Solving Systems of Homogeneous Linear Equations (Algebraic and Differential)

$$A\,x = y = 0$$

ill-conditioned problems
Eigenvalues and Eigenvectors

Selis Önel, PhD

“ill-conditioned” Problems

Some linear problems are solvable but solutions become inaccurate due to:

- Approximate empirical data
- Floating point numbers approximation of real numbers
- Small rounding errors
- Small changes in coefficients

Small change in data → large change in solution

“ill-conditioned” Problems

Ex:

Small changes in coefficients → large changes in solution

$$\begin{pmatrix} 0.12065 & 0.98775 \\ 0.12032 & 0.98755 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2.01045 \\ 2.00555 \end{pmatrix} \rightarrow \quad x_1 = 14.7403, \quad x_2 = 0.23942$$

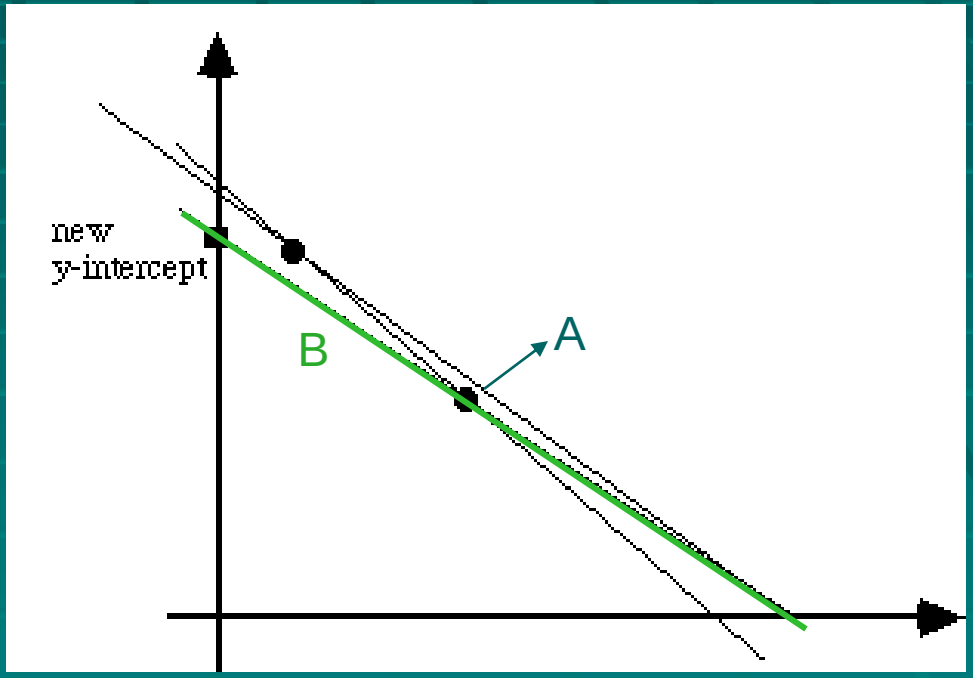
$$\begin{pmatrix} 0.12065 & 0.98775 \\ 0.12032 & 0.98755 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 2.01145 \\ 2.00555 \end{pmatrix} \rightarrow \quad x_1 = 17.9756, \quad x_2 = -0.15928$$

“ill-conditioned” Problems

Ex: Two lines (functions) are nearly parallel

If the intercepts with the vertical axis vary just a little, then the intersection will vary a lot

Hence, the solution of the corresponding 2D system will vary a lot



Ref: http://www2.krellinst.org/UCES/archive/classes/CNA/dir1.7/uces1.7.html

“ill-conditioned” Problems

Ex: Function approximation

To determine the coefficients in an nth order polynomial:

- Set the polynomial at distinct points equal to the function evaluated at these points

Ex: Set a quadratic polynomial near points x_1, x_2 and x_3 ,
then find the three coefficients of

$$P(x) = a_0 + a_1 x + a_2 x^2$$

Solve the system associated with $P(x_i) = f(x_i)$ for i = 1,2,3.

