

Math 1080: Spring 2011
Homework #8 (due April 1)

Problem 1:

Find the diagonalization $\mathbf{A} = \mathbf{X}\mathbf{\Lambda}\mathbf{X}^{-1}$ of the following matrix:

$$\mathbf{A} = \begin{bmatrix} -1 & 0 & -4 & 0 \\ 0 & 2 & -1 & 0 \\ 0 & 0 & 3 & 0 \\ 0 & 3 & 3 & -1 \end{bmatrix}$$

SOLUTION:

In order to calculate the decomposition we must first determine the eigenvalues and eigenvectors of \mathbf{A} .

The characteristic polynomial is

$$\begin{aligned} \det(\mathbf{A} - \lambda\mathbf{I}) &= \begin{vmatrix} -1-\lambda & 0 & -4 & 0 \\ 0 & 2-\lambda & -1 & 0 \\ 0 & 0 & 3-\lambda & 0 \\ 0 & 3 & 3 & -1-\lambda \end{vmatrix} = (-1-\lambda) \begin{vmatrix} 2-\lambda & -1 & 0 \\ 0 & 3-\lambda & 0 \\ 3 & 3 & -1-\lambda \end{vmatrix} \\ &= (-1-\lambda)(3-\lambda) \begin{vmatrix} 2-\lambda & 0 \\ 3 & -1-\lambda \end{vmatrix} = (-1-\lambda)^2 (3-\lambda)(2-\lambda) \end{aligned}$$

which has the roots 3, 2, and -1. Thus the eigenvalues are $\lambda_1 = 3$, $\lambda_2 = 2$, $\lambda_3 = -1$.

For $\lambda_1 = 3$ the eigenvector is obtained by solving

$$(\mathbf{A} - \lambda_1\mathbf{I})\mathbf{x}_1 = \begin{bmatrix} -4 & 0 & -4 & 0 \\ 0 & -1 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 3 & 3 & -4 \end{bmatrix} \mathbf{x}_1 = \mathbf{0} \quad \text{which yields, for example, } \mathbf{x}_1 = \begin{bmatrix} 1 \\ 1 \\ -1 \\ 0 \end{bmatrix}$$

For $\lambda_2 = 2$ the eigenvector is obtained by solving

$$(\mathbf{A} - \lambda_2\mathbf{I})\mathbf{x}_2 = \begin{bmatrix} -3 & 0 & -4 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 3 & 3 & -3 \end{bmatrix} \mathbf{x}_2 = \mathbf{0} \quad \text{which yields, for example, } \mathbf{x}_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}$$

The eigenvalue $\lambda_3 = -1$ is a double eigenvalue and we expect two linearly independent eigenvectors obtained by solving

$$(\mathbf{A} - \lambda_3\mathbf{I})\mathbf{x}_3 = \begin{bmatrix} 0 & 0 & -4 & 0 \\ 0 & 3 & -1 & 0 \\ 0 & 0 & 4 & 0 \\ 0 & 3 & 3 & 0 \end{bmatrix} \mathbf{x}_3 = \mathbf{0} \quad \text{which yields } \mathbf{x}_3 = \begin{bmatrix} s \\ 0 \\ 0 \\ t \end{bmatrix}$$

with s and t free parameters. We can pick, for example, $\mathbf{x}_3 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ and $\mathbf{x}_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$.

The columns of the matrix \mathbf{X} are the eigenvectors of \mathbf{A} and the diagonal elements of $\mathbf{\Lambda}$ are the corresponding eigenvalues. Thus

$$\mathbf{X} = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}, \quad \mathbf{\Lambda} = \begin{bmatrix} 3 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

Problem 2:

Find the Schurr decomposition $\mathbf{A} = \mathbf{Q}\mathbf{T}\mathbf{Q}^T$ for the following symmetric matrix \mathbf{A} :

$$\mathbf{A} = \begin{bmatrix} 4 & -2 & 1 \\ -2 & 4 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$

SOLUTION:

Schurr decomposition of a symmetric matrix is identical to diagonalization.

