 :

$$\begin{array}{cc}
A_{12} & Q_{12}(:,1) \\
\begin{bmatrix}
6.9525e-01 & -3.8666e-01 & -8.7825e-01 & -1.4922e+00 & -4.9689e-01 & 1.0232e+00 \\
-7.9909e-05 & 1.5766e+00 & -4.9295e-01 & -1.0653e+00 & 9.0330e-01 & -1.4783e+00 \\
0 & 1.4161e+00 & 7.4398e-01 & 1.9875e+00 & 7.1971e-01 & 3.2317e-01 \\
0 & 0 & 3.9805e-01 & -2.0739e+00 & -8.6430e-01 & 2.5337e-02 \\
0 & 0 & 0 & 2.5597e-01 & -2.7158e+00 & -3.7790e+00 \\
0 & 0 & 0 & 0 & 2.0132e-01 & -2.3133e+00
\end{bmatrix}, &
\begin{bmatrix}
-1.0000e+00 \\
8.5507e-05 \\
-7.0018e-05 \\
-3.1433e-05 \\
-2.8754e-06 \\
-2.0869e-07
\end{bmatrix}
\end{array}$$

Example

A_0 random Hessenberg matrix of order 6. Apply the reverse implicit QR method with zero shift.

Iteration 16 :

$$
\begin{array}{cc}
A_{16} & Q_{16}(:,1) \\
\begin{bmatrix}
6.9534e-01 & 1.0233e+00 & 4.3544e-01 & -1.3925e+00 & 1.0340e+00 & 4.4403e-01 \\
-4.8008e-06 & 1.5777e+00 & -1.3327e+00 & -9.8825e-01 & 5.4083e-01 & 1.0299e+00 \\
0 & 5.7733e-01 & 1.0699e+00 & -1.5714e+00 & -1.0803e+00 & -9.0116e-01 \\
0 & 0 & 1.1093e-01 & -2.5524e+00 & -8.8285e-01 & 7.6526e-01 \\
0 & 0 & 0 & 4.2870e-02 & -2.9008e+00 & -3.0216e-01 \\
0 & 0 & 0 & 0 & 3.6724e+00 & -1.9770e+00
\end{bmatrix}, &
\begin{bmatrix}
-1.0000e+00 \\
6.4837e-06 \\
-2.3658e-06 \\
-1.7938e-07 \\
-1.9438e-09 \\
-2.4496e-09
\end{bmatrix}
\end{array}
$$

The QR algorithm for computing the eigenvalues is backward stable and the error on th3e computed eigenvalues depends on the condition numbers of the eigenvalues

First Example

We consider a matrix of order 18 considered in the paper

N. Guglielmi, M.L. Overton, G.W. Stewart, *An Efficient Algorithm for Computing the Generalized Null Space Decomposition*, **SIAM J. Matrix Anal. Appl.** 36, 38–54, 2015.

First Example

We consider a matrix of order 18 considered in the paper

N. Guglielmi, M.L. Overton, G.W. Stewart, *An Efficient Algorithm for Computing the Generalized Null Space Decomposition*, **SIAM J. Matrix Anal. Appl.** 36, 38–54, 2015.

$$\sigma(A) = \begin{bmatrix} 1.9288e+00 \\ 1.3561e+00 \\ 8.7206e-01 \\ 6.8301e-01 \\ 6.3444e-01 \\ 3.1994e-01 \\ 3.1287e-01 \\ 1.6030e-01 \\ 3.1547e-02 \\ 1.7171e-02 \\ 1.4103e-02 \\ 1.1660e-02 \\ 8.8228e-03 \\ 8.4642e-03 \\ 7.5872e-07 \\ 6.5816e-07 \\ 9.1614e-11 \\ 2.5727e-16 \end{bmatrix}$$

First Example

We consider a matrix of order 18 considered in the paper

N. Guglielmi, M.L. Overton, G.W. Stewart, *An Efficient Algorithm for Computing the Generalized Null Space Decomposition*, **SIAM J. Matrix Anal. Appl.** 36, 38–54, 2015.

$$\sigma(A) = \begin{bmatrix} 1.9288e+00 \\ 1.3561e+00 \\ 8.7206e-01 \\ 6.8301e-01 \\ 6.3444e-01 \\ 3.1994e-01 \\ 3.1287e-01 \\ 1.6030e-01 \\ 3.1547e-02 \\ 1.7171e-02 \\ 1.4103e-02 \\ 1.1660e-02 \\ 8.8228e-03 \\ 8.4642e-03 \\ 7.5872e-07 \\ 6.5816e-07 \\ 9.1614e-11 \\ 2.5727e-16 \end{bmatrix} \qquad \lambda(A) = \begin{bmatrix} 1.0000e+00+0.0000e+00i \\ 2.5000e-01+0.0000e+00i \\ 2.5000e-01+0.0000e+00i \\ -1.7684e-04+5.6050e-05i \\ -1.7684e-04-5.6050e-05i \\ -1.4499e-04+0.0000e+00i \\ -3.3473e-05+0.0000e+00i \\ -1.6317e-04+0.0000e+00i \\ 2.3786e-05+1.7410e-04i \\ 2.3786e-05-1.7410e-04i \\ 8.4030e-05+0.0000e+00i \\ 1.8033e-04+4.3683e-05i \\ 1.8033e-04-4.3683e-05i \\ 1.8292e-04+7.0199e-06i \\ 1.8292e-04-7.0199e-06i \\ 6.2500e-02+0.0000e+00i \\ 6.2500e-02+0.0000e+00i \\ 6.2500e-02+0.0000e+00i \end{bmatrix}$$

First Example

We consider a matrix of order 18 considered in the paper

N. Guglielmi, M.L. Overton, G.W. Stewart, *An Efficient Algorithm for Computing the Generalized Null Space Decomposition*, **SIAM J. Matrix Anal. Appl.** 36, 38–54, 2015.

$$\sigma(A) = \begin{bmatrix} 1.9288e+00 \\ 1.3561e+00 \\ 8.7206e-01 \\ 6.8301e-01 \\ 6.3444e-01 \\ 3.1994e-01 \\ 3.1287e-01 \\ 1.6030e-01 \\ 3.1547e-02 \\ 1.7171e-02 \\ 1.4103e-02 \\ 1.1660e-02 \\ 8.8228e-03 \\ 8.4642e-03 \\ 7.5872e-07 \\ 6.5816e-07 \\ 9.1614e-11 \\ 2.5727e-16 \end{bmatrix} \qquad \lambda(A) = \begin{bmatrix} 1.0000e+00+0.0000e+00i \\ 2.5000e-01+0.0000e+00i \\ 2.5000e-01+0.0000e+00i \\ -1.7684e-04+5.6050e-05i \\ -1.7684e-04-5.6050e-05i \\ -1.4499e-04+0.0000e+00i \\ -3.3473e-05+0.0000e+00i \\ -1.6317e-04+0.0000e+00i \\ 2.3786e-05+1.7410e-04i \\ 2.3786e-05-1.7410e-04i \\ 8.4030e-05+0.0000e+00i \\ 1.8033e-04+4.3683e-05i \\ 1.8033e-04-4.3683e-05i \\ 1.8292e-04+7.0199e-06i \\ 1.8292e-04-7.0199e-06i \\ 6.2500e-02+0.0000e+00i \\ 6.2500e-02+0.0000e+00i \\ 6.2500e-02+0.0000e+00i \end{bmatrix}$$

$$\mathrm{rank}(A) = 17, \qquad \mathrm{rank}\ \mathrm{diag}(\lambda(A)) = 18.$$

First Example

We consider a matrix of order 18 considered in the paper

N. Guglielmi, M.L. Overton, G.W. Stewart, *An Efficient Algorithm for Computing the Generalized Null Space Decomposition*, **SIAM J. Matrix Anal. Appl.** 36, 38–54, 2015.

$$\sigma(A) = \begin{bmatrix} 1.9288e+00 \\ 1.3561e+00 \\ 8.7206e-01 \\ 6.8301e-01 \\ 6.3444e-01 \\ 3.1994e-01 \\ 3.1287e-01 \\ 1.6030e-01 \\ 3.1547e-02 \\ 1.7171e-02 \\ 1.4103e-02 \\ 1.1660e-02 \\ 8.8228e-03 \\ 8.4642e-03 \\ 7.5872e-07 \\ 6.5816e-07 \\ 9.1614e-11 \\ 2.5727e-16 \end{bmatrix} \qquad \lambda(A) = \begin{bmatrix} 1.0000e+00+0.0000e+00i \\ 2.5000e-01+0.0000e+00i \\ 2.5000e-01+0.0000e+00i \\ -1.7684e-04+5.6050e-05i \\ -1.7684e-04-5.6050e-05i \\ -1.4499e-04+0.0000e+00i \\ -3.3473e-05+0.0000e+00i \\ -1.6317e-04+0.0000e+00i \\ 2.3786e-05+1.7410e-04i \\ 2.3786e-05-1.7410e-04i \\ 8.4030e-05+0.0000e+00i \\ 1.8033e-04+4.3683e-05i \\ 1.8033e-04-4.3683e-05i \\ 1.8292e-04+7.0199e-06i \\ 1.8292e-04-7.0199e-06i \\ 6.2500e-02+0.0000e+00i \\ 6.2500e-02+0.0000e+00i \\ 6.2500e-02+0.0000e+00i \end{bmatrix}$$

$\mathrm{rank}(A) = 17, \qquad \mathrm{rank}\ \mathrm{diag}(\lambda(A)) = 18.$

$\lambda_n, \mathbf{y}_{\lambda_n}, \mathbf{x}_{\lambda_n}$ smallest eigenvalue of A and associated eigenvectors.

$\mathrm{cond}(\lambda_n) = \frac{1}{|\mathbf{y}_{\lambda_n}^H \mathbf{x}_{\lambda_n}|} = 2.6901e+11.$

This means that $O(\epsilon)$ perturbations in A can induce $\frac{\epsilon}{|\mathbf{y}_{\lambda_n}^H \mathbf{x}_{\lambda_n}|}$ changes in the eigenvalue

(if $\epsilon = 2.2204e-16$, $\frac{\epsilon}{|\mathbf{y}_{\lambda_n}^H \mathbf{x}_{\lambda_n}|} = 5.9732e-05$.)

Second example

Z. ZENG, *Sensitivity and Computation of a Defective Eigenvalue*, **SIAM. J. Matrix Anal. Appl.**, 37(2), 798817.

$$A_{20} = \begin{bmatrix}
0 & 4 & 0 & -4 & 0 & -2 & 1 & 0 & 0 & -1 & -1 & -1 & -1 & 2 & 1 & 0 & 0 & -1 & 0 & 0 \\
0 & 3 & 3 & -4 & 1 & 0 & 4 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & -1 & -1 \\
1 & -4 & 2 & 10 & -3 & 1 & -7 & -2 & -1 & 3 & 2 & 0 & 0 & -1 & 0 & 3 & -1 & 1 & 1 & 2 \\
-1 & -1 & 2 & 5 & -2 & -1 & -5 & -1 & -1 & 2 & 0 & -1 & 0 & 1 & 0 & 2 & -1 & -1 & -1 & 1 \\
-1 & -2 & 2 & 1 & 1 & 1 & -1 & 0 & -2 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & -1 & 0 & 0 \\
1 & 4 & 1 & -12 & 4 & 2 & 13 & 3 & 0 & -4 & 0 & 0 & -2 & -1 & 1 & -6 & 1 & 1 & 0 & -3 \\
-1 & -1 & 1 & 5 & -2 & 0 & -4 & -2 & 0 & 1 & -1 & 0 & 0 & 1 & 0 & 4 & 0 & -1 & -1 & 2 \\
1 & 2 & -4 & 0 & 1 & 0 & 1 & 1 & 4 & -2 & -1 & 1 & 0 & -1 & 0 & 1 & 2 & 1 & 0 & 1 \\
0 & -5 & 2 & 10 & -5 & -1 & -10 & -2 & -1 & 6 & 3 & -2 & 0 & 0 & 0 & 3 & -3 & 0 & 1 & 2 \\
1 & 1 & 1 & -1 & 2 & 2 & 4 & 0 & 1 & -1 & -1 & 2 & 0 & -1 & 0 & 0 & 2 & 1 & -1 & 0 \\
1 & -1 & 0 & 2 & 1 & 2 & 1 & 0 & 1 & -1 & 3 & 2 & 0 & -1 & 0 & 0 & 1 & 1 & -1 & 0 \\
-1 & -3 & 0 & 5 & -1 & 2 & -4 & -1 & 0 & 1 & -1 & 4 & 4 & 1 & -2 & 2 & 0 & -1 & 0 & 1 \\
-2 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 3 & 2 & 0 & 0 & 0 & -1 & 0 & 0 \\
-3 & 4 & -1 & -4 & 0 & -2 & 1 & 0 & 0 & -1 & -1 & -1 & -2 & 5 & 2 & 0 & 0 & -1 & 1 & 0 \\
-2 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 2 & 3 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & -2 & 3 & 1 & 2 & -1 & -1 & 2 & -2 & -2 & 2 & 0 & 0 & 0 & 5 & 2 & 0 & 0 & 1 \\
6 & 3 & -6 & 3 & 6 & 4 & 7 & 0 & 7 & -7 & 1 & 5 & -2 & -6 & 1 & 0 & 8 & 6 & -1 & 0 \\
0 & 2 & -4 & -4 & 1 & -1 & 4 & 1 & 0 & -1 & 0 & -1 & -1 & 0 & 1 & -2 & 0 & 3 & 4 & -1 \\
1 & -4 & -1 & 11 & -4 & 1 & -8 & -3 & -1 & 3 & 2 & 0 & 0 & -1 & 0 & 4 & -1 & 1 & 4 & 3 \\
0 & 0 & -1 & 1 & -2 & 0 & -1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 4
\end{bmatrix}$$

- The eigenvalues of A_{20} are 2 and 3 of nonzero Segre characteristic $\{4, 3, 3\}$ and $\{5, 5\}$, respectively.

Second example

Z. Zeng, *Sensitivity and Computation of a Defective Eigenvalue*, **SIAM. J. Matrix Anal. Appl.**, 37(2), 798817.

$$A_{20} = \begin{bmatrix}
0 & 4 & 0 & -4 & 0 & -2 & 1 & 0 & 0 & -1 & -1 & -1 & -1 & 2 & 1 & 0 & 0 & -1 & 0 & 0 \\
0 & 3 & 3 & -4 & 1 & 0 & 4 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & -1 & -1 \\
1 & -4 & 2 & 10 & -3 & 1 & -7 & -2 & -1 & 3 & 2 & 0 & 0 & -1 & 0 & 3 & -1 & 1 & 1 & 2 \\
-1 & -1 & 2 & 5 & -2 & -1 & -5 & -1 & -1 & 2 & 0 & -1 & 0 & 1 & 0 & 2 & -1 & -1 & -1 & 1 \\
-1 & -2 & 2 & 1 & 1 & 1 & -1 & 0 & -2 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & -1 & 0 & 0 \\
1 & 4 & 1 & -12 & 4 & 2 & 13 & 3 & 0 & -4 & 0 & 0 & -2 & -1 & 1 & -6 & 1 & 1 & 0 & -3 \\
-1 & -1 & 1 & 5 & -2 & 0 & -4 & -2 & 0 & 1 & -1 & 0 & 0 & 1 & 0 & 4 & 0 & -1 & -1 & 2 \\
1 & 2 & -4 & 0 & 1 & 0 & 1 & 1 & 4 & -2 & -1 & 1 & 0 & -1 & 0 & 1 & 2 & 1 & 0 & 1 \\
0 & -5 & 2 & 10 & -5 & -1 & -10 & -2 & -1 & 6 & 3 & -2 & 0 & 0 & 0 & 3 & -3 & 0 & 1 & 2 \\
1 & 1 & 1 & -1 & 2 & 2 & 4 & 0 & 1 & -1 & -1 & 2 & 0 & -1 & 0 & 0 & 2 & 1 & -1 & 0 \\
1 & -1 & 0 & 2 & 1 & 2 & 1 & 0 & 1 & -1 & 3 & 2 & 0 & -1 & 0 & 0 & 1 & 1 & -1 & 0 \\
-1 & -3 & 0 & 5 & -1 & 2 & -4 & -1 & 0 & 1 & -1 & 4 & 4 & 1 & -2 & 2 & 0 & -1 & 0 & 1 \\
-2 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 3 & 2 & 0 & 0 & 0 & -1 & 0 & 0 \\
-3 & 4 & -1 & -4 & 0 & -2 & 1 & 0 & 0 & -1 & -1 & -1 & -2 & 5 & 2 & 0 & 0 & -1 & 1 & 0 \\
-2 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 2 & 3 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & -2 & 3 & 1 & 2 & -1 & -1 & 2 & -2 & -2 & 2 & 0 & 0 & 0 & 5 & 2 & 0 & 0 & 1 \\
6 & 3 & -6 & 3 & 6 & 4 & 7 & 0 & 7 & -7 & 1 & 5 & -2 & -6 & 1 & 0 & 8 & 6 & -1 & 0 \\
0 & 2 & -4 & -4 & 1 & -1 & 4 & 1 & 0 & -1 & 0 & -1 & -1 & 0 & 1 & -2 & 0 & 3 & 4 & -1 \\
1 & -4 & -1 & 11 & -4 & 1 & -8 & -3 & -1 & 3 & 2 & 0 & 0 & -1 & 0 & 4 & -1 & 1 & 4 & 3 \\
0 & 0 & -1 & 1 & -2 & 0 & -1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 4
\end{bmatrix}$$

