University of Rostock
Institute of Mathematics
PD Dr. M. Sawall

Rostock, den 21.11.2022

Suggested solutions for self-study & additional practice for the 6. Tutorial

Sample solution for the additional exercise 24:

For the power method the iterations read

$$x_2 = \begin{pmatrix} 0.30151 \\ -0.30151 \\ -0.90453 \end{pmatrix}, \quad x_3 = \begin{pmatrix} 0.09017 \\ -0.09017 \\ 0.99184 \end{pmatrix}, \quad x_4 = \begin{pmatrix} -0.06312 \\ 0.31560 \\ -0.94679 \end{pmatrix}.$$

The Rayleight-quotients are

$$r_1 = -1.6667, \quad r_2 = -3.0, \quad r_3 = -4.1707.$$

The eigenvalue of target is $\lambda_1 = -4.7321$.

For the inverse power method we get

$$x_2 = \begin{pmatrix} -0.70436 \\ -0.61632 \\ -0.35218 \end{pmatrix}, \quad x_2 = \begin{pmatrix} 0.75324 \\ 6.0259 \\ 0.26364 \end{pmatrix}, \quad x_2 = \begin{pmatrix} -0.77350 \\ -5.9010 \\ -0.23125 \end{pmatrix}.$$

The Rayleight-quotients are

$$r_1 = -1.4211, \quad r_2 = -1.29, \quad r_3 = -1.2714,$$

which approximate $\lambda_3 = -1.2679$.

An implementation:

```
A = [-2 1 0;1 -3 1;0 1 -4];

x = [1;1;1]/sqrt(3);
x = x/norm(x);
for i=1:3
    y = A*x;
    r = x'*y;
    x = y/norm(y,2);
    R(i) = r;
    X(:,i) = x;
end
X
R

x = [1;1;1]/sqrt(3);
x = x/norm(x);
for i=1:3
    y = A\x;
    r = 1/(x'*y);
    x = y/norm(y,2);
    R(i) = r;
    X(:,i) = x;
```

```
end
X
R
```

Sample solution for the additional exercise 25:

The iterations read

$$x_2 = \begin{pmatrix} 0.94346 \\ 0.10483 \\ -0.31449 \end{pmatrix}, \quad x_3 = \begin{pmatrix} -0.1957 \\ 0.74365 \\ 0.63928 \end{pmatrix}, \quad x_4 = \begin{pmatrix} 0.70790 \\ -0.45956 \\ -0.53636 \end{pmatrix}$$

and the shifted Rayleight-quotients are

$$r_1 = -0.14286, \quad r_2 = -4.7244, \quad r_3 = -3.1556,$$

which approximate $\lambda_3 = -3$.

An implementation:

```
A = [-2 1 0;1 -3 1;0 1 -4];

mu = -2.5;
x = [1;1;1]/sqrt(3);
x = x/norm(x);
for i=1:3
    y = (A-mu*eye(3,3))\x;
    r = 1/(x'*y)+mu;
    x = y/norm(y,2);
    R(i) = r;
    X(:,i) = x;
end
X
R
```

Sample solution for the additional exercise 26:

For the QR-algorithm

$$A^{(1)} = \begin{pmatrix} -3.0000 & 1.0954 & 0.0000 \\ 1.0954 & -3.0000 & -1.3416 \\ 0 & -1.3416 & -3.0000 \end{pmatrix},$$
$$A^{(2)} = \begin{pmatrix} -3.7059 & 0.9558 & -0.0000 \\ 0.9558 & -3.5214 & 0.9738 \\ 0 & 0.9738 & -1.7727 \end{pmatrix},$$
$$A^{(3)} = \begin{pmatrix} -4.1566 & 0.8284 & 0.0000 \\ 0.8284 & -3.4880 & -0.4162 \\ 0 & -0.4162 & -1.3553 \end{pmatrix}.$$

An implementation:

```
A = [-2 1 0;1 -3 1;0 1 -4];

Qall = eye(3,3);
for ell=1:3
    [q,r] = qr(A);
    A = r*q
```

```
    Qall = Qall*q;
end
```