The characteristic polynomial is

$$\begin{aligned} \det(\mathbf{A} - \lambda\mathbf{I}) &= \begin{vmatrix} 4-\lambda & -2 & 1 \\ -2 & 4-\lambda & 1 \\ 1 & 1 & 1-\lambda \end{vmatrix} = (4-\lambda) \begin{vmatrix} 4-\lambda & 1 \\ 1 & 1-\lambda \end{vmatrix} - (-2) \begin{vmatrix} -2 & 1 \\ 1 & 1-\lambda \end{vmatrix} + \begin{vmatrix} -2 & 4-\lambda \\ 1 & 1 \end{vmatrix} \\ &= (4-\lambda)((4-\lambda)(1-\lambda)-1) + 2(-2(1-\lambda)-1) + (-2-(4-\lambda)) \quad \text{which has the} \\ &= -\lambda(\lambda^2 - 9\lambda + 18) \\ &= -\lambda(\lambda-3)(\lambda-6) \end{aligned}$$

roots 6, 3, and 0. Thus the eigenvalues are $\lambda_1 = 6$, $\lambda_2 = 3$, $\lambda_3 = 0$.

For $\lambda_1 = 6$ the eigenvector is obtained by solving

$$(\mathbf{A} - \lambda_1\mathbf{I})\mathbf{x}_1 = \begin{bmatrix} -2 & -2 & 1 \\ -2 & -2 & 1 \\ 1 & 1 & -5 \end{bmatrix} \mathbf{x}_1 = \mathbf{0} \quad \text{which yields, for example, } \mathbf{x}_1 = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}$$

For $\lambda_2 = 3$ the eigenvector is obtained by solving

$$(\mathbf{A} - \lambda_2\mathbf{I})\mathbf{x}_2 = \begin{bmatrix} 1 & -2 & 1 \\ -2 & 1 & 1 \\ 1 & 1 & -2 \end{bmatrix} \mathbf{x}_2 = \mathbf{0} \quad \text{which yields, for example, } \mathbf{x}_2 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

For $\lambda_3 = 0$ the eigenvector is obtained by solving

$$(\mathbf{A} - \lambda_3 \mathbf{I}) \mathbf{x}_3 = \begin{bmatrix} 4 & -2 & 1 \\ -2 & 4 & 1 \\ 1 & 1 & 1 \end{bmatrix} \mathbf{x}_3 = \mathbf{0} \quad \text{which yields, for example, } \mathbf{x}_3 = \begin{bmatrix} 1 \\ 1 \\ -2 \end{bmatrix}.$$

The columns $\mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3$ of the matrix \mathbf{Q} are the normalized eigenvectors, i.e.,

$$\mathbf{q}_i = \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|}$$

The result is

$$\mathbf{Q} = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{6}} \\ 0 & \frac{1}{\sqrt{3}} & -\frac{2}{\sqrt{6}} \end{bmatrix}, \quad \mathbf{\Lambda} = \begin{bmatrix} 6 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Computer Assignment 5:

- Write a MATLAB function `[lambda,v]=poweriter(A,v0,eps)` that computes one eigenvalue of a square, symmetric matrix \mathbf{A} using *power iteration* and Rayleigh quotient with the initial guess $\mathbf{v0}$. The program should terminate when the difference between two consecutive estimates, $|\lambda^{(k+1)} - \lambda^{(k)}|$, is smaller than the tolerance `eps`. The program should also print out $\lambda^{(k)}$ and $\mathbf{v}^{(k)}$ for each step of the iteration.
- Write a MATLAB function `[lambda,v]=inverseiter(A,v0,mu,eps)` that computes one eigenvalue of a square, symmetric matrix \mathbf{A} using *inverse iteration* and Rayleigh quotient with the initial guess $\mathbf{v0}$ and `mu`. The program should terminate when $|\lambda^{(k+1)} - \lambda^{(k)}|$ is smaller than `eps` and it should print out $\lambda^{(k)}$ and $\mathbf{v}^{(k)}$ for each step of the iteration.
- Calculate at the leading eigenvalue of the following matrix using both algorithms starting with two different initial guesses $\mathbf{v0}$ using tolerance 10^{-6} . With inverse iteration find at least two different eigenvalues using different `mu`. Note how many iterations were needed in each case.