- The eigenvalues of A_{20} are 2 and 3 of nonzero Segre characteristic $\{4, 3, 3\}$ and $\{5, 5\}$, respectively.
- Means of clusters of defective eigenvalues are not hypersensitive (Ruhe, '70) :

Second example

Z. Zeng, *Sensitivity and Computation of a Defective Eigenvalue*, **SIAM. J. Matrix Anal. Appl.**, 37(2), 798817.

$$
A_{20} = \begin{bmatrix}
0 & 4 & 0 & -4 & 0 & -2 & 1 & 0 & 0 & -1 & -1 & -1 & -1 & 2 & 1 & 0 & 0 & -1 & 0 & 0 \\
0 & 3 & 3 & -4 & 1 & 0 & 4 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & -1 & -1 \\
1 & -4 & 2 & 10 & -3 & 1 & -7 & -2 & -1 & 3 & 2 & 0 & 0 & -1 & 0 & 3 & -1 & 1 & 1 & 2 \\
-1 & -1 & 2 & 5 & -2 & -1 & -5 & -1 & -1 & 2 & 0 & -1 & 0 & 1 & 0 & 2 & -1 & -1 & -1 & 1 \\
-1 & -2 & 2 & 1 & 1 & 1 & -1 & 0 & -2 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & -1 & -1 & 0 & 0 \\
1 & 4 & 1 & -12 & 4 & 2 & 13 & 3 & 0 & -4 & 0 & 0 & -2 & -1 & 1 & -6 & 1 & 1 & 0 & -3 \\
-1 & -1 & 1 & 5 & -2 & 0 & -4 & -2 & 0 & 1 & -1 & 0 & 0 & 1 & 0 & 4 & 0 & -1 & -1 & 2 \\
1 & 2 & -4 & 0 & 1 & 0 & 1 & 1 & 4 & -2 & -1 & 1 & 0 & -1 & 0 & 1 & 2 & 1 & 0 & 1 \\
0 & -5 & 2 & 10 & -5 & -1 & -10 & -2 & -1 & 6 & 3 & -2 & 0 & 0 & 0 & 3 & -3 & 0 & 1 & 2 \\
1 & 1 & 1 & -1 & 2 & 2 & 4 & 0 & 1 & -1 & -1 & 2 & 0 & -1 & 0 & 0 & 2 & 1 & -1 & 0 \\
1 & -1 & 0 & 2 & 1 & 2 & 1 & 0 & 1 & -1 & 3 & 2 & 0 & -1 & 0 & 0 & 1 & 1 & -1 & 0 \\
-1 & -3 & 0 & 5 & -1 & 2 & -4 & -1 & 0 & 1 & -1 & 4 & 4 & 1 & -2 & 2 & 0 & -1 & 0 & 1 \\
-2 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 3 & 2 & 0 & 0 & 0 & -1 & 0 & 0 \\
-3 & 4 & -1 & -4 & 0 & -2 & 1 & 0 & 0 & -1 & -1 & -1 & -2 & 5 & 2 & 0 & 0 & -1 & 1 & 0 \\
-2 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 2 & 3 & 0 & 0 & -1 & 0 & 0 \\
0 & 0 & -2 & 3 & 1 & 2 & -1 & -1 & 2 & -2 & -2 & 2 & 0 & 0 & 0 & 5 & 2 & 0 & 0 & 1 \\
6 & 3 & -6 & 3 & 6 & 4 & 7 & 0 & 7 & -7 & 1 & 5 & -2 & -6 & 1 & 0 & 8 & 6 & -1 & 0 \\
0 & 2 & -4 & -4 & 1 & -1 & 4 & 1 & 0 & -1 & 0 & -1 & -1 & 0 & 1 & -2 & 0 & 3 & 4 & -1 \\
1 & -4 & -1 & 11 & -4 & 1 & -8 & -3 & -1 & 3 & 2 & 0 & 0 & -1 & 0 & 4 & -1 & 1 & 4 & 3 \\
0 & 0 & -1 & 1 & -2 & 0 & -1 & -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 4
\end{bmatrix}
$$

- The eigenvalues of A_{20} are 2 and 3 of nonzero Segre characteristic $\{4, 3, 3\}$ and $\{5, 5\}$, respectively.
- Means of clusters of defective eigenvalues are not hypersensitive (Ruhe, '70) :
- $\frac{\mathtt{sum(eig}(A_{20}(1{:}10)))}{10} = 3.000000000000001e + 00$
  $\frac{\mathtt{sum(eig}(A_{20}(11{:}20)))}{10} = 2.000000000000001e + 00$

$$
\texttt{eig}(A_{20}) = \begin{bmatrix}
3.000398490901253 + 1.224915665189403e-03i \\
3.000398490901253 - 1.224915665189403e-03i \\
3.000646066870935 + 4.696276460577337e-04i \\
3.000646066870935 - 4.696276460577337e-04i \\
3.001287762162967 + 0.000000000000000e+00i \\
2.999753002133234 + 7.591919143315616e-04i \\
2.999753002133234 - 7.591919143315616e-04i \\
2.998957628017279 + 7.576817588335188e-04i \\
2.998957628017279 - 7.576817588335188e-04i \\
2.999201861991639 + 0.000000000000000e+00i \\
2.000118556521482 + 1.183979295902058e-04i \\
2.000118556521482 - 1.183979295902058e-04i \\
1.999881443477439 + 1.187148607245887e-04i \\
1.999881443477439 - 1.187148607245887e-04i \\
2.000013778528383 + 1.810574229519768e-05i \\
2.000013778528383 - 1.810574229519768e-05i \\
2.000008786021464 + 2.097972084850244e-05i \\
2.000008786021464 - 2.097972084850244e-05i \\
1.999977435451235 + 2.873978888097808e-06i \\
1.999977435451235 - 2.873978888097808e-06i
\end{bmatrix} ; \quad \texttt{condeig}(A_{20}) = \begin{bmatrix}
4.4929e+11 \\
4.4929e+11 \\
1.5169e+12 \\
1.5169e+12 \\
4.4949e+11 \\
1.5263e+12 \\
1.5263e+12 \\
4.4896e+11 \\
4.4897e+11 \\
1.5321e+12 \\
7.7385e+11 \\
7.7385e+11 \\
7.7077e+11 \\
7.7078e+11 \\
1.3631e+11 \\
1.3631e+11 \\
1.3633e+11 \\
1.3633e+11 \\
1.3636e+11 \\
1.3636e+11
\end{bmatrix}
$$

- Can we compute the eigenvalues in a more accurate way?

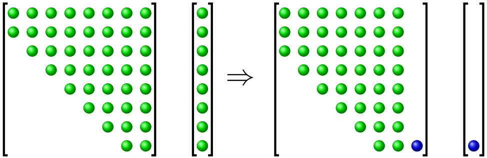
- Can we compute the eigenvalues in a more accurate way?
- Since we are interested in the Jordan structure at 0 we first compute the corresponding right (left) eigenvector (=singular vector) with one step of inverse iteration.

- Can we compute the eigenvalues in a more accurate way?
- Since we are interested in the Jordan structure at 0 we first compute the corresponding right (left) eigenvector (=singular vector) with one step of inverse iteration.
- Problem: how to to deflate the matrix

Businger's algorithm

Given a Hessemberg matrix $H \in \mathbb{R}^{n \times n}$ with eigenpair (λ, x) the algorithm deflates the H removing the eigenvalue λ into two steps:

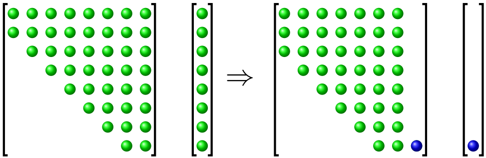
- Transform x into $\pm e_n \Rightarrow H$ is transformed into a similar one with one more subdiagonal:



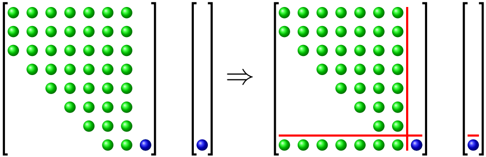
Businger's algorithm

Given a Hessemberg matrix $H \in \mathbb{R}^{n \times n}$ with eigenpair (λ, x) the algorithm deflates the H removing the eigenvalue λ into two steps:

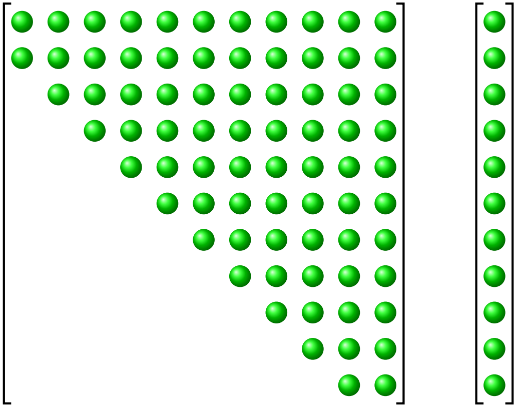
- Transform x into $\pm e_n \Rightarrow H$ is transformed into a similar one with one more subdiagonal:



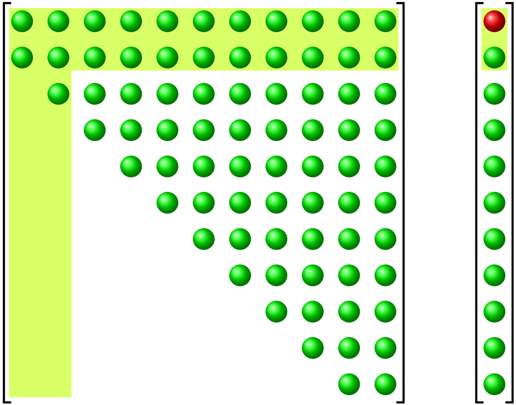
- The second subdiagonal is removed by a *"chasing the bulge"* technique:



