# MA2507 Computing Mathematics Laboratory: Week 2

1. Eigenvalue problem. For a square matrix $A$, the eigenvalues satisfy $\det(\lambda I - A) = 0$. For each eigenvalue $\lambda_j$, you can find an eigenvector $\mathbf{v}_j$ such that $A\mathbf{v}_j = \lambda_j \mathbf{v}_j$. There are many eigenvalue problems for linear operators (often given as differential operators). For a string (e.g. of a violin), there is an eigenvalue problem given by an ordinary differential equation, for which the eigenvalues are related to the vibration frequencies (the vibrations give rise to the sound). In quantum mechanics, the energy of an electron in an atom cannot take arbitrary values. The allowed energey levels are the eigenvalues of a differential operator (given in the so-called Schrödinger equation). Eigenvalue problems of linear operators can be reduced to matrix eigenvalue problems.

```
>> A=[1 2 3; 4 5 6; 3 1 2]
A =
     1     2     3
     4     5     6
     3     1     2
>> eig(A)    % find the eigenvalues
ans =
    8.5772
   -1.3528
    0.7756
>> [V,D]=eig(A) % find eigenvalues and eigenvectors
V =
   -0.3514   -0.6970    0.0526
   -0.8884   -0.2090   -0.8327
   -0.2954    0.6860    0.5512
D =
    8.5772         0         0
         0   -1.3528         0
         0         0    0.7756
>> v1=V(:,1)   % first column of V is the first eigenvector
v1 =
   -0.3514
   -0.8884
   -0.2954
>> ev(1)=D(1,1) % (1,1) entry of D is the first eigenvalue
ev =
    8.5772
>> A*v1         % Matrix multiplies an eigenvector
ans =
   -3.0143
   -7.6200
   -2.5334
>> ev(1)*v1     % eigenvalue multiplies the eigenvector
ans =
   -3.0143
   -7.6200
   -2.5334
>> A*V          % AV=VD, not DV
ans =
   -3.0143    0.9429    0.0408
   -7.6200    0.2828   -0.6459
   -2.5334   -0.9280    0.4275
>> V*D
ans =
   -3.0143    0.9429    0.0408
   -7.6200    0.2828   -0.6459
   -2.5334   -0.9280    0.4275
>> V*D*inv(V)  % eigenvalue decomposition A=V*D*(V inverse)
ans =
    1.0000    2.0000    3.0000
    4.0000    5.0000    6.0000
    3.0000    1.0000    2.0000
>> V*D/V      % inv of V replaced by right division
ans =
    1.0000    2.0000    3.0000
    4.0000    5.0000    6.0000
    3.0000    1.0000    2.0000
>> s=diag(D)  % diagonals of D are the eigenvalues
s =
    8.5772
   -1.3528
    0.7756
>> ss=sort(s) % order real eigenvalues from small to large
ss =
   -1.3528
    0.7756
    8.5772
>> [ss,II]=sort(s)  %II gives the index for the ordering
ss =
   -1.3528
    0.7756
    8.5772
II =
     2
     3
     1
>> VV=V(:,II)   % re-order the eigenvectors
VV =
   -0.6970    0.0526   -0.3514
   -0.2090   -0.8327   -0.8884
    0.6860    0.5512   -0.2954
>> A=[4 -1 1; 1 2 0; -1 1 1]     % A is not diagonalizable
```

```
A =
     4    -1     1
     1     2     0
    -1     1     1
>> [V,D]=eig(A)
V =
  -0.7071 + 0.0000i   0.0000 - 0.0000i   0.0000 + 0.0000i
  -0.7071 + 0.0000i  -0.7071 - 0.0000i  -0.7071 + 0.0000i
   0.0000 + 0.0000i  -0.7071 + 0.0000i  -0.7071 + 0.0000i
D =
   3.0000 + 0.0000i   0.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   2.0000 + 0.0000i   0.0000 + 0.0000i
   0.0000 + 0.0000i   0.0000 + 0.0000i   2.0000 - 0.0000i
>> real(V)      % show the real part only
ans =
   -0.7071    0.0000    0.0000
   -0.7071   -0.7071   -0.7071
    0.0000   -0.7071   -0.7071
>> real(D)      % should be all real
ans =
    3.0000         0         0
         0    2.0000         0
         0         0    2.0000
>> det(V)      % should be exactly zero
ans =
   5.5511e-17 - 1.8795e-08i
```

2. Simple graphics. MATLAB has excellent capability for graphics. Here, we use command `plot` to draw a Bessel function. The result is shown in the next page.

```
>> x=0:0.05:40;
>> y=besselj(0,x);
>> plot(x,y)
>> xlabel('x')     % add a label for the x-axis.
>> title('Bessel function J_0(x)')   % add a title to the figure.
```