Then, the matrix form is

$$\begin{bmatrix} 1 & x_1 & x_1^2 \\ 1 & x_2 & x_2^2 \\ 1 & x_3 & x_3^2 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \\ a_2 \end{bmatrix} = \begin{bmatrix} f(x_1) \\ f(x_2) \\ f(x_3) \end{bmatrix}.$$

→ If two of the interpolation points are close, then the computed solution may have significant error

“ill-conditioned” Problems

How can we reduce the errors?

Using Gauss elimination to solve the problem

1. Increase the precision of the floating point numbers
2. Avoid division by small numbers

   Use *row pivoting* in the forward sweep of the Gaussian elimination algorithm

Check the “condition” of the coefficient matrix

Ex: ill-conditioned system

Use Gaussian elimination algorithm to solve the following linear system

$$\begin{bmatrix} 10^{-d} & 2 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 4 \\ 3 \end{bmatrix}.$$

1. Use real numbers: Add (-10^d)(row 1) to (row 2) to get

$$\begin{bmatrix} 10^{-d} & 2 \\ 0 & 1-2(10^d) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 4 \\ 3-4(10^d) \end{bmatrix}.$$

for d=4, solve for x_2 and x_1: $x_2 = 3.9997/1.9999$

$x_1 = (4 - 2(3.9997/1.9999))10^4$
$=2(2(1.9999) - 3.9997))/1.9999$
104
$= 2/1.9999$

2. Use floating point numbers with 3 digits

$$\begin{bmatrix} .1 \cdot 10^{-3} & .2 \cdot 10^{1} \\ 0 & fl(1-2(10^4)) \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} .4 \cdot 10^{1} \\ fl(3-4(10^4)) \end{bmatrix}$$

$$fl(1-2(10^4)) = fl((-.2+.0001)10^5) = -.2 \cdot 10^5$$

$$fl(3-4(10^4)) = fl((-.4+.0003)10^5) = -4 \cdot 10^5.$$

3. Interchange rows of the system to avoid division by a small number

Ref: http://www2.krellinst.org/UCES/archive/classes/CNA/dir1.7/uces1.7.html

ill-conditioned system

If computing precision is high, ill-conditioned problems can be solved accurately

Check an ill-conditioned problem, if

- $\det(A)\det((A^{-1}))$ deviates from 1
- $(A^{-1})^{-1}$ is different from A
- AA^{-1} deviates from the identity matrix
- $A^{-1}(A^{-1})^{-1})$ deviates from the identity matrix more significantly than AA^{-1}

Double-precision solves the problems in mildly ill-conditioned problems, so above problems may disappear with double-precision

Ex: Hilbert matrices

Very difficult to invert numerically, very ill-conditioned !

Defined by $A = [a_{i,i}]$ where $a_{i,i} = 1/(i+j-1)$, $i,j = 1,2,\ldots n$

4x4 Hilbert Matrix

$$A = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} & \frac{1}{6} \\ \frac{1}{4} & \frac{1}{5} & \frac{1}{6} & \frac{1}{7} \end{pmatrix}$$

```
% Hilbert.m Compute the condition
    number  and det(A)det(inv(A)) for
    the 5x5 and 14x14
% Hilbert matrices
clear
for n=5:14
   for i=1:n,
      for j=1:n,
         a(i,j)=1/(i+j-1);
      end
   end
   c=cond(a);
   d=det(a)*det(a^(-1));
   fprintf('n=%3.0f   cond(a)=%e
    det*det=%e  \n', n,c,d)
end
```