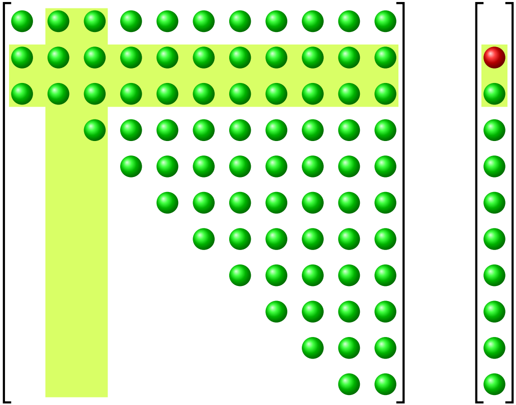
Businger's algorithm 1



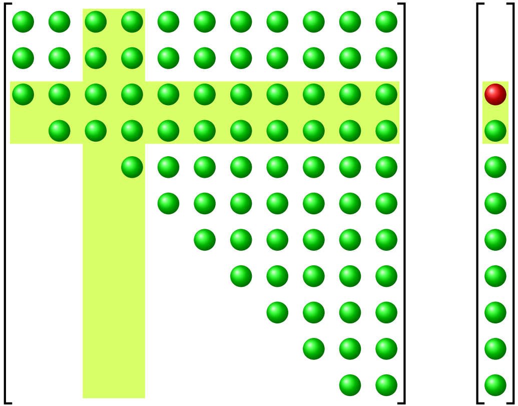
Businger's algorithm 2



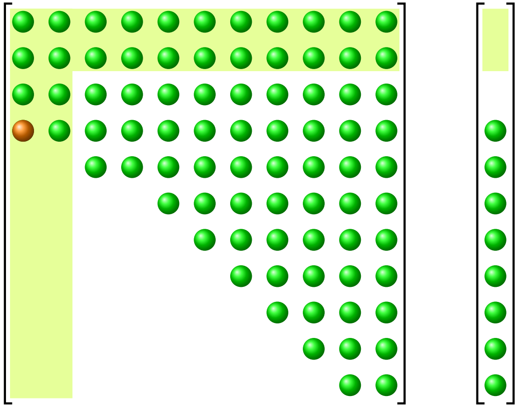
Businger's algorithm 3



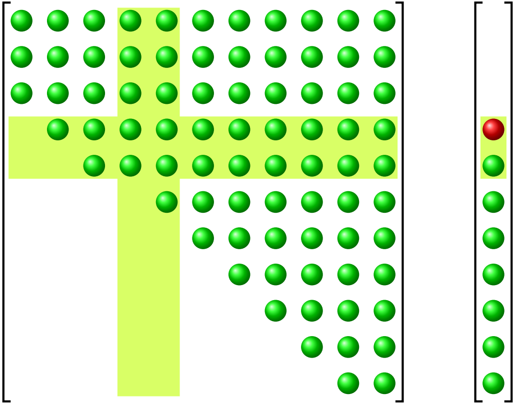
Businger's algorithm 4



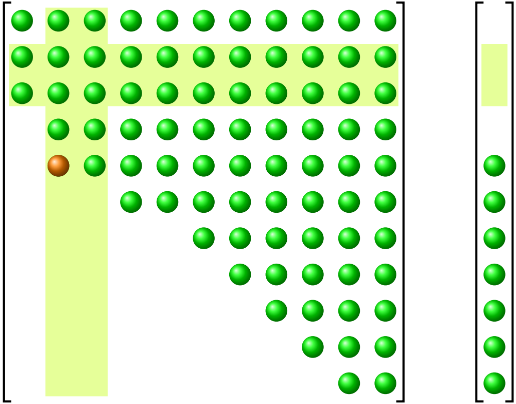
Businger's algorithm 5



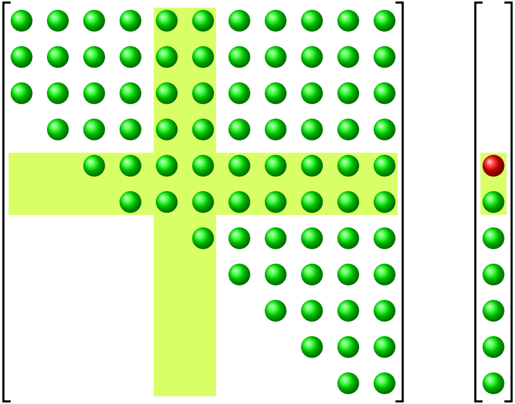
Businger's algorithm 6



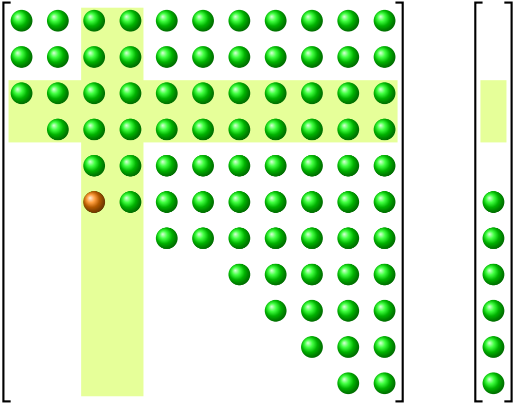
Businger's algorithm 7



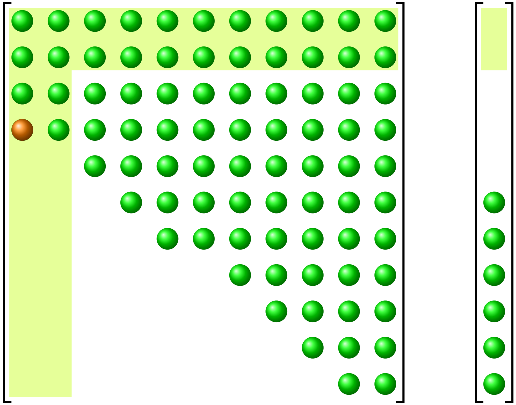
Businger's algorithm 8



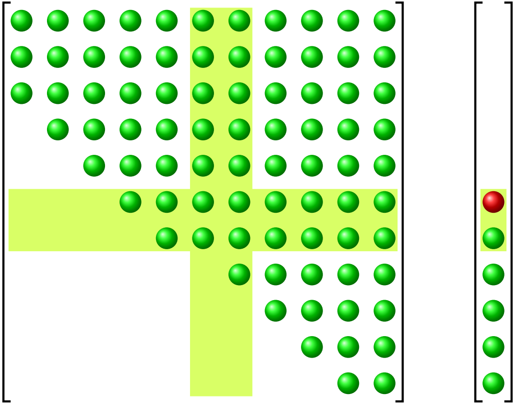
Businger's algorithm 9



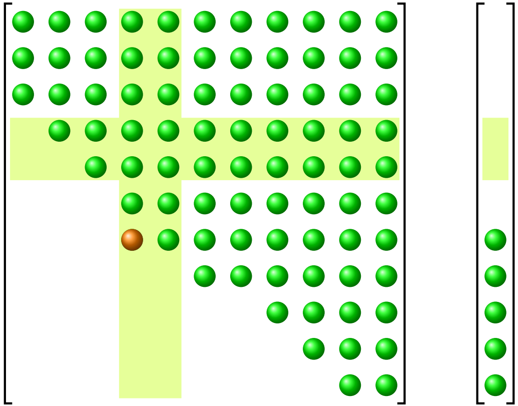
Businger's algorithm 10



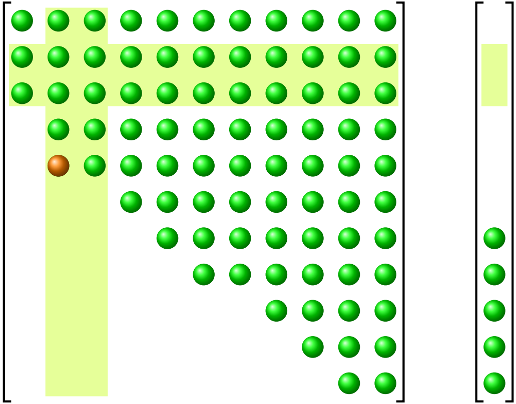
Businger's algorithm 11



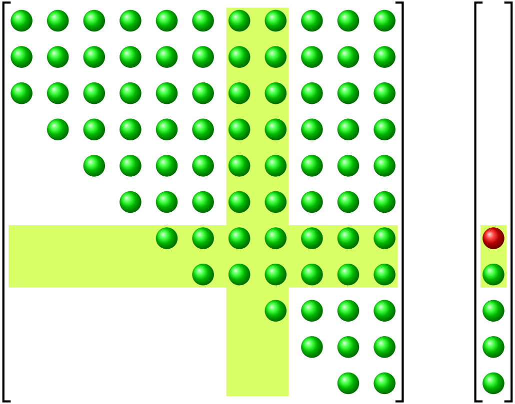
Businger's algorithm 12



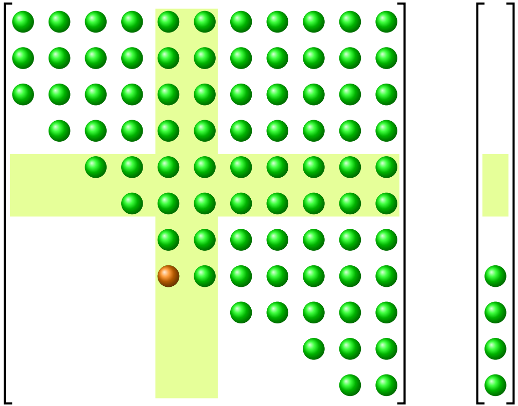
Businger's algorithm 13



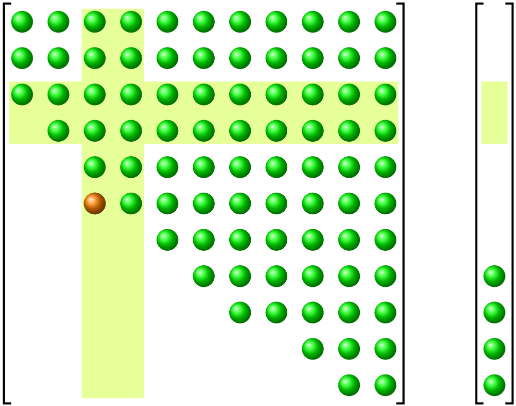
Businger's algorithm 14



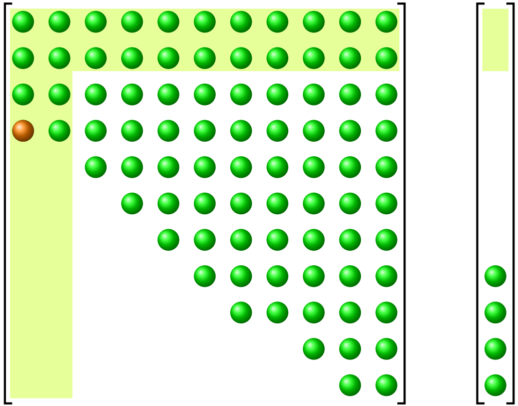
Businger's algorithm 15



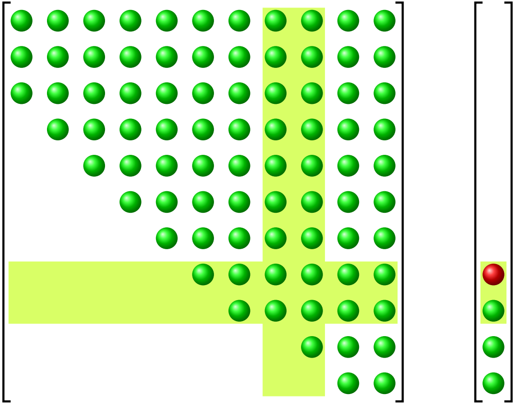
Businger's algorithm 16



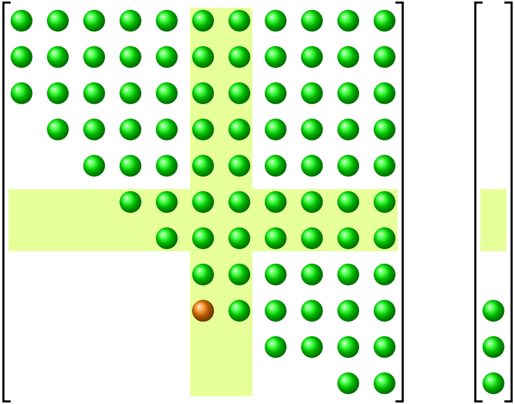
Businger's algorithm 17



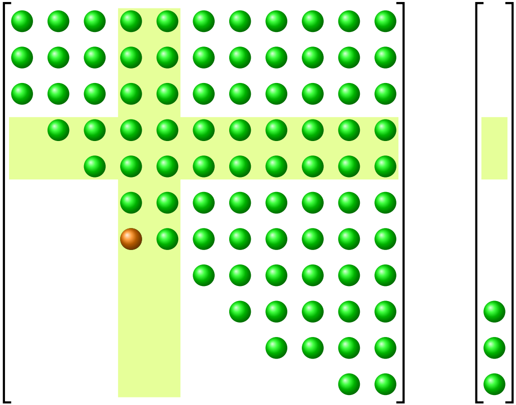
Businger's algorithm 18



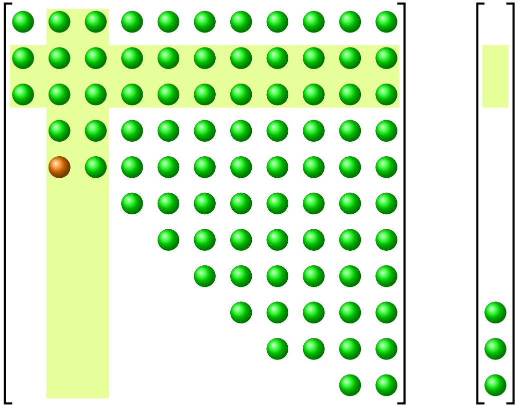
Businger's algorithm 19



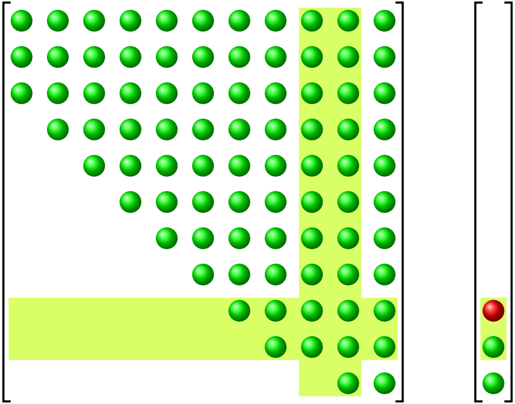
Businger's algorithm 20



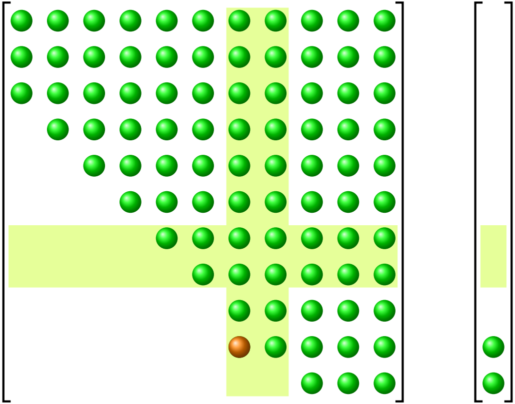
Businger's algorithm 21



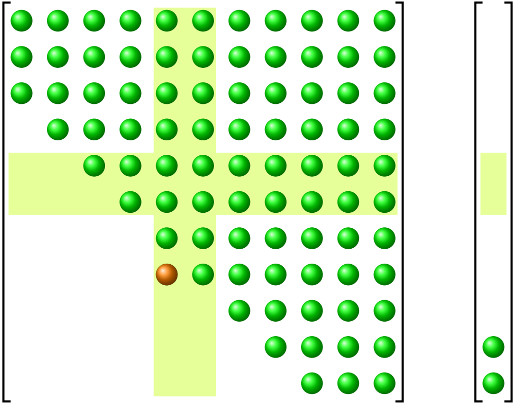
Businger's algorithm 22



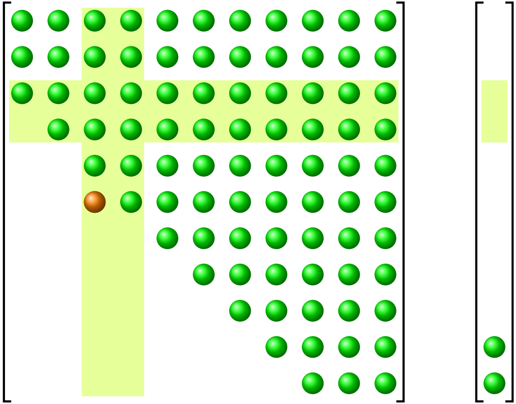
Businger's algorithm 23



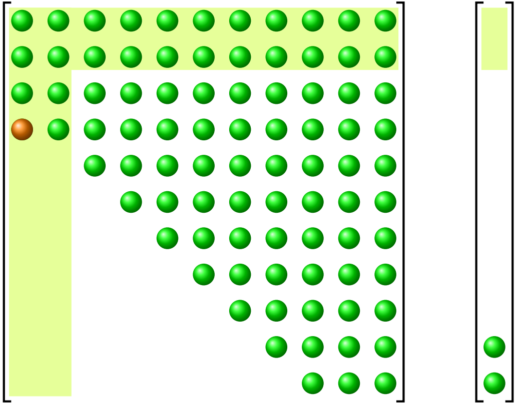
Businger's algorithm 24



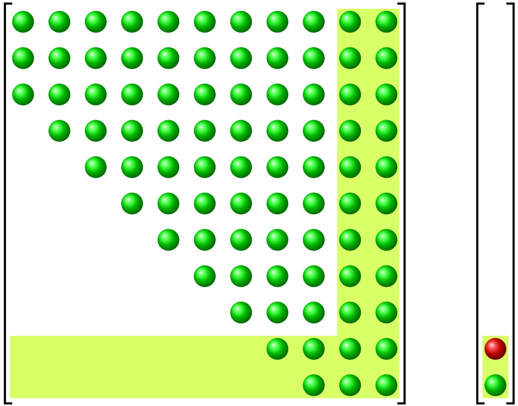
Businger's algorithm 25



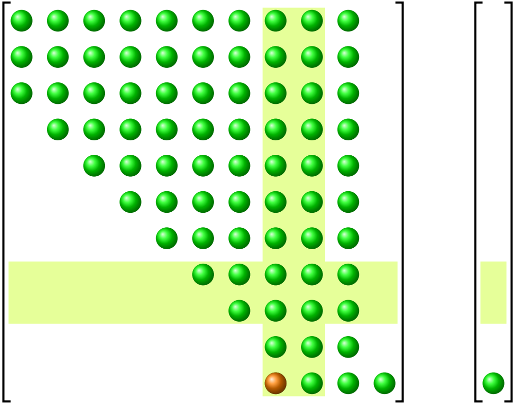
Businger's algorithm 26



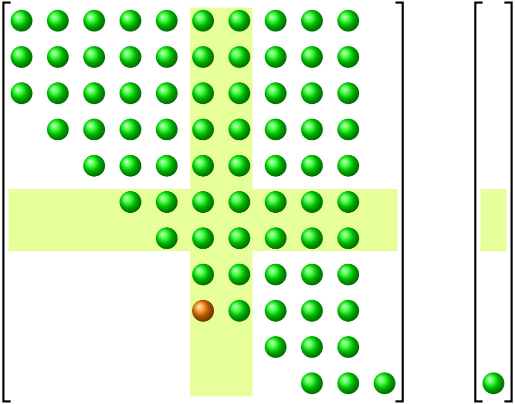
Businger's algorithm 27



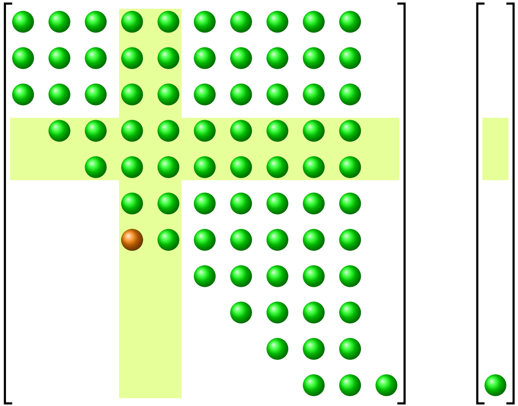