$$\mathbf{A} = \begin{bmatrix} 2 & 6 & 4 & -4 & -5 & -10 \\ 6 & 12 & -2 & 9 & 5 & 9 \\ 4 & -2 & 0 & -1 & -3 & 14 \\ -4 & 9 & -1 & 14 & -6 & 8 \\ -5 & 5 & -3 & -6 & 2 & 8 \\ -10 & 9 & 14 & 8 & 8 & 8 \end{bmatrix}$$

PROGRAMS:

```
function [r,v] = poweriter(A,v0,eps)
v = v0/norm(v0);
r = 0;
rold = 100;
k = 0;
disp('Iter    lambda        v');
while abs(r-rold) > eps
    rold = r;
    w = A*v;
    k = k + 1;
    v = w/norm(w);
    r = v'*A*v;
    disp([sprintf('%d\t %6.4f\t ',k,r),sprintf('%6.4f  ',v')])
end
```

```
function [r,v] = inverseiter(A,v0,mu,eps)
n = size(A);
v = v0/norm(v0);
r = 0;
rold = 100;
k = 0;
disp('Iter    lambda        v');
while abs(r-rold) > eps
    rold = r;
    w = (A-mu*eye(n))\v;
    k = k + 1;
    v = w/norm(w);
    r = v'*A*v;
    disp([sprintf('%d\t %6.4f\t ',k,r),sprintf('%6.4f  ',v')])
end
```

OUTPUT

```
>> A = [2 6 4 -4 -5 -10; 6 12 -2 9 5 9; 4 -2 0 -1 -3 14; -4 9 -1 14 -6 8; -5 5 -3 -6 2 8;
-10 9 14 8 8 8]
```

```
>> [r,v] = poweriter(A,[0 0 0 0 0 1]','1e-6);
```

Iter	lambda	v					
1	17.3568	-0.4192	0.3773	0.5869	0.3354	0.3354	0.3354
2	24.7762	-0.0943	0.3103	0.0335	0.3575	0.1293	0.8656
3	28.6235	-0.3003	0.4962	0.3491	0.4809	0.2342	0.5056
4	30.2512	-0.1427	0.4402	0.1206	0.4788	0.1487	0.7211
5	30.8863	-0.2264	0.5065	0.2486	0.5150	0.1850	0.5753
6	31.1272	-0.1654	0.4796	0.1624	0.5103	0.1539	0.6574
7	31.2178	-0.1981	0.5031	0.2116	0.5216	0.1678	0.6023
8	31.2519	-0.1753	0.4919	0.1794	0.5189	0.1566	0.6335
9	31.2646	-0.1879	0.5004	0.1981	0.5226	0.1620	0.6129
10	31.2694	-0.1794	0.4960	0.1861	0.5214	0.1579	0.6248
11	31.2712	-0.1842	0.4991	0.1932	0.5227	0.1600	0.6171
12	31.2719	-0.1811	0.4974	0.1887	0.5222	0.1585	0.6216
13	31.2722	-0.1829	0.4986	0.1914	0.5227	0.1593	0.6187
14	31.2723	-0.1817	0.4979	0.1897	0.5224	0.1588	0.6204
15	31.2723	-0.1824	0.4984	0.1907	0.5226	0.1591	0.6193
16	31.2723	-0.1820	0.4981	0.1901	0.5225	0.1589	0.6200
17	31.2723	-0.1822	0.4983	0.1905	0.5226	0.1590	0.6196
18	31.2723	-0.1821	0.4982	0.1902	0.5226	0.1589	0.6198
19	31.2723	-0.1822	0.4982	0.1904	0.5226	0.1590	0.6196

```
>> [r,v] = poweriter(A,[1 0 0 0 0 0]',1e-6);
```