Ex: Hilbert matrices

n= 5 cond(a)=4.766073e+005 det*det=1.000000e+000
n= 6 cond(a)=1.495106e+007 det*det=1.000000e+000
n= 7 cond(a)=4.753674e+008 det*det=1.000000e+000
n= 8 cond(a)=1.525758e+010 det*det=1.000000e+000
n= 9 cond(a)=4.931538e+011 det*det=1.000000e+000
n= 10 cond(a)=1.602501e+013 det*det=1.000013e+000
n= 11 cond(a)=5.223947e+014 det*det=1.000118e+000
Warning: Matrix is close to singular or badly scaled.
Results may be inaccurate. RCOND = 2.570030e-017.
> In Hilbert at 11 n= 12 cond(a)=1.632619e+016 det*det=1.019081e+000
Warning: Matrix is close to singular or badly scaled.
Results may be inaccurate. RCOND = 1.342113e-018.
> In Hilbert at 11 n= 13 cond(a)=1.316877e+018 det*det=1.211551e+000
Warning: Matrix is close to singular or badly scaled.
Results may be inaccurate. RCOND = 5.393028e-019.
> In Hilbert at 11 n= 14 cond(a)=2.376739e+017 det*det=2.039773e-001

How to find if a matrix is ill-conditioned

Condition number of matrix A

An ill-conditioned problem generates large errors in the computed solution

Let the exact solution be x where Ax = d

Let X be the computed solution where

AX - d = -r → nonzero residual

Relative error is given by the ratio of the "size" of x - X and the "size" of x

In ill-conditioned problem this ratio is large!

How to find if a matrix is ill-conditioned

Condition number of matrix A

In order to be more precise about the "size" we define the max norm of a vector, and it is analogous to the absolute value of a single real number

Cond(A) = $||A||\ ||A^{-1}||$, where ||A|| = Norm of matrix A

Euclidean norm

$$\|A\| = \left(\sum_{i,j} |A_{i,j}|^2 \right)^{1/2}$$

Euclidean Norm is the Length of a vector x : $\|x\| = \sqrt{x_1^2 + x_2^2 + \ldots}$

How to find if a matrix is ill-conditioned

Basic Properties.

1. $||x|| \geq 0$ and $||x|| = 0$ if and only if $x = 0$.
2. $||x+y|| \leq ||x|| + ||y||$
3. $||\alpha x|| \leq |\alpha| \, ||x||$ where α is real
4. $||Ax|| \leq ||A|| \, ||x||$ where $||A|| \equiv \max \sum_i |a_{i,j}|$ (Natural matrix norm)

Matrix Norm:

$$Let\ x = \begin{bmatrix} -1 \\ 6 \\ -9 \end{bmatrix} and\ A = \begin{bmatrix} 1 & 3 & -4 \\ 1 & 3 & 1 \\ 3 & 0 & 5 \end{bmatrix}.$$

$$\|x\| = \max\{1, 6, 9\} = 9 \ and \ \|A\| = \max\{8, 5, 8\} = 8.$$

Matrix Eigenvalues and Eigenvectors

Eigenvalues are important in:

- Analysis of convergence characteristics of iterative methods for solving linear systems
- Determining the condition of a matrix → Ratio of the largest to the smallest eigenvalue of a matrix (if condition is large, matrix is ill-conditioned)
- Solving sets of linear differential equations
- Finding physical characteristics of a structure (principal stress, moments of inertia, …)

Eigenvalue (Characteristic-Value) Problems

Given: A X = C	A is a n by n matrix of coefficients
If C ≠ 0 and det(A) ≠ 0	This is a necessary condition for a unique solution
If C ≠ 0 and det(A) = 0	Either there is a family of solutions (rank(A)<n) or there is no solution (rank(A)>n) due to inconsistency (or incompatibility)
If C = 0 and det(A) ≠ 0	Trivial solution, i.e., X = 0
If C = 0 and det(A) = 0	There is a family of solutions

Eigenvalue (Characteristic-Value) Problems

Given: $A X = \lambda X$ or $(A - \lambda I) X = 0$	A is a n by n matrix of coefficients I is the identity matrix
Set $B = (A - \lambda I)$	The problem becomes $B X = 0$ This is the same as the $A X = C = 0$ case
Remember for $A X = C = 0$	Either, there is a family of solutions if $\det (A - \lambda I) = 0$ or there is only the trivial solution $X = 0$
$\det (A - \lambda I) = 0$	Gives the non-trivial solution, i.e., $X \neq 0$ Nontrivial solution: Eigenvalues λ_i
Every eigenvalue λ_i	Gives a solution vector, X_i called the eigenvector