Businger's algorithm 28



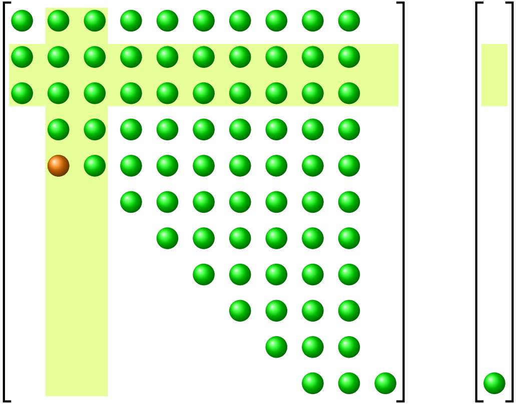
Businger's algorithm 29



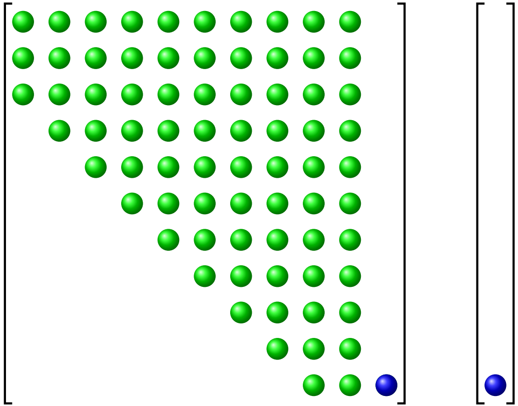
Businger's algorithm 30



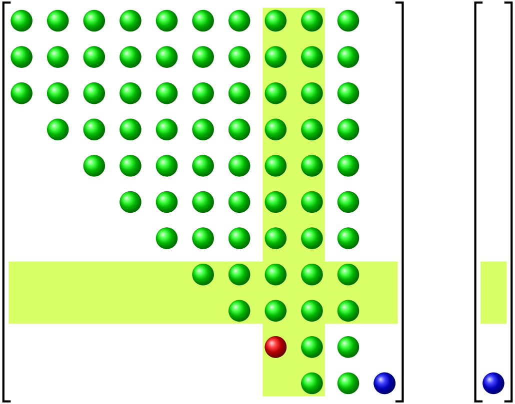
Businger's algorithm 31



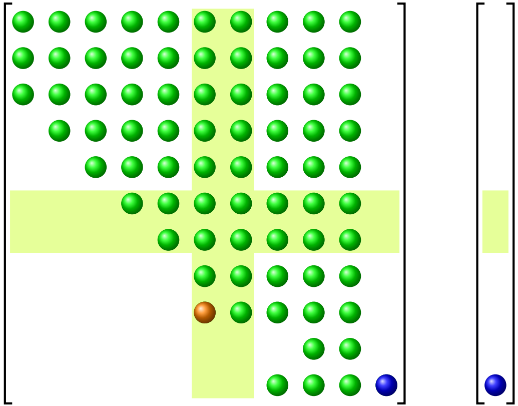
Businger's algorithm 32



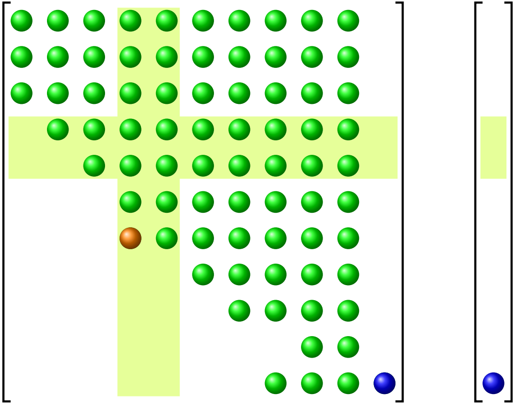
Businger's algorithm 33



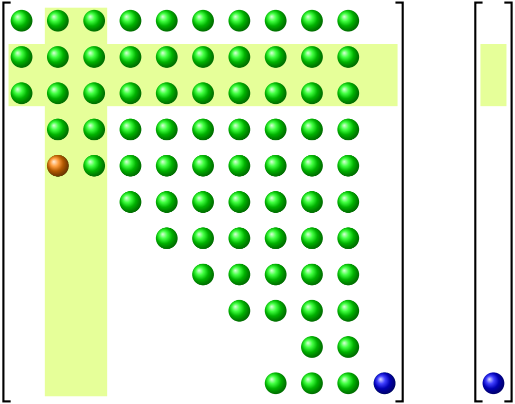
Businger's algorithm 34



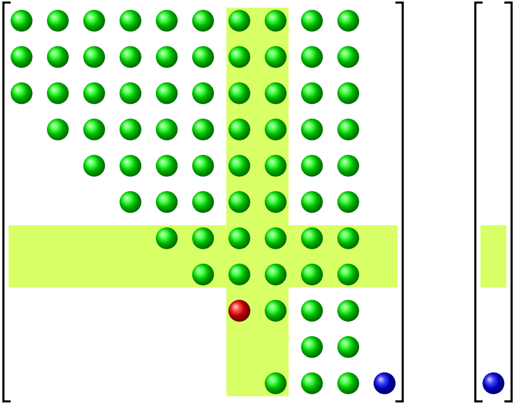
Businger's algorithm 35



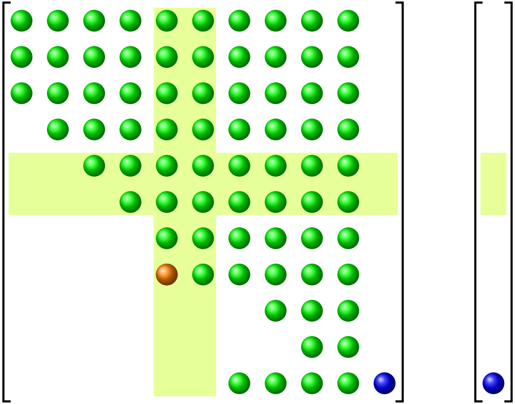
Businger's algorithm 36



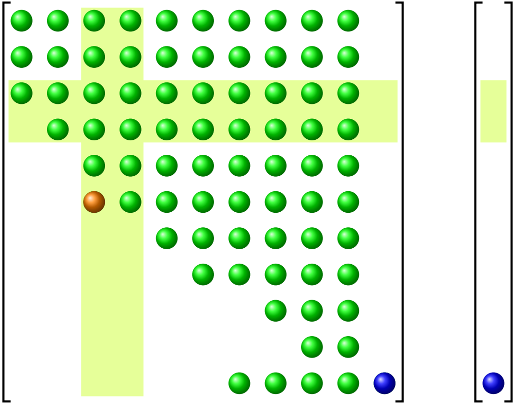
Businger's algorithm 37



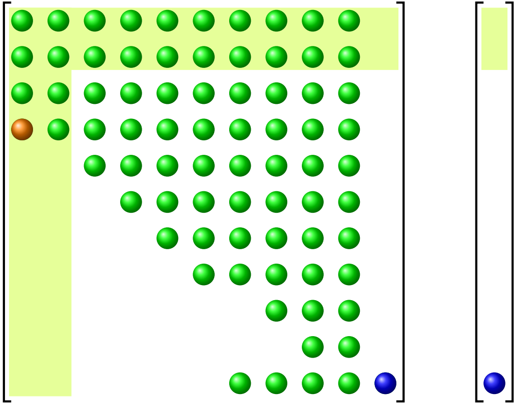
Businger's algorithm 38



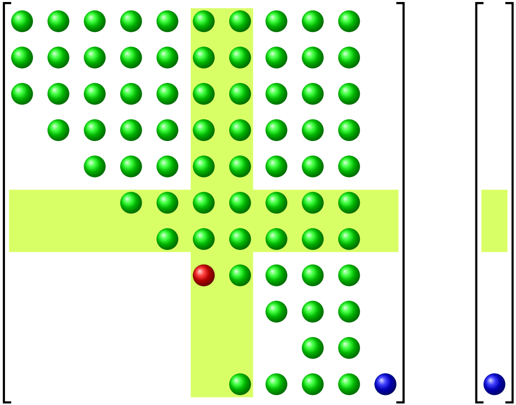
Businger's algorithm 39



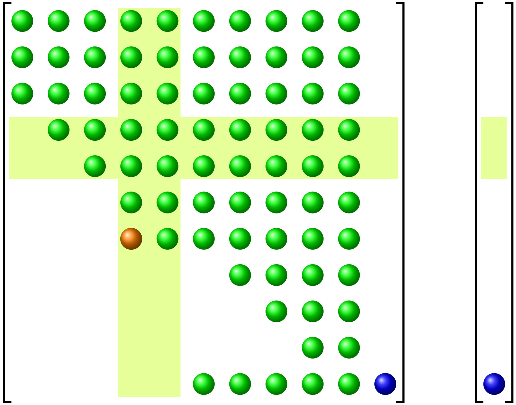
Businger's algorithm 40



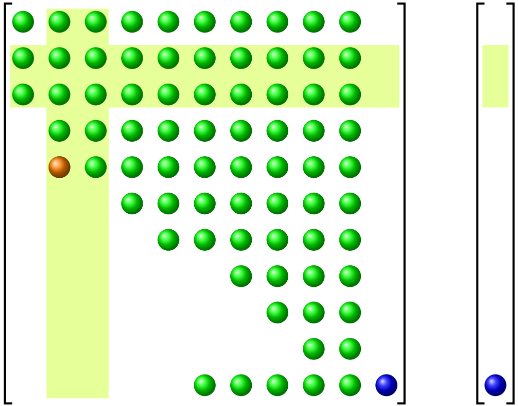
Businger's algorithm 41



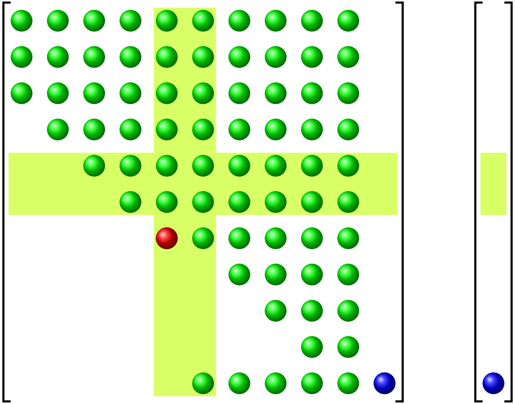
Businger's algorithm 42



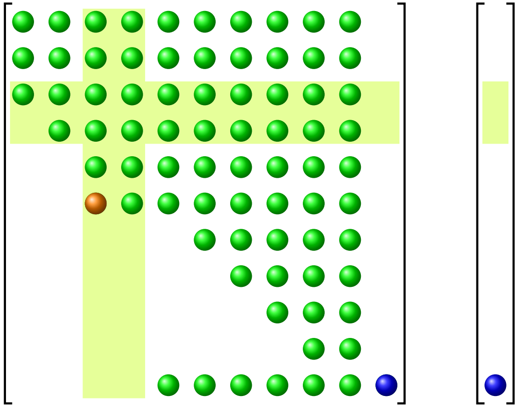
Businger's algorithm 43



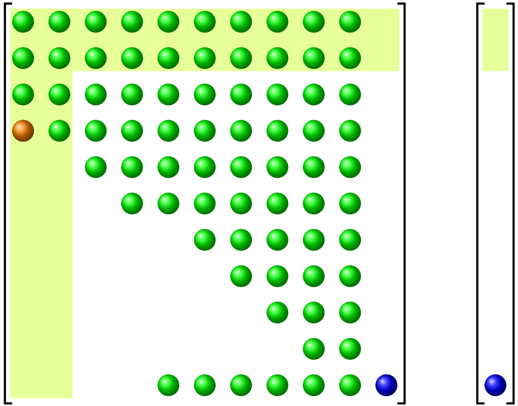
Businger's algorithm 44



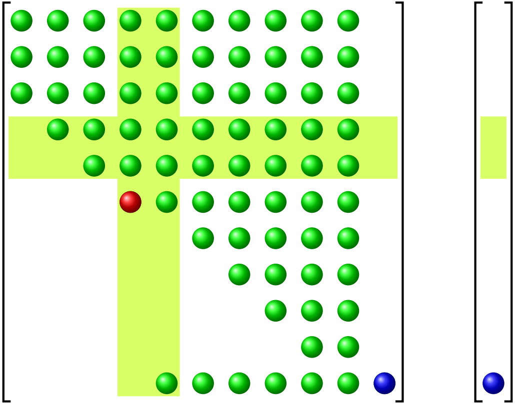
Businger's algorithm 45



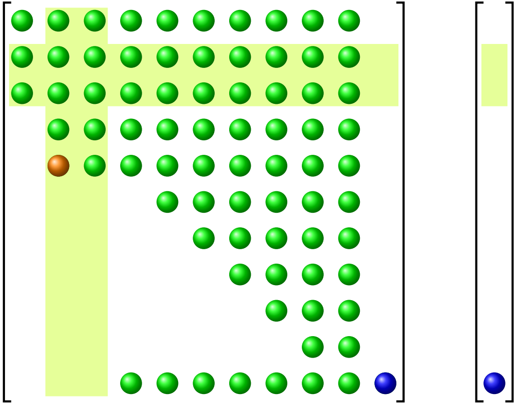
Businger's algorithm 46



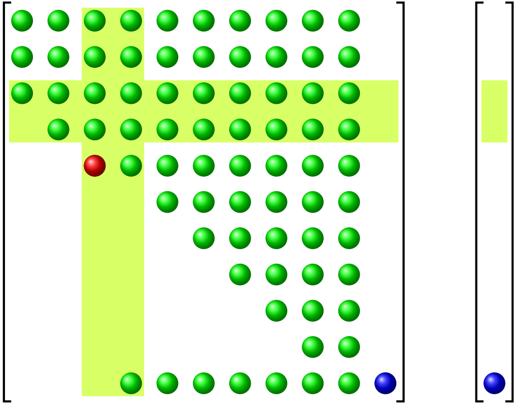
Businger's algorithm 47



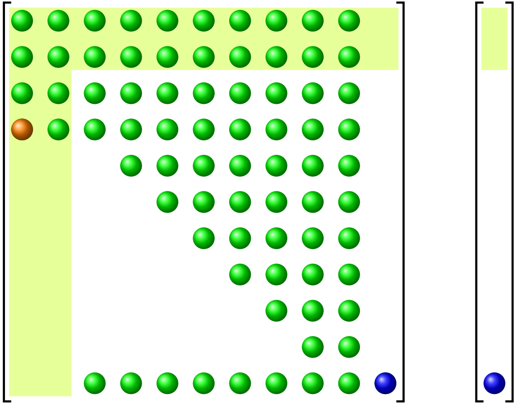
Businger's algorithm 48



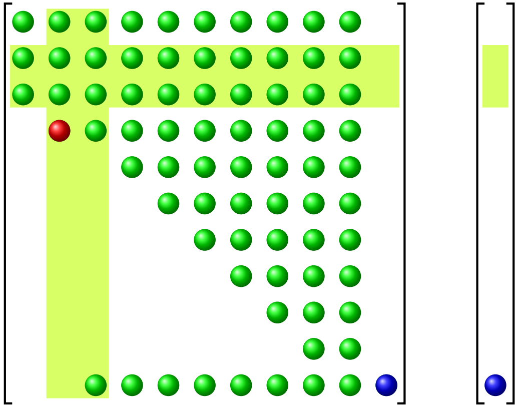
Businger's algorithm 49



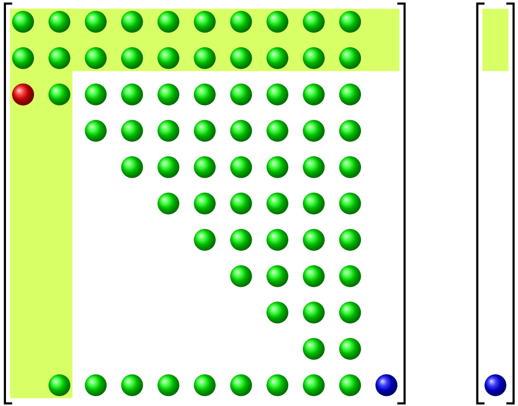
Businger's algorithm 50



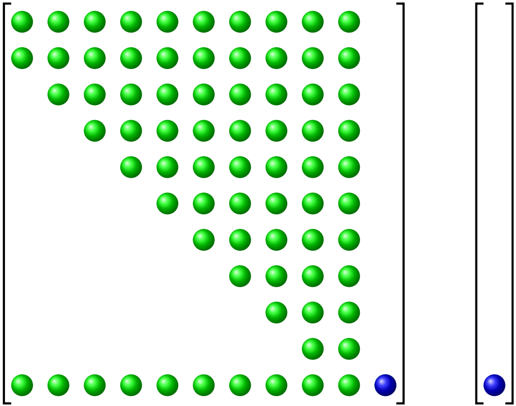
Businger's algorithm 51



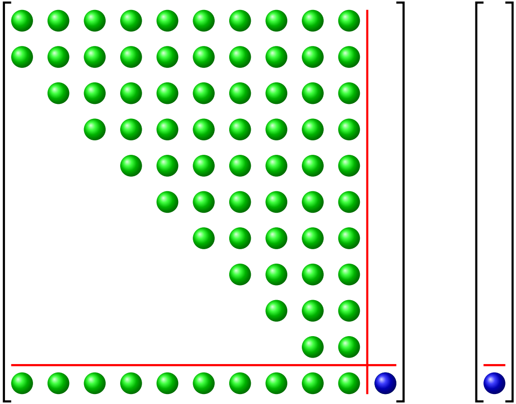
Businger's algorithm 52



Businger's algorithm 53



Businger's algorithm 54



Businger's algorithm: conclusions

The Businger's algorithm is stable but too expensive requiring $\frac{1}{2}(n-1)(n-2)+1$ Givens rotations $\Rightarrow O(n^3)$ flops!!