Iter	lambda	v					
1	3.0964	0.1425	0.4275	0.2850	-0.2850	-0.3562	-0.7125
2	10.3012	0.7374	-0.2808	-0.4679	-0.2396	-0.2171	-0.2321
3	19.9370	0.0932	-0.1363	0.0471	-0.3640	-0.1853	-0.8965
4	26.3244	0.3858	-0.4755	-0.3886	-0.4529	-0.2352	-0.4622
5	29.3149	0.1320	-0.3995	-0.0936	-0.4671	-0.1525	-0.7568
6	30.5261	0.2494	-0.5056	-0.2705	-0.5094	-0.1924	-0.5590
7	30.9912	0.1598	-0.4695	-0.1520	-0.5055	-0.1531	-0.6723
8	31.1667	0.2052	-0.5041	-0.2201	-0.5206	-0.1712	-0.5958
9	31.2327	0.1729	-0.4890	-0.1753	-0.5173	-0.1559	-0.6390
10	31.2574	0.1903	-0.5011	-0.2013	-0.5225	-0.1632	-0.6103
11	31.2667	0.1784	-0.4951	-0.1845	-0.5209	-0.1576	-0.6268
12	31.2702	0.1851	-0.4995	-0.1944	-0.5227	-0.1605	-0.6161
13	31.2715	0.1807	-0.4971	-0.1881	-0.5220	-0.1584	-0.6223
14	31.2720	0.1832	-0.4987	-0.1918	-0.5227	-0.1595	-0.6183
15	31.2722	0.1816	-0.4978	-0.1895	-0.5224	-0.1587	-0.6207
16	31.2723	0.1825	-0.4984	-0.1909	-0.5226	-0.1592	-0.6192
17	31.2723	0.1819	-0.4981	-0.1900	-0.5225	-0.1589	-0.6201
18	31.2723	0.1823	-0.4983	-0.1905	-0.5226	-0.1590	-0.6195
19	31.2723	0.1821	-0.4982	-0.1902	-0.5226	-0.1589	-0.6198
20	31.2723	0.1822	-0.4982	-0.1904	-0.5226	-0.1590	-0.6196
21	31.2723	0.1821	-0.4982	-0.1903	-0.5226	-0.1589	-0.6198

```
>> [r,v] = inverseiter(A,[1 0 0 0 0 0]',30,1e-6);
```

Iter	lambda	v					
1	29.3499	-0.0975	-0.5850	-0.2062	-0.5297	-0.1112	-0.5593
2	31.2627	0.2001	-0.4866	-0.1898	-0.5187	-0.1645	-0.6253
3	31.2723	0.1809	-0.4992	-0.1902	-0.5230	-0.1584	-0.6191
4	31.2723	0.1822	-0.4981	-0.1903	-0.5225	-0.1590	-0.6198
5	31.2723	0.1821	-0.4982	-0.1903	-0.5226	-0.1590	-0.6197

```
>> [r,v] = inverseiter(A,[1 0 0 0 0 0]',20,1e-6);
```

Iter	lambda	v					
1	14.0122	-0.7313	-0.5329	0.0179	-0.3004	0.2356	0.1874
2	15.8398	0.5507	0.4531	-0.2060	0.3586	-0.3640	-0.4335
3	16.1050	-0.4449	-0.4436	0.2547	-0.4709	0.3802	0.4173
4	16.1497	0.4126	0.4031	-0.2792	0.4898	-0.3959	-0.4388
5	16.1577	-0.3963	-0.3935	0.2835	-0.5079	0.4007	0.4349
6	16.1593	0.3908	0.3856	-0.2857	0.5124	-0.4043	-0.4370
7	16.1596	-0.3882	-0.3831	0.2859	-0.5155	0.4058	0.4364
8	16.1596	0.3871	0.3816	-0.2860	0.5165	-0.4066	-0.4365
9	16.1596	-0.3867	-0.3810	0.2860	-0.5171	0.4070	0.4364
10	16.1596	0.3865	0.3807	-0.2860	0.5173	-0.4072	-0.4365
11	16.1596	-0.3864	-0.3806	0.2860	-0.5174	0.4073	0.4364

```
>> [r,v] = inverseiter(A,[1 0 0 0 0 0]',-10,1e-6);
```

Iter	lambda	v					
1	-10.3032	-0.4653	0.4071	-0.0850	-0.4239	-0.6513	0.0820
2	-10.3079	0.4729	-0.3915	0.1107	0.4215	0.6502	-0.1025
3	-10.3079	-0.4722	0.3914	-0.1110	-0.4217	-0.6505	0.1031
4	-10.3079	0.4722	-0.3914	0.1111	0.4217	0.6505	-0.1031