Nonhomogeneous Set of Equations

- A x = y : Nonhomogeneous set of n equations

If the n equations are linearly independent, i.e.,

det(A)≠0 → *unique solution*

(one set of x values that makes the equations balance)

- To solve a non-homogeneous set of linear equations, use
  - Elimination methods such as Gauss or Gauss-Jordan
  - Matrix inverse
  - Cramer's rule
  - Iterative methods such as Jacobi, Gauss-Siedel, …

Homogeneous Set of Equations

- A x = 0 : Homogeneous set of n equations
  - Nontrivial solutions (solutions other than all x=0) are possible but generally *not unique*
  - Simultaneous equations establish relationships among x's that can be satisfied by various combinations of values.

Eigenvalue problems are typically of the general form:

$$
\begin{aligned}
&(A_{11}-\lambda)x_1 + A_{12}x_2 + \ldots + A_{1n}x_n = 0\\
&A_{21}x_1 + (A_{22}-\lambda)x_2 + \ldots + A_{2n}x_n = 0\\
&\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots\\
&A_{n1}x_1 + A_{n2}x_2 + \ldots + (A_{nn}-\lambda)x_n = 0
\end{aligned}
$$

$$
\begin{aligned}
&\left[\left[A\right]-\lambda\left[I\right]\right]\left[x\right]=0\\
&where\\
&\det\left(\left[\left[A\right]-\lambda\left[I\right]\right]\right)=0
\end{aligned}
$$

where λ is an unknown parameter (eigenvalue or characteristic value)

A solution x or [x] for such a system is the set of eigenvectors

Homogeneous Set of Equations r<n

- Ax=0 : Homogeneous set of n equations

If det(A)≠0, the unique solution is the *trivial solution* x=0.

If det(A)=0, there exists *nontrivial solutions* (solutions other than x=0),

i.e., r=rank (A) where r<n

This means there are:

- r independent equations
- r unknowns that can be evaluated independently
- (n-r) unknowns that must be chosen arbitrarily

If nonzero values are chosen for (n-r) unknowns:

Homogeneous set → Nonhomogeneous set of order n

- Use Gauss or Gauss-Jordan elimination to:
  - Solve non-homogeneous set
  - Solve homogeneous set first to determine number of independent equations r (rank of A) , then solve set of r nonhomogeneous independent equations to evaluate r unknowns

Homogeneous Set of Equations r<n

$(A - \lambda I_n)x = 0$ is a homogeneous linear system with coefficient matrix **$A - \lambda I_n$**

- Eigenvector **$x \neq 0$**, so find a nontrivial solution to the linear system

  $(A - \lambda I_n)x = 0$

- After Gauss elimination **$(A - \lambda I_n)$** must contain a zero row

  i.e. matrix **$A - \lambda I_n$** must be singular, so

  1. **$\det(A - \lambda In) = 0$** → **characteristic equation** of matrix **A** and solving it for λ gives the eigenvalues of A
  2. **$c(\lambda) = \det(A - \lambda I_n)$** → the **characteristic polynomial** of matrix **A** roots of the characteristic polynomial are the eigenvalues

- n eigenvalues of **A** are $\lambda_1, \lambda_2, \ldots, \lambda_n$: the corresponding eigenvectors are nontrivial solutions of the homogeneous linear system

  $(A - \lambda_i I_n)x = 0$ for i = 1,2, ..., n

Summary of computational approach for determining eigen-pairs (λ,**x**) as a two-step procedure:

1. To find the eigenvalues of **A** compute the roots of the characteristic equation **$\det(A - \lambda I_n) = 0$**
2. To find an eigenvector corresponding to an eigenvalue λ_i, compute a nontrivial solution to the homogeneous linear system **$(A - \lambda_i I_n)x = 0$**

Homogeneous Set of Equations r<n

- Ax=0 : Homogeneous set of linear algebraic equations

Special case of Ax=0