Alternate ways of deflation

Implicit Q theorem revisited

Let $H \in \mathbb{R}^{n\times n}$ irreducible Hessenberg with eigenvalue λ. Then

1. H has an "essentially unique" normalized eigenvector $\mathbf{x}$ corresponding to λ:

$$H\mathbf{x} = \lambda\mathbf{x}, \quad \|\mathbf{x}\|_2 = 1$$

   and its last component $x_n \neq 0$;

2. there is an "essentially unique" sequence of Givens rotations $G_{n-1,n}, \dots, G_{1,2}$ whose product

$$Q := G_{1,2}G_{2,3}\cdots G_{n-1,n}$$

   transforms the pair $(H, \mathbf{x})$ to a similar one

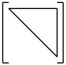
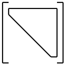
$$(\tilde{H}, \tilde{\mathbf{x}}) := (QHQ^T, Q\mathbf{x})$$

   where

$$\tilde{\mathbf{x}} = \alpha\mathbf{e}_1, \quad |\alpha| = \|\mathbf{x}\|_2 = 1$$

$$\tilde{H}\mathbf{e}_1 = \lambda\mathbf{e}_1, \quad \tilde{H} \text{ in Hessenberg form.}$$

Implicit Q theorem revisited, proof

1. The fact that $\mathbf{x}$ is unique follows from
$$(H - \lambda I)\mathbf{x} = 0, \quad \|\mathbf{x}\|_2 = 1,$$
where $(H - \lambda I)$ has rank $n - 1$ since H irreducible Hessenberg and $x_n \neq 0$.
2. The reduction of $\mathbf{x}$ to $\tilde{\mathbf{x}} = Q\mathbf{x} = \alpha\mathbf{e}_1$ requires a sequence of Givens rotations
$$G_{i-1,i} \in \mathbb{R}^{n \times n}, \quad i = n, n-1, \ldots, 2, \tag{1}$$
in order to eliminate the entries x_i, $i = n, n-1, \ldots, 2$ of the vector $\mathbf{x}$. These are the same rotations that reduce
$$(H - \lambda I)Q^T =$$

$$= R \ \text{triangular}$$
$$Q(H - \lambda I)Q^T =$$

$$= \tilde{H} - \lambda I \ \text{Hessenberg}$$
Since $\mathbf{x} = \alpha Q^T\mathbf{e}_1$, $\alpha = \pm 1$, we have
$R\mathbf{e}_1 = 0, \quad (\tilde{H} - \lambda I)\mathbf{e}_1 = 0 \Rightarrow \tilde{H}\mathbf{e}_1 = \lambda\mathbf{e}_1.$

Implicit Q theorem revisited, remark

The implicit Q theorem is closely related to this lemma. It explains that the transformation Q can also be determined from the first rotation $G_{n-1,n}$ that computes

$$\begin{bmatrix} h_{n,n-1}, & h_{n,n} - \lambda \end{bmatrix} G_{n-1,n}^T = \begin{bmatrix} 0 & x \end{bmatrix}$$

and from the fact that QHQ^T is still Hessenberg. This is known as "chasing the bulge" tecknique (Watkins '07).

Implicit Q theorem revisited

- In theory, the backward (forward) QR with perfect shift computes the same Hessenberg matrix computed transforming $x(y)$ to $e_1(e_n)$.
  In practice, the two techniques have a different behavior.

Implicit Q theorem revisited

- In theory, the backward (forward) QR with perfect shift computes the same Hessenberg matrix computed transforming $x(y)$ to $e_1(e_n)$.
  In practice, the two techniques have a different behavior.
- In particular, the backward (forward) QR with perfect shift, due to small entries in the subdiagonal, does not yield the exact eigenvalue (blurred shift, Watkins '07).

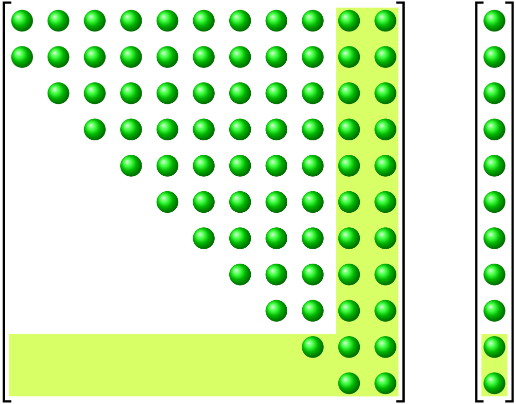
Implicit Q theorem revisited

- In theory, the backward (forward) QR with perfect shift computes the same Hessenberg matrix computed transforming $x(y)$ to $e_1(e_n)$.
  In practice, the two techniques have a different behavior.
- In particular, the backward (forward) QR with perfect shift, due to small entries in the subdiagonal, does not yield the exact eigenvalue (blurred shift, Watkins '07).
- It can be shown that, if the last two (first) entries of the right (left) eigenvectors are *"large enough"* ($\approx \frac{1}{n^2}$) the QR method with perfect shift applied to the eigenvector $x(y)$ works properly.
  If the last two (first) entries of the right (left) eigenvectors are *"small"* the lower part below the subdiagonal is filled by a rank-one structured matrix.

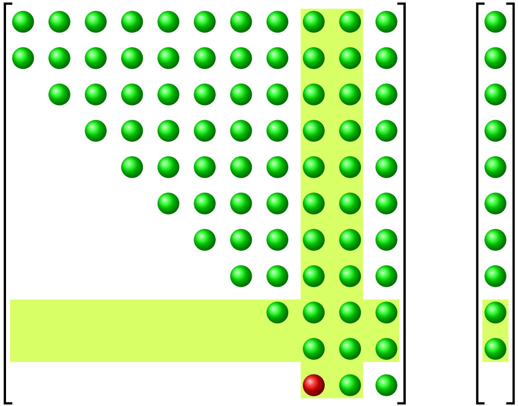
Implicit Q theorem revisited

- In theory, the backward (forward) QR with perfect shift computes the same Hessenberg matrix computed transforming $x(y)$ to $e_1(e_n)$.
  In practice, the two techniques have a different behavior.
- In particular, the backward (forward) QR with perfect shift, due to small entries in the subdiagonal, does not yield the exact eigenvalue (blurred shift, Watkins '07).
- It can be shown that, if the last two (first) entries of the right (left) eigenvectors are *"large enough"* ($\approx \frac{1}{n^2}$) the QR method with perfect shift applied to the eigenvector $x(y)$ works properly.
  If the last two (first) entries of the right (left) eigenvectors are *"small"* the lower part below the subdiagonal is filled by a rank-one structured matrix.
- In the sequel we will show how to overcome this problem.