For Jacobi's iteration we get

$$B^{(1)} = \begin{pmatrix} -1.3820 & 0.0000 & 0.5257 \\ 0.0000 & -3.6180 & 0.8507 \\ 0.5257 & 0.8507 & -4.0000 \end{pmatrix},$$

$$B^{(2)} = \begin{pmatrix} -1.38200.32850.4105 & & \\ 0.3285 & -2.9372 & 0.0000 \\ 0.4105 & 0.0000 & -4.6808 \end{pmatrix},$$

$$B^{(3)} = \begin{pmatrix} -1.33170.3261 - 0.0000 & & \\ 0.3261 & -2.9372 & -0.0400 \\ -0.0000 & -0.0400 & -4.7311 \end{pmatrix}.$$

An implementation:

```
A = [-2 1 0;1 -3 1;0 1 -4];

Jall = eye(3,3);
for ell=1:3
    [i,j] = iden(A);
    rho = (A(j,j)-A(i,i))/2/A(i,j);
    if rho>=0
        t = 1/(rho+sqrt(1+rho^2));
    else
        t = 1/(rho-sqrt(1+rho^2));
    end
    c = 1/sqrt(1+t^2); s = t*c;
    J = eye(3,3); J(i,j) = s; J(j,i) = -s; J(i,i) = c; J(j,j) = c;
    A = J'*A*J
    Jall = Jall*J;
end

[val,id] = sort(diag(A));
vec = Jall(:,id);

function [id,jd] = iden(A)
B = A-diag(diag(A));
[ma,id] = max(abs(B));
[~,jd] = max(ma);
id = id(jd);
end
```

For both methods the diagonal elements of $A^{(3)}$ resp. $B^{(3)}$ are good approximations to

$$\lambda_1 = -4.7321, \quad \lambda_2 = -3.0, \quad \lambda_3 = -1.2679.$$

But for both methods the columns of Qall and Jall are poor approximations to the eigenvectors of A as 3 iterations are too less.

Sample solution for the additional exercise 27:

An implementation for (a):

```
A = zeros(6,6);
for i=1:6
```

```
    for j=1:6
        if i==j
            A(i,j) = 8-i;
        elseif abs(i-j)==1
            A(i,j) = -1;
        end
    end
end

maxiter = 1000;
Qall = eye(6,6);
while ell< maxiter && of(A)>1.E-7
    [q,r] = qr(A);
    A = r*q;
    Qall = Qall*q;
    ell = ell+1;
end
[ell of(A)]
Qall

function e = of(A)
B = A-diag(diag(A));
e = (B(:))'*B(:);
end
```

And an implementation for (b):

```
n = 30;
A = (n+1)^2*(-2*diag(ones(n,1))+diag(ones(n-1,1),1)+diag(ones(n-1,1),-1)); A0 = A;
maxiter = 10000;
ell = 1;
Jall = eye(n,n);
oa = of(A);

while oa(end)>1.E-4 && ell<maxiter
    [i,j] = iden(A);
    rho = (A(j,j)-A(i,i))/2/A(i,j);
    if rho>=0
        t = 1/(rho+sqrt(1+rho^2));
    else
        t = 1/(rho-sqrt(1+rho^2));
    end
    c = 1/sqrt(1+t^2); s =t*c;
    J = eye(n,n); J(i,j) = s; J(j,i) = -s; J(i,i) = c; J(j,j) = c;
    A = J'*A*J;
    Jall = Jall*J;
    oa(end+1) = of(A);
    app(:,ell) = sort(diag(A));
    ell = ell+1;
end

[val,id] = sort(diag(A));
vec = Jall(:,id);

plot((1:n)/(n+1),vec(:,[1 2 29 30]))

function e = of(A)
B = A-diag(diag(A));
e = (B(:))'*B(:);
```

```
end

function [id,jd] = iden(A)
B = A-diag(diag(A));
[ma,id] = max(abs(B));
[~,jd] = max(ma);
id = id(jd);
end
```