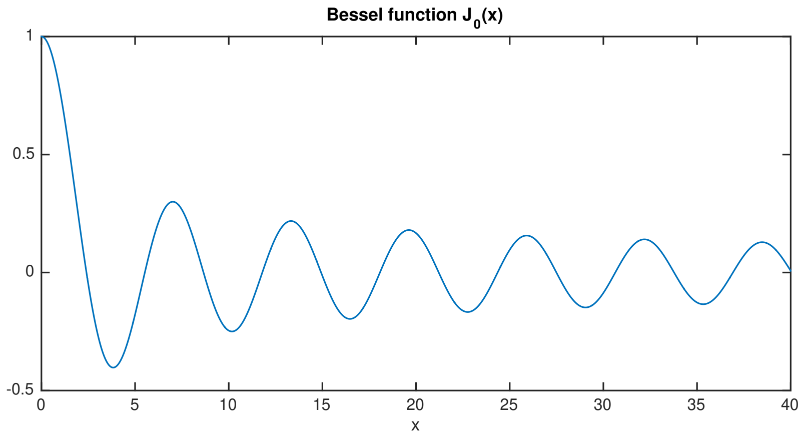
There are two kinds of Bessel functions $J_n(x)$ and $Y_n(x)$, where $n$ is an index called the order. In the graphics window, you can save the figure in a proper format. The MATLAB figure format is useful if you want to load the figure and modify the figure later. For easy communications, you may use JPEG image format. I saved the figure in EPS file format to produce this document using LaTeX.

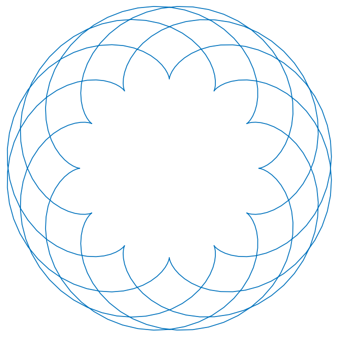
For two constants $a$ and $b$, the curve given by

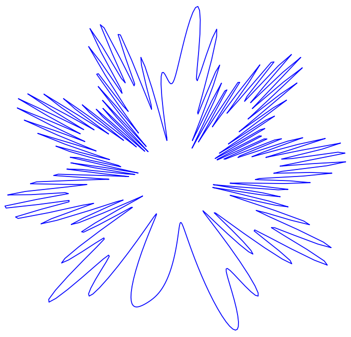
$$x = (a+b)\cos(t) - b\cos[(a/b+1)t], \quad y = (a+b)\sin(t) - b\sin[(a/b+1)t]$$

is quite interesting. The script

```
a=12;
```







```
b=5;
t=linspace(0,10*pi,500);
x=(a+b)*cos(t)-b*cos((a/b+1)*t);
y=(a+b)*sin(t)-b*sin((a/b+1)*t);
plot(x,y)
axis equal
axis off
```

draws the left-lower curve above for $a = 12$, $b = 5$ and $0 \le t \le 10\pi$, with $x$ and $y$ using the same scaling, and the axes removed. For a complex vector, the `plot` command use the real part as the horizontal axis and the imaginary part as the vertical axis. The following script

```
x=linspace(-1,1,500);
f=(3+sin(10*pi*x)+sin(61*exp(.8*sin(pi*x)+.7))).*exp(1i*pi*x);
plot(f,'b')
axis off
axis equal
```

produces the right-lower curve above.

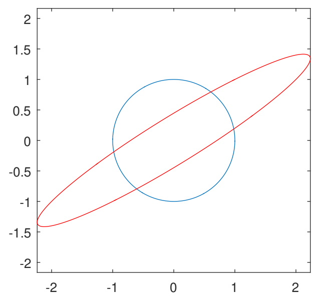
3. Matrix norm and SVD. In MA2503, we studied a little bit of singular value decomposition (SVD). That is, $A = USV^*$, where $U$ and $V$ are unitary matrices (inverse is conjugate transpose) and $S$ is the diagonal matrix for the singular values. The largest singular value of a matrix is also the matrix norm which is defined as $||A|| = \max||Ax||/||x||$. Based on SVD, we claimed that in general a $2 \times 2$ matrix transforms the unit circle to an ellipse. Here is a script file that tests these things.

```
% Matrix multiplication maps a circle to an ellipse.
% This script file is saved as cirell.m
t=linspace(0,2*pi,300);   % 300 points for angle from 0 to 2pi
x = [cos(t); sin(t)];     % the unit circle
A=[1,2;1,1]               % a 2x2 matrix
y = A*x;
plot(x(1,:),x(2,:),y(1,:),y(2,:),'r') % draw two curves
axis equal                % use same scale for two axes
norm(A)                   % matrix norm
svd(A)                    % singular values only
[U,S,V]=svd(A)            % SVD of matrix A
```

If we run th escript in MATLAB, we get

```
>> A=[1,2;1,1]                % a 2x2 matrix
A =
     1     2
     1     1
>> norm(A)
ans =
    2.6180
>> svd(A)
ans =
    2.6180
    0.3820
>> [U,S,V]=svd(A)
U =
   -0.8507   -0.5257
   -0.5257    0.8507
S =
    2.6180         0
         0    0.3820
V =
   -0.5257    0.8507
   -0.8507   -0.5257
```

and the following figure



2
1.5
1
0.5
0
-0.5
-1
-1.5
-2
-2
-1
0
1
2