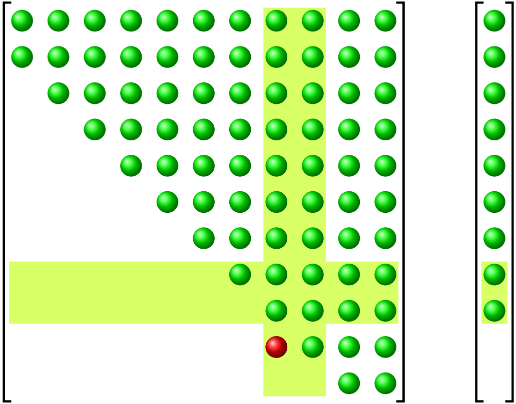
Perfect shift backward 1



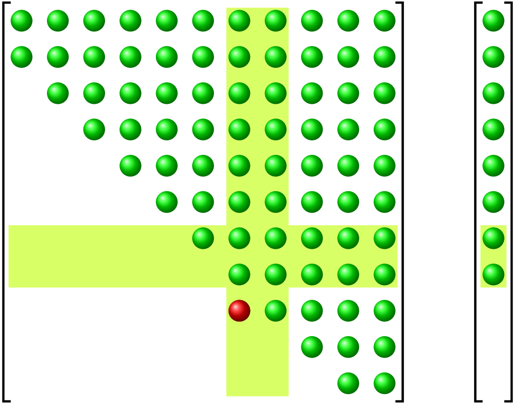
Perfect shift backward 2



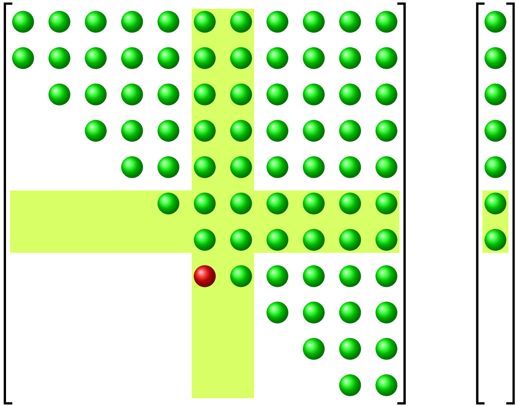
Perfect shift backward 3



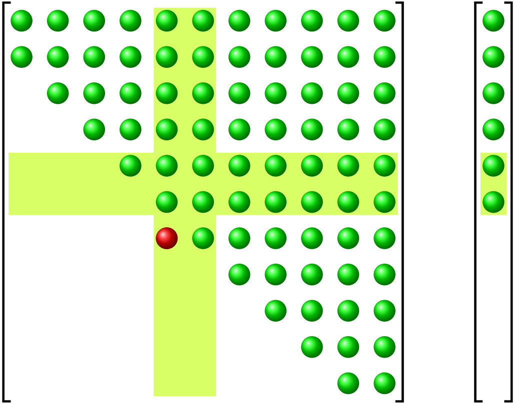
Perfect shift backward 4



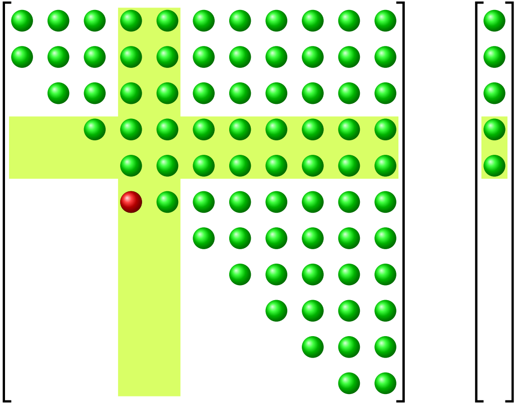
Perfect shift backward 5



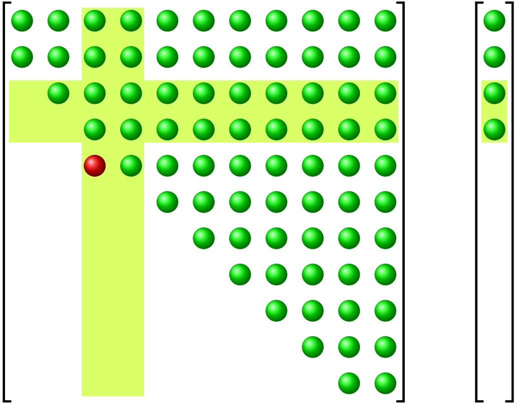
Perfect shift backward 6



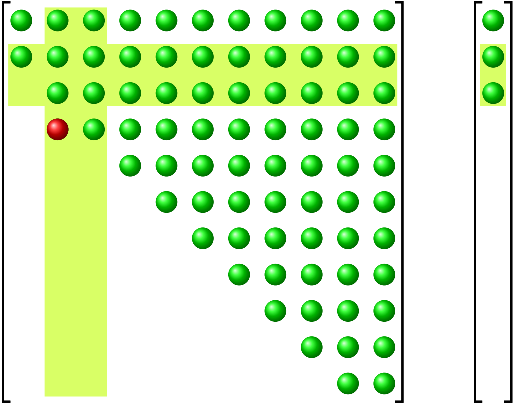
Perfect shift backward 7



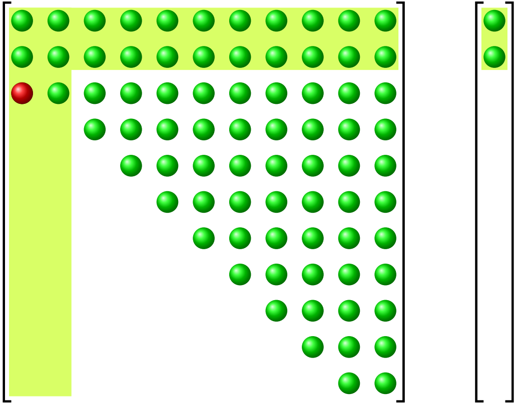
Perfect shift backward 8



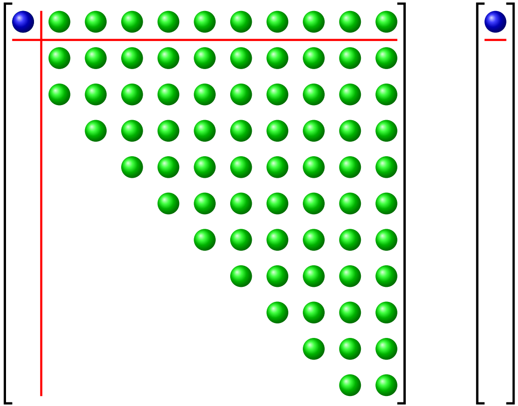
Perfect shift backward 9



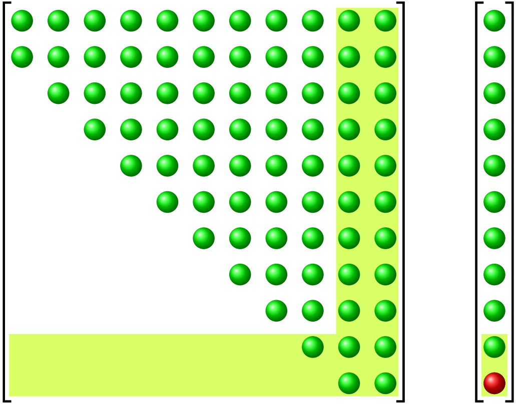
Perfect shift backward 10



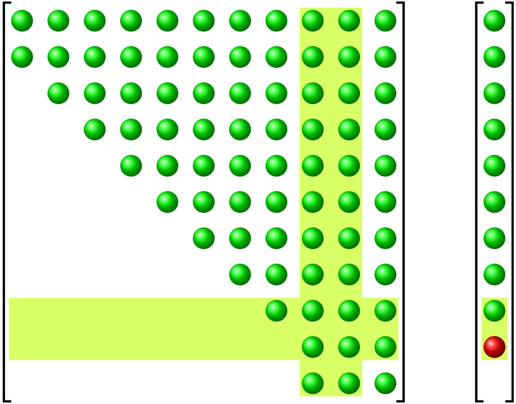
Perfect shift backward 11



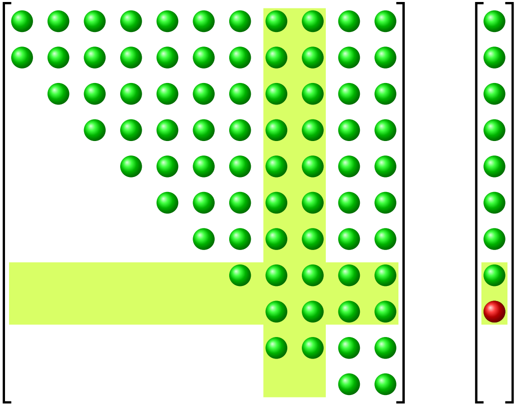
Perfect shift x 1



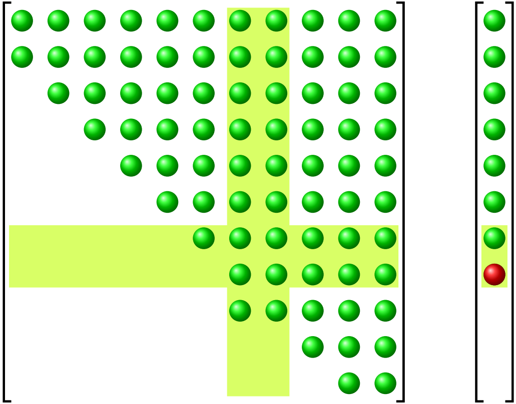
Perfect shift x 2



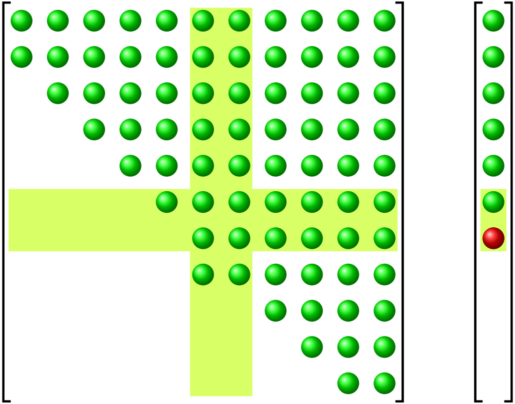
Perfect shift x 3



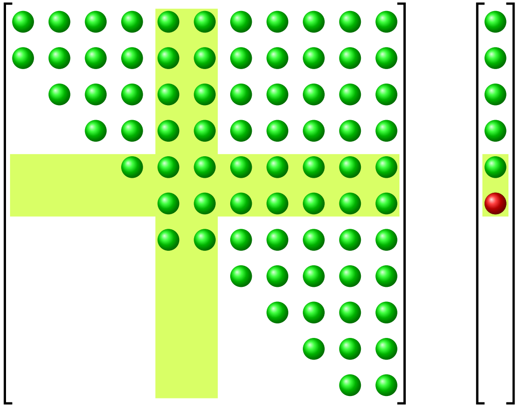
Perfect shift x 4



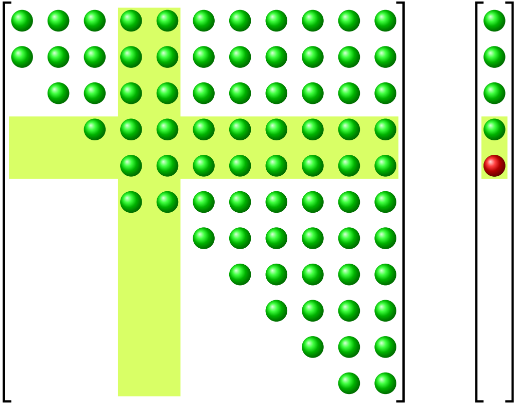
Perfect shift x 5



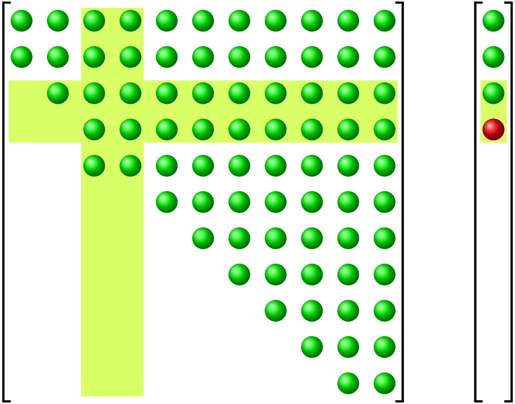
Perfect shift x 6



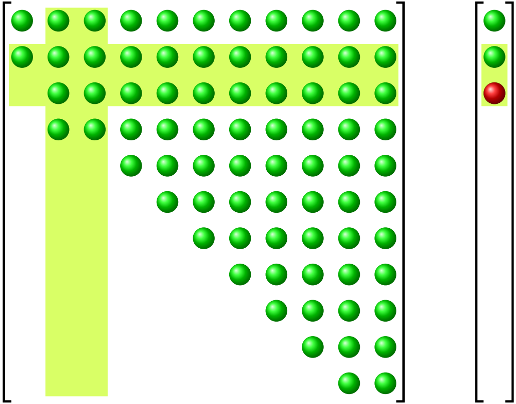
Perfect shift x 7



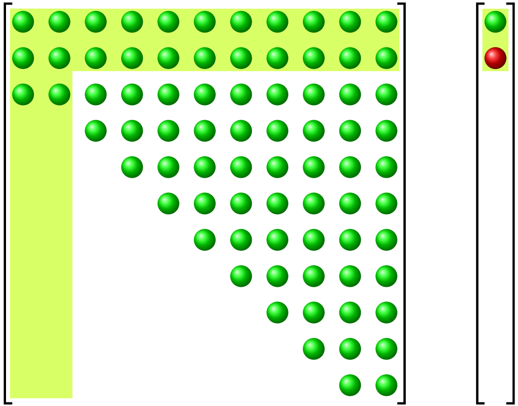
Perfect shift x 8



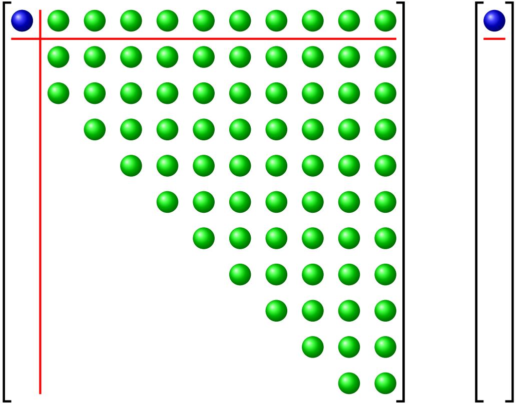
Perfect shift x 9



Perfect shift x 10



Perfect shift x 11



To compute the eigenvalues of A_{20} we proceed in the following way:

- Reduce A_{20} to upper Hessenberg form ($O(n^3)$ flops;

To compute the eigenvalues of A_{20} we proceed in the following way:

- Reduce A_{20} to upper Hessenberg form ($O(n^3)$ flops;
- Compute the smallest singular triplet (σ_i, u_i, v_i) of $A_i - \lambda_k I$; ($O(n^2)$ flops);

To compute the eigenvalues of A_{20} we proceed in the following way:

- Reduce A_{20} to upper Hessenberg form ($O(n^3)$ flops;
- Compute the smallest singular triplet (σ_i, u_i, v_i) of $A_i - \lambda_k I$; ($O(n^2)$ flops);
- Check whether the last two (first) entries of the right (left) eigenvectors are *"large enough"* ($\approx \frac{1}{n^2}$)
  If yes apply the revisited Q theorem on the right (left) eigenvector and deflate the matrix

$$\texttt{eig}(A_{17}) = \begin{bmatrix} 1.999737310251318e+00+0.000000000000000e+00i \\ 1.999975551553143e+00+0.000000000000000e+00i \\ 1.999999847581437e+00-2.628426245622164e-04i \\ 1.999999847581437e+00+2.628426245622164e-04i \\ 2.000012224221711e+00-2.116794988907987e-05i \\ 2.000012224221711e+00+2.116794988907987e-05i \\ 2.000262994589237e+00+0.000000000000000e+00i \\ 2.999016519182911e+00+0.000000000000000e+00i \\ 2.999360887826527e+00-4.594596788891278e-04i \\ 2.999360887826527e+00+4.594596788891278e-04i \\ 2.999696051382021e+00-9.353280336779713e-04i \\ 2.999696051382021e+00+9.353280336779713e-04i \\ 3.000239521327504e+00-7.576797639221387e-04i \\ 3.000239521327504e+00+7.576797639221387e-04i \\ 3.000795689046296e+00-5.781723116350447e-04i \\ 3.000795689046296e+00+5.781723116350447e-04i \\ 3.000799181652402e+00+0.000000000000000e+00i \end{bmatrix}$$

The eigenvalues of A_{17} are 2 and 3.

perfect shift left eigenvector

$$A_{17}^{(1)} = Q_1 A_{17} Q_1^T,$$

$$y = \begin{bmatrix} 1.0232e-16 \\ -2.7123e-16 \\ -8.8156e-16 \\ -1.0473e-15 \\ 1.1862e-15 \\ 2.0485e-15 \\ -1.5446e-14 \\ -3.0139e-03 \\ -1.1602e-02 \\ -9.8263e-03 \\ 1.8285e-02 \\ 5.7980e-02 \\ 1.0316e-01 \\ 7.0550e-01 \\ 6.5006e-01 \\ 2.5102e-01 \\ -4.5752e-02 \end{bmatrix}, \quad d_{-2} = \begin{bmatrix} 5.2677e-02 \\ 3.8240e-02 \\ 8.2147e-02 \\ 8.0910e-03 \\ -1.6409e-02 \\ -9.3025e-04 \\ -3.8075e-15 \\ -1.0401e-16 \\ -8.5684e-17 \\ -8.7287e-16 \\ -5.6910e-16 \\ 1.6770e-16 \\ -1.2485e-17 \\ -4.4618e-16 \\ 3.2289e-17 \\ 1.4691e-17 \end{bmatrix}, \quad w = \begin{bmatrix} -2.7297e-16 \\ -5.6114e-16 \\ 4.8380e-16 \\ 2.6044e-16 \\ -7.1543e-18 \\ -4.6883e-17 \\ 1.9433e-16 \\ -2.3211e-16 \\ -7.2482e-17 \\ -2.9224e-15 \\ 2.4195e-16 \\ -6.8281e-16 \\ 8.7837e-16 \\ 1.4132e-15 \\ -1.3296e-16 \\ 4.9960e-16 \\ 2.0000e+00 \end{bmatrix},$$

$$d_{-2} = \mathtt{diag}(A_{18}^{(1)}, -2)$$

$$w^T = (A_{17}^{(1)}(17, :)$$

$$A_{17}^{(1)}(17, 17) = 2.000000000000001e+00,$$

$$\|\mathtt{tril}(A_{17}^{(1)}, -2)\|_2 = 6.8942e-01.$$

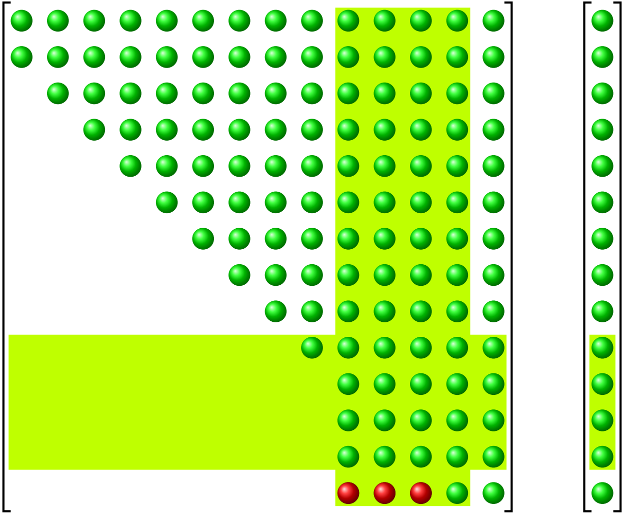
perfect shift right eigenvector

$$A_{17}^{(2)} = Q_2 A_{17} Q_2^T,$$

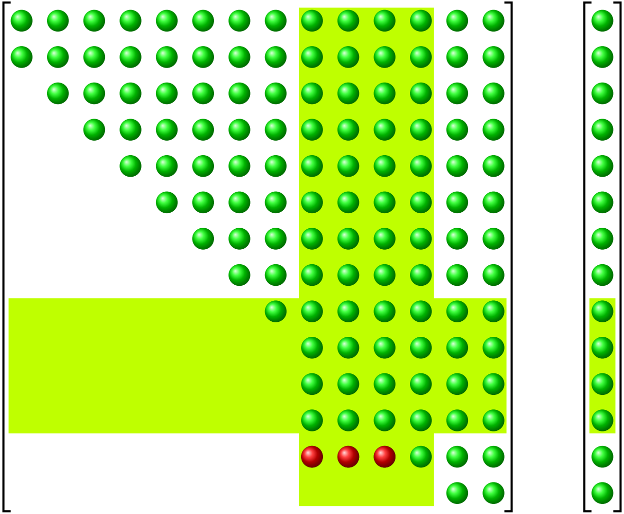
$$x = \begin{bmatrix} -3.2644e-01 \\ -2.1155e-01 \\ -1.1201e-01 \\ -3.6169e-01 \\ -5.8776e-01 \\ -5.8387e-01 \\ 1.1730e-02 \\ -7.2694e-02 \\ 1.0267e-01 \\ 4.1093e-02 \\ 1.4175e-02 \\ 3.3022e-02 \\ -2.2944e-03 \\ -7.9542e-04 \\ -2.8838e-08 \\ 4.8694e-15 \\ 4.4041e-16 \end{bmatrix}, \quad d_{-2} = \begin{bmatrix} 2.7756e-16 \\ -1.1494e-15 \\ -4.1284e-16 \\ -2.0039e-16 \\ 1.6412e-15 \\ 2.5694e-16 \\ -1.4412e-15 \\ 6.7440e-17 \\ 4.4265e-15 \\ -1.3015e-15 \\ -6.0012e-16 \\ 1.9700e-15 \\ 2.4631e-13 \\ 5.9216e-13 \\ -1.3890e-08 \end{bmatrix}, \quad z = \begin{bmatrix} 2.0000e+00 \\ 4.4409e-16 \\ 2.7756e-16 \\ 2.4905e-16 \\ -1.4008e-15 \\ -3.6529e-16 \\ 1.1118e-15 \\ 2.5818e-15 \\ 1.5292e-15 \\ -1.9832e-15 \\ -1.5952e-15 \\ 5.6414e-16 \\ -8.9336e-16 \\ 1.7634e-16 \\ 3.2376e-16 \\ 1.8241e-16 \\ 1.3688e-15 \end{bmatrix},$$

$$d_{-2} = \mathtt{diag}(A_{17}^{(1)}, -2)$$
$$z = (A_{17}^{(1)}(:, 1)$$
$$A_{17}^{(2)}(17, 17) = 2.000000000000000e+00,$$
$$\|\mathtt{tril}(A_{17}^{(2)}, -2)\|_2 = 1.3890e-08.$$

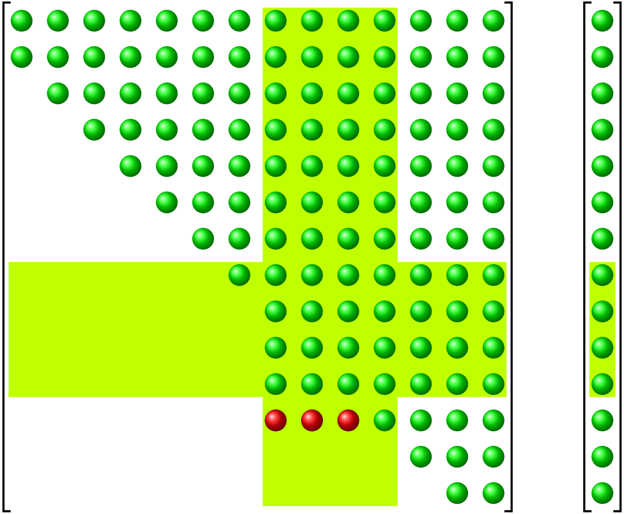
Multishift backward algorithm 1



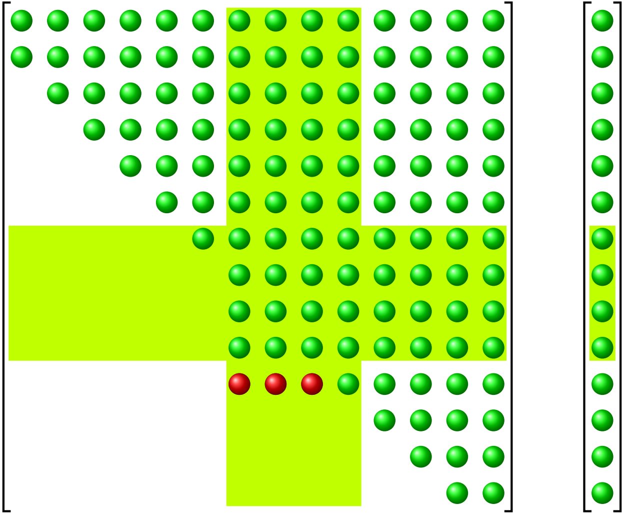
Multishift backward algorithm 2



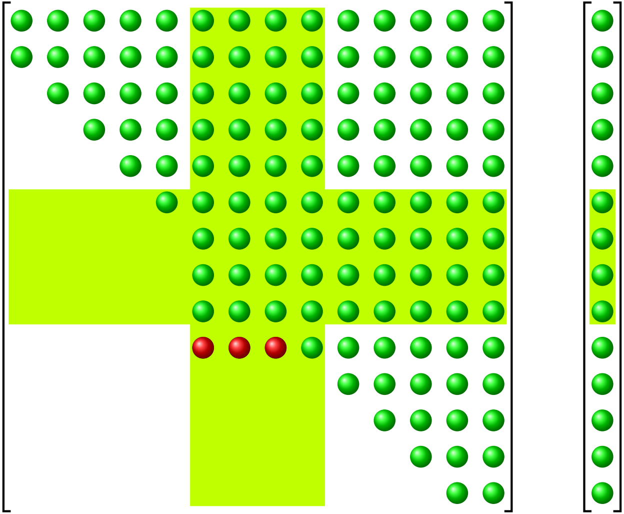
Multishift backward algorithm 3



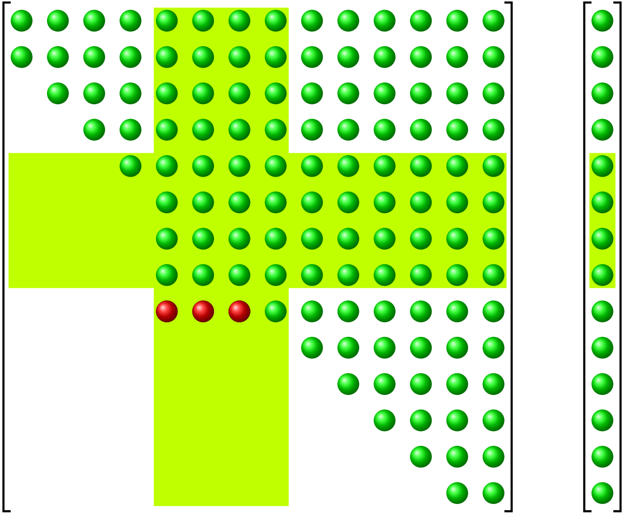
Multishift backward algorithm 4



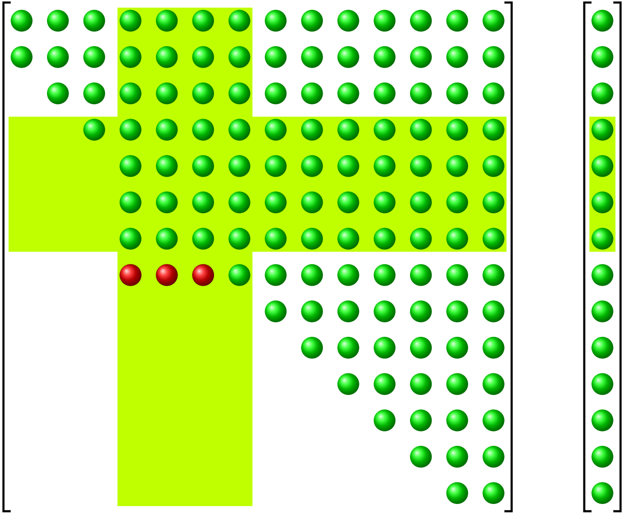
Multishift backward algorithm 5



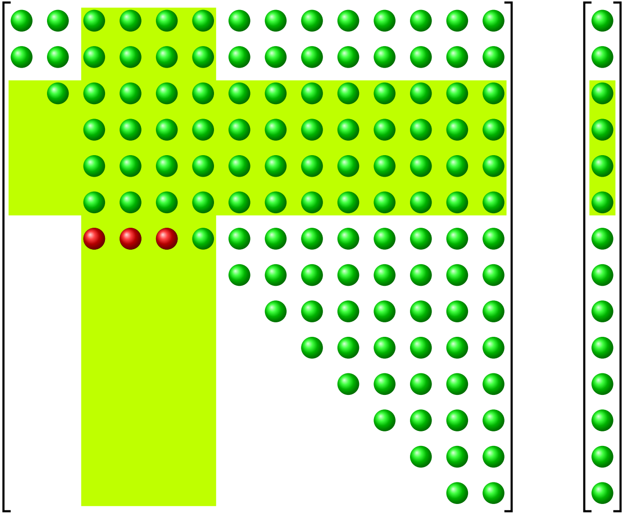
Multishift backward algorithm 6



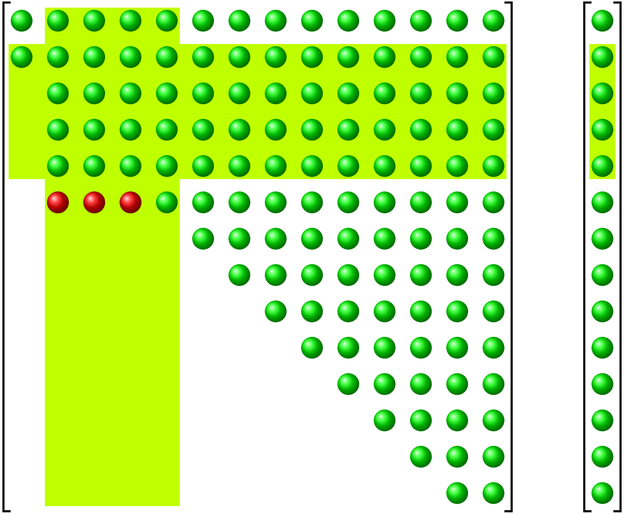
Multishift backward algorithm 7



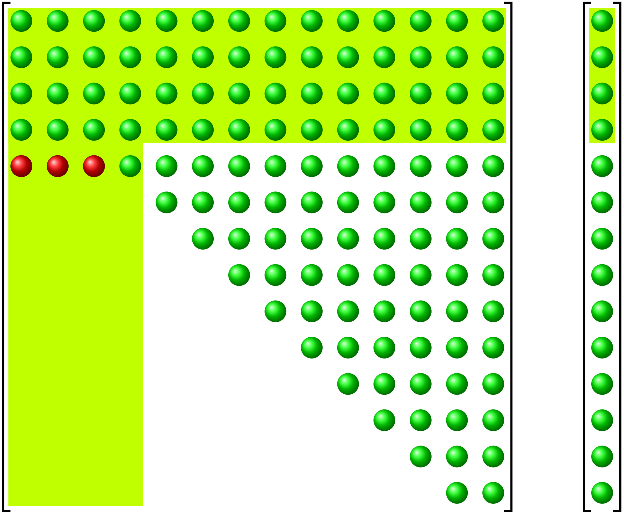
Multishift backward algorithm 8



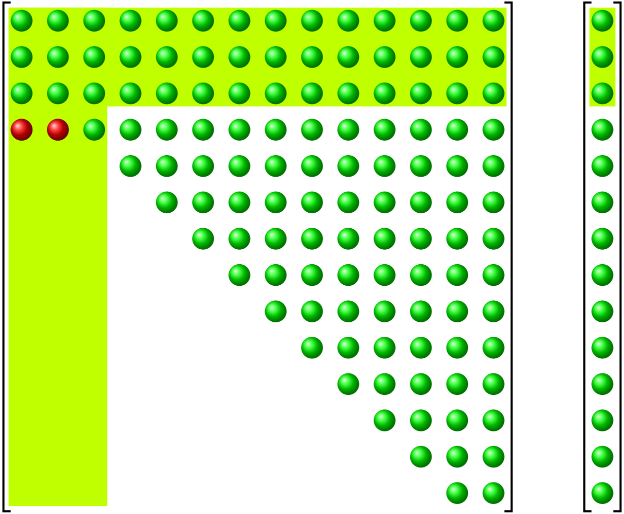
Multishift backward algorithm 9



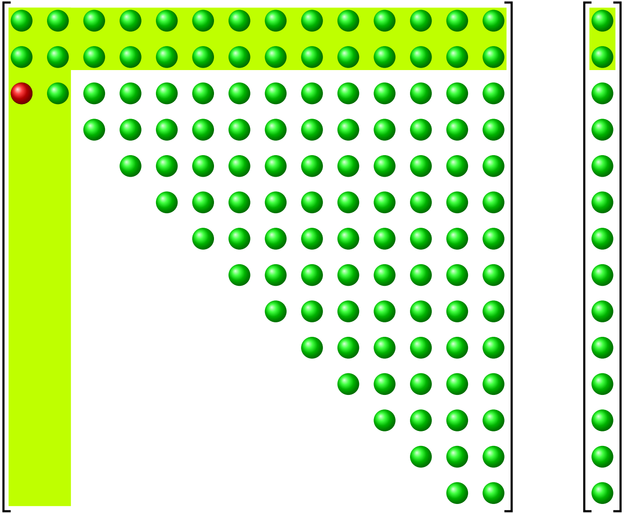
Multishift backward algorithm 10



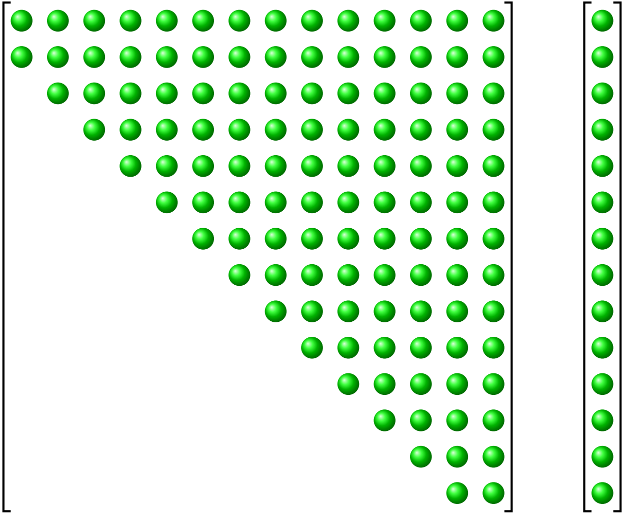
Multishift backward algorithm 11



Multishift backward algorithm 12



Multishift backward algorithm 13



perfect shift right eigenvector modified

$$A_{18}^{(2)} = Q_2 A_{18} Q_2^T,$$

$$x = \begin{bmatrix} 1.6219e-01 \\ 9.1224e-01 \\ -2.4749e-01 \\ 2.2343e-01 \\ -1.3038e-01 \\ 1.7690e-02 \\ 5.2873e-02 \\ 7.8050e-02 \\ -3.6873e-02 \\ -4.4924e-02 \\ 6.5400e-03 \\ -1.9780e-02 \\ 4.8091e-04 \\ 1.5581e-02 \\ 8.6396e-03 \\ -9.1965e-04 \\ 2.1808e-03 \end{bmatrix}, \quad d_{-2} = \begin{bmatrix} -1.9223e-16 \\ 3.3762e-17 \\ -5.7734e-16 \\ 7.3044e-16 \\ 1.3134e-15 \\ 8.8303e-16 \\ -2.0834e-16 \\ -1.9033e-15 \\ 1.0139e-15 \\ 6.6845e-16 \\ 1.2260e-14 \\ 1.9789e-15 \\ -3.2387e-15 \\ -6.3310e-16 \\ -5.2576e-15 \end{bmatrix}, \quad z = \begin{bmatrix} 2.0000e+00 \\ -1.1102e-16 \\ -1.9223e-16 \\ 2.0541e-16 \\ -2.3492e-16 \\ -3.1452e-16 \\ 2.9004e-16 \\ -1.3622e-16 \\ -1.5520e-16 \\ -4.1542e-16 \\ 8.4472e-17 \\ -6.1413e-17 \\ -3.9564e-16 \\ 2.2257e-16 \\ 7.8689e-17 \\ -4.2538e-16 \\ -5.4326e-17 \end{bmatrix},$$

$$d_{-2} = \mathtt{diag}(A_{17}^{(1)}, -2)$$
$$z = (A_{17}^{(1)}(:, 1)$$
$$A_{18}^{(2)}(18, 18) = 2.00000000000000e+00,$$
$$\|\mathtt{tril}(A_{17}^{(2)}, -2)\|_2 = 2.7224e-14.$$

New algorithm for computing the generalized null–space

First step: compute the "*reverse*" Hessenberg reduction of A :

$$A = Q_1 H Q_1^H, \quad H = \begin{bmatrix} H_1 & * & \cdots & * \\ & H_2 & \ddots & * \\ & & \ddots & * \\ & & & H_j \end{bmatrix},$$

$H_i, \ i = 1, \ldots, j$, irreducible upper Hessenberg matrices.

New algorithm for computing the generalized null–space

Theorem

Let $m(\lambda) = \prod_{i=1}^{\ell}(\lambda - \lambda_i)^{k_i}$ *be the monic minimal polynomial of a matrix* A*, and let* $d := \sum_{i=1}^{\ell} k_i$ *be its degree. Then*

1. *the Krylov subspace* $\mathcal{K}_k(A, b) = \mathrm{Im}\left[b, Ab, \ldots, A^{k-1}b\right]$ *has dimension bounded by* $\min(k, d)$,
2. *this upper bound is reached for almost any vector* b*, i.e. it is* generic,
3. *such a Krylov subspace of maximal dimension* d *is an invariant subspace of* A *corresponding to a largest Jordan block of each eigenvalue, and the vector* b *is its cyclic generator.*

New algorithm for computing the generalized null–space

$$A = Q_1 H Q_1^H, \quad H = \begin{bmatrix} H_1 & * & \cdots & * \\ & H_2 & \ddots & * \\ & & \ddots & * \\ & & & H_j \end{bmatrix}.$$

Second step: Check whether $H_i,\ i = 1, \ldots, j$, are singular (via inverse iteration). If H_i is singular, deflate the 0 eignevalue by the perfect shift technique.

New algorithm

Third step: Reduce the block Hessenberg matrix to upper *echelon* form, i.e., multiply $\hat{H}$ to the right by a unitary matrix U so that, if $\mathrm{rank}(\hat{H}) = \hat{n} \leq n$, the first $n - \hat{n}$ columns of $\hat{H}U$ are $\mathbf{0}$, where $\hat{n}$ are the number of singular blocks H_j.

Example

$$\left[\begin{array}{ccc|cccccccccc}
0 & 0 & 0 & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\
 & & & \times & \times & \times & \times & \times & \times & \times & \times & \times & \times \\
 & & & & \times & \times & \times & \times & \times & \times & \times & \times & \times \\
 & & & & \times & \times & \times & \times & \times & \times & \times & \times & \times \\
 & & & & & \times & \times & \times & \times & \times & \times & \times & \times \\
 & & & & & & \times & \times & \times & \times & \times & \times & \times \\
 & & & & & & \times & \times & \times & \times & \times & \times & \times \\
 & & & & & & & \times & \times & \times & \times & \times & \times \\
 & & & & & & & & \times & \times & \times & \times & \times \\
 & & & & & & & & & \times & \times & \times & \times \\
 & & & & & & & & & & \times & \times & \times \\
 & & & & & & & & & & & \times & \times \\
 & & & & & & & & & & & & \times
\end{array}\right]$$

Second Example

We consider a matrix of order 18 considered in the paper

N. Guglielmi, M.L. Overton, G.W. Stewart, An Efficient Algorithm for Computing the Generalized Null Space Decomposition, **SIAM J. Matrix Anal. Appl.** 36, 38–54, 2015.

Second Example

We consider a matrix of order 18 considered in the paper

N. Guglielmi, M.L. Overton, G.W. Stewart, An Efficient Algorithm for Computing the Generalized Null Space Decomposition, **SIAM J. Matrix Anal. Appl.** 36, 38–54, 2015.

$$\sigma(A) = \begin{bmatrix} 1.9288e+00 \\ 1.3561e+00 \\ 8.7206e-01 \\ 6.8301e-01 \\ 6.3444e-01 \\ 3.1994e-01 \\ 3.1287e-01 \\ 1.6030e-01 \\ 3.1547e-02 \\ 1.7171e-02 \\ 1.4103e-02 \\ 1.1660e-02 \\ 8.8228e-03 \\ 8.4642e-03 \\ 7.5872e-07 \\ 6.5816e-07 \\ 9.1614e-11 \\ 2.5727e-16 \end{bmatrix}$$

Second Example

We consider a matrix of order 18 considered in the paper

N. Guglielmi, M.L. Overton, G.W. Stewart, An Efficient Algorithm for Computing the Generalized Null Space Decomposition, **SIAM J. Matrix Anal. Appl.** 36, 38–54, 2015.

$$\sigma(A) = \begin{bmatrix} 1.9288e+00 \\ 1.3561e+00 \\ 8.7206e-01 \\ 6.8301e-01 \\ 6.3444e-01 \\ 3.1994e-01 \\ 3.1287e-01 \\ 1.6030e-01 \\ 3.1547e-02 \\ 1.7171e-02 \\ 1.4103e-02 \\ 1.1660e-02 \\ 8.8228e-03 \\ 8.4642e-03 \\ 7.5872e-07 \\ 6.5816e-07 \\ 9.1614e-11 \\ 2.5727e-16 \end{bmatrix} \qquad \lambda(A) = \begin{bmatrix} 1.0000e+00+0.0000e+00i \\ 2.5000e-01+0.0000e+00i \\ 2.5000e-01+0.0000e+00i \\ -1.7684e-04+5.6050e-05i \\ -1.7684e-04-5.6050e-05i \\ -1.4499e-04+0.0000e+00i \\ -3.3473e-05+0.0000e+00i \\ -1.6317e-04+0.0000e+00i \\ 2.3786e-05+1.7410e-04i \\ 2.3786e-05-1.7410e-04i \\ 8.4030e-05+0.0000e+00i \\ 1.8033e-04+4.3683e-05i \\ 1.8033e-04-4.3683e-05i \\ 1.8292e-04+7.0199e-06i \\ 1.8292e-04-7.0199e-06i \\ 6.2500e-02+0.0000e+00i \\ 6.2500e-02+0.0000e+00i \\ 6.2500e-02+0.0000e+00i \end{bmatrix}$$

Second Example

We consider a matrix of order 18 considered in the paper

N. Guglielmi, M.L. Overton, G.W. Stewart, An Efficient Algorithm for Computing the Generalized Null Space Decomposition, **SIAM J. Matrix Anal. Appl.** 36, 38–54, 2015.

$$\sigma(A) = \begin{bmatrix} 1.9288e+00 \\ 1.3561e+00 \\ 8.7206e-01 \\ 6.8301e-01 \\ 6.3444e-01 \\ 3.1994e-01 \\ 3.1287e-01 \\ 1.6030e-01 \\ 3.1547e-02 \\ 1.7171e-02 \\ 1.4103e-02 \\ 1.1660e-02 \\ 8.8228e-03 \\ 8.4642e-03 \\ 7.5872e-07 \\ 6.5816e-07 \\ 9.1614e-11 \\ 2.5727e-16 \end{bmatrix} \qquad \lambda(A) = \begin{bmatrix} 1.0000e+00+0.0000e+00i \\ 2.5000e-01+0.0000e+00i \\ 2.5000e-01+0.0000e+00i \\ -1.7684e-04+5.6050e-05i \\ -1.7684e-04-5.6050e-05i \\ -1.4499e-04+0.0000e+00i \\ -3.3473e-05+0.0000e+00i \\ -1.6317e-04+0.0000e+00i \\ 2.3786e-05+1.7410e-04i \\ 2.3786e-05-1.7410e-04i \\ 8.4030e-05+0.0000e+00i \\ 1.8033e-04+4.3683e-05i \\ 1.8033e-04-4.3683e-05i \\ 1.8292e-04+7.0199e-06i \\ 1.8292e-04-7.0199e-06i \\ 6.2500e-02+0.0000e+00i \\ 6.2500e-02+0.0000e+00i \\ 6.2500e-02+0.0000e+00i \end{bmatrix}$$

$$\mathrm{rank}(A) = 17, \qquad \mathrm{rank}\ \mathrm{diag}(\lambda(A)) = 18.$$

Second Example

We consider a matrix of order 18 considered in the paper

N. Guglielmi, M.L. Overton, G.W. Stewart, An Efficient Algorithm for Computing the Generalized Null Space Decomposition, **SIAM J. Matrix Anal. Appl.** 36, 38–54, 2015.

$$\sigma(A) = \begin{bmatrix} 1.9288e+00 \\ 1.3561e+00 \\ 8.7206e-01 \\ 6.8301e-01 \\ 6.3444e-01 \\ 3.1994e-01 \\ 3.1287e-01 \\ 1.6030e-01 \\ 3.1547e-02 \\ 1.7171e-02 \\ 1.4103e-02 \\ 1.1660e-02 \\ 8.8228e-03 \\ 8.4642e-03 \\ 7.5872e-07 \\ 6.5816e-07 \\ 9.1614e-11 \\ 2.5727e-16 \end{bmatrix} \qquad \lambda(A) = \begin{bmatrix} 1.0000e+00+0.0000e+00i \\ 2.5000e-01+0.0000e+00i \\ 2.5000e-01+0.0000e+00i \\ -1.7684e-04+5.6050e-05i \\ -1.7684e-04-5.6050e-05i \\ -1.4499e-04+0.0000e+00i \\ -3.3473e-05+0.0000e+00i \\ -1.6317e-04+0.0000e+00i \\ 2.3786e-05+1.7410e-04i \\ 2.3786e-05-1.7410e-04i \\ 8.4030e-05+0.0000e+00i \\ 1.8033e-04+4.3683e-05i \\ 1.8033e-04-4.3683e-05i \\ 1.8292e-04+7.0199e-06i \\ 1.8292e-04-7.0199e-06i \\ 6.2500e-02+0.0000e+00i \\ 6.2500e-02+0.0000e+00i \\ 6.2500e-02+0.0000e+00i \end{bmatrix}$$

$\mathrm{rank}(A) = 17, \qquad \mathrm{rank}\ \mathrm{diag}(\lambda(A)) = 18.$

$\lambda_n, \mathbf{y}_{\lambda_n}, \mathbf{x}_{\lambda_n}$ smallest eigenvalue of A and associated eigenvectors.

$\mathrm{cond}(\lambda_n) = \frac{1}{|\mathbf{y}_{\lambda_n}^H \mathbf{x}_{\lambda_n}|} = 2.6901e+11.$

This means that $O(\epsilon)$ perturbations in A can induce $\frac{\epsilon}{|\mathbf{y}_{\lambda_n}^H \mathbf{x}_{\lambda_n}|}$ changes in the eigenvalue

(if $\epsilon = 2.2204e-16$, $\frac{\epsilon}{|\mathbf{y}_{\lambda_n}^H \mathbf{x}_{\lambda_n}|} = 5.9732e-05$.)

Second Example

After having applied the first step of the algorithm, i.e., after having computed H by the reverse Hessenberg reduction, we have

$$H(1:2,1) = \begin{bmatrix} 8.2665e-03 \\ -1.1695e-02 \end{bmatrix} \text{ and } \mathrm{diag}(H,-1) = \begin{bmatrix} -1.1695e-02 \\ 3.0480e-04 \\ 2.6277e-07 \\ 4.8788e-04 \\ 1.3198e-05 \\ 3.5209e-03 \\ 8.2828e-03 \\ 9.8768e-04 \\ 1.6310e-05 \\ 1.4189e-06 \\ 8.0967e-05 \\ 3.1571e-02 \\ 1.2242e-02 \\ 6.5106e-02 \\ 1.2294e-01 \\ 1.1377e+00 \\ 4.9063e-01 \end{bmatrix}$$

Second Example

After having applied the first step of the algorithm, i.e., after having computed H by the reverse Hessenberg reduction, we have

$$H(1:2,1) = \begin{bmatrix} 8.2665e-03 \\ -1.1695e-02 \end{bmatrix} \text{ and } \mathrm{diag}(H,-1) = \begin{bmatrix} -1.1695e-02 \\ 3.0480e-04 \\ 2.6277e-07 \\ 4.8788e-04 \\ 1.3198e-05 \\ 3.5209e-03 \\ 8.2828e-03 \\ 9.8768e-04 \\ 1.6310e-05 \\ 1.4189e-06 \\ 8.0967e-05 \\ 3.1571e-02 \\ 1.2242e-02 \\ 6.5106e-02 \\ 1.2294e-01 \\ 1.1377e+00 \\ 4.9063e-01 \end{bmatrix}$$

H is an irreducible Hessenberg matrix

Second Example

Rescue procedure: instead of computing the eigenvector $\mathbf{x}$ associated to the smallest eigenvalue λ of $\hat{H}$, the matrix obtained after one iteration of the QR method with zero shift we :

- Compute the eigenvector $\mathbf{x}$ associated to the smallest eigenvalue λ of H, the matrix obtained after the reverse Hessenberg reduction.

Second Example

Rescue procedure: instead of computing the eigenvector **x** associated to the smallest eigenvalue λ of $\hat{H}$, the matrix obtained after one iteration of the QR method with zero shift we :

- Compute the eigenvector **x** associated to the smallest eigenvalue λ of H, the matrix obtained after the reverse Hessenberg reduction.
- If $\lambda < \texttt{tol}$, $\texttt{tol}$ fixed tolerance, apply the lemma

Computed λ : $\lambda = 3.5721e - 16$;

Second Example

Rescue procedure: instead of computing the eigenvector $\mathbf{x}$ associated to the smallest eigenvalue λ of $\hat{H}$, the matrix obtained after one iteration of the QR method with zero shift we :

- Compute the eigenvector $\mathbf{x}$ associated to the smallest eigenvalue λ of H, the matrix obtained after the reverse Hessenberg reduction.
- If $\lambda <$ `tol`, `tol` fixed tolerance, apply the lemma

$$\text{Computed } \lambda : \; \lambda = 3.5721e-16; \;\; \text{Computed } \mathbf{x} = \begin{bmatrix} -7.4729e-01 \\ -5.2624e-01 \\ -3.6717e-01 \\ 5.0551e-02 \\ 1.2347e-01 \\ -2.2328e-02 \\ -5.7673e-03 \\ -4.9440e-02 \\ 2.2424e-02 \\ 2.1667e-02 \\ 3.3528e-02 \\ 5.6918e-03 \\ 2.8479e-02 \\ 1.4739e-03 \\ -2.7244e-03 \\ 3.2894e-03 \\ -6.1881e-03 \\ 7.7724e-02 \end{bmatrix}$$

Second Example

After having applied the perfect shift srategy on the right eigenvector x of H we obtain $\tilde{H}$, whose first column, the second subdiagonal of $\tilde{H}$ are, respectively,

$$\tilde{H}(:,1) = \begin{bmatrix} -1.6443e-16 \\ -2.4807e-16 \\ 2.7929e-16 \\ 3.2569e-16 \\ 2.1330e-16 \\ 1.0583e-16 \\ 2.6678e-17 \\ -3.8877e-16 \\ 2.7456e-16 \\ -1.5241e-16 \\ 2.0498e-18 \\ -4.0576e-16 \\ 2.5667e-16 \\ -1.3661e-16 \\ -1.4935e-16 \\ -2.7086e-16 \\ -4.1496e-16 \\ 8.8517e-18 \end{bmatrix}, \quad \mathrm{diag}(\tilde{H},-2) = \begin{bmatrix} 2.7929e-16 \\ -3.6899e-16 \\ -4.1543e-16 \\ -5.4518e-16 \\ 4.7972e-17 \\ -2.6513e-15 \\ -5.2372e-16 \\ 7.6815e-17 \\ -3.4518e-19 \\ -1.0449e-15 \\ 6.6930e-16 \\ -6.1971e-16 \\ -1.2311e-16 \\ -1.1904e-15 \\ -1.0128e-16 \\ 7.1288e-18 \end{bmatrix},$$

and $\|\mathrm{tril}(\tilde{H},-2)\|_2 = 9.1426e-15$

Numerical example 2

$\|A - Q_{GOS} B_{GOS} Q_{GOS}^H\|_2$	$\|A - Q_{MV} B_{MV} Q_{MV}^H\|_2$
1.1185e-15	2.5532e-15

$\texttt{tol} = 1.0e-13, \;\; \texttt{Index} = 1, \; s_1 = 1.$

Numerical example 3

The design of smooth surfaces using subdivision algorithms, a common technique used in computer graphics, leads to certain eigenvalue optimization problems.
Computations for a triangular mesh led to the matrix

$$A = \begin{bmatrix} 69/448 & 2101/9632 & 2101/9632 & 2101/9632 & 295/19264 & 1403/28896 & 295/19264 & 1403/28896 & 295/19264 & 1403/28896 \\ 233/896 & 248/896 & 171/896 & 171/896 & 15/896 & 29/896 & 0 & 0 & 0 & 29/896 \\ 233/896 & 171/896 & 248/896 & 171/896 & 0 & 29/896 & 15/896 & 29/896 & 0 & 0 \\ 233/896 & 171/896 & 171/896 & 248/896 & 0 & 0 & 0 & 29/896 & 15/896 & 29/896 \\ 3/32 & 7/16 & 3/32 & 3/32 & 3/32 & 3/32 & 0 & 0 & 0 & 3/32 \\ 9/64 & 39/128 & 39/128 & 3/64 & 3/128 & 9/64 & 3/128 & 1/128 & 0 & 1/128 \\ 3/32 & 3/32 & 7/16 & 3/32 & 0 & 3/32 & 3/32 & 3/32 & 0 & 0 \\ 9/64 & 3/64 & 39/128 & 39/128 & 0 & 1/128 & 3/128 & 9/64 & 3/128 & 1/128 \\ 3/32 & 3/32 & 3/32 & 7/16 & 0 & 0 & 0 & 3/32 & 3/32 & 3/32 \\ 9/64 & 39/128 & 3/64 & 39/128 & 3/128 & 1/128 & 0 & 1/128 & 3/128 & 9/64 \end{bmatrix}$$

Numerical examples 3

$$B_{GOS} = \begin{bmatrix}
0 & 0 & 0 & -3.84e-03 & -1.61e-02 & 1.54e-01 & 1.07e-01 & -3.49e-01 & -3.41e-02 & 4.85e-02 \\
0 & 0 & 0 & -5.22e-03 & -2.19e-02 & -3.12e-01 & 2.36e-01 & -6.45e-02 & 4.91e-02 & 3.23e-02 \\
0 & 0 & 0 & 2.53e-02 & 1.06e-01 & 1.86e-01 & 2.97e-01 & 1.72e-01 & 4.92e-03 & 1.40e-02 \\
0 & 0 & 0 & 0 & -1.13e-01 & -3.30e-01 & -3.37e-01 & -3.45e-01 & 0 & 0 \\
0 & 0 & 0 & 0 & 1.17e-01 & 4.57e-02 & 4.67e-02 & 4.78e-02 & 0 & 0 \\
0 & 0 & 0 & 0 & 3.37e-01 & 4.71e-01 & 2.26e-01 & 2.31e-01 & -3.29e-02 & 0 \\
0 & 0 & 0 & 0 & 3.45e-01 & 2.26e-01 & 4.81e-01 & 2.36e-01 & 1.57e-02 & 2.85e-02 \\
0 & 0 & 0 & 0 & 3.53e-01 & 2.31e-01 & 2.36e-01 & 4.92e-01 & 1.60e-02 & -2.78e-02 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6.25e-02 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6.25e-02
\end{bmatrix}$$

$$B_{MV} = \begin{bmatrix}
0 & 0 & 0 & -4.97e-04 & -2.18e-02 & -2.84e-05 & -3.41e-01 & -1.06e-02 & -2.09e-01 & -5.62e-03 \\
0 & 0 & 0 & 1.75e-02 & -1.62e-02 & -6.60e-02 & 1.51e-01 & 5.42e-02 & -2.54e-01 & 2.59e-01 \\
0 & 0 & 0 & -1.94e-02 & 2.69e-02 & 4.52e-02 & 1.44e-01 & -8.18e-02 & -2.27e-01 & -2.84e-01 \\
0 & 0 & 0 & 0 & 5.57e-02 & 1.42e-01 & -5.65e-05 & -1.77e-01 & -3.84e-03 & -5.48e-01 \\
0 & 0 & 0 & 0 & 6.25e-02 & 1.47e-05 & 3.71e-02 & 3.76e-05 & 1.00e-02 & -1.22e-01 \\
0 & 0 & 0 & 0 & 0 & 6.24e-02 & -1.44e-02 & 9.61e-05 & 2.56e-02 & -3.13e-01 \\
0 & 0 & 0 & 0 & 0 & 7.46e-05 & 2.50e-01 & -3.82e-08 & -1.01e-05 & 1.24e-04 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 6.25e-02 & 2.98e-02 & 3.91e-01 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 6.51e-04 & 2.49e-01 & 4.74e-03 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 6.10e-03 & 9.99e-01
\end{bmatrix}$$

$\|A - Q_{GOS} B_{GOS} Q_{GOS}^H\|_2$	$\|A - Q_{MV} B_{MV} Q_{MV}^H\|_2$
7.59e-16	1.53e-15

$\mathtt{tol} = 1.0e-13, \;\; \mathtt{Index} = 2, \; s_1 = 3, \; s_2 = 1.$

Conclusions

- An algorithm for computing the generalized null–space of a matrix has been presented.

Conclusions

- An algorithm for computing the generalized null–space of a matrix has been presented.
- The algorithm is backward stable, relying only on orthogonal transformations.

K. M. Anstreicher and U. G. Rothblum, *Using Gauss-Jordan elimination to compute the index, generalized nullspaces, and Drazin inverse*, Linear Algebra and its Applications, 85 (1987), pp. 221–239.

T. Beelen and P. Van Dooren, *Computational aspects of the Jordan canonical form*, Oxford Univ. Press, New York, 1990, pp. 57–72.

G. H. Golub and J. H. Wilkinson, *Ill–conditioned eigensystems and the computation of the Jordan canonical form*, SIAM Rev., 36 (1976), pp. 578–619.

N. Guglielmi, M. Overton, and G. Stewart, *An efficient algorithm for computing the generalized null space decomposition*, SIAM J. Matrix Anal. Appl., 36 (2015), pp. 38–54.

R. Horn and C. Johnson, *Matrix analysis. Second Ed.*

V. N. Kublanovskaya, *On solving the complete eigenvalue problem for a degenerate matrix*, USSR Computational Math. and Math. Phys., 6 (1968), pp. 1–14.

P. Lancaster and M. Tismenetsky, *Theory of Matrices*, Academic Press, 1985.

N. Mastronardi and P. Van Dooren, *Computing the Jordan structure of an eigenvalue*, in preparation, (2015).

———, *Creating a nilpotent pencil via deadbeat*, International Journal of Control, (2015).

A. Ruhe, *An algorithm for numerical determination of the structure of a general matrix*, BIT, 10 (1970), pp. 196–216.

H. Shapiro, *The Weyr charactersitic*, American Mathematical Monthly, 106 (1999), pp. 919–929.

D. S. Watkins, *The Matrix Eigenvalue Problem*, SIAM, Philadelphia, 2007.

Z. Zeng, *Sensitivity and computation of a defective eigenvalue*, SIAM. J. Matrix Anal. Appl., 37 (2016), pp. 798–817.

[1] [12] [5] [2] [11] [10] [6] [3] [4] [9] [8] [6] [7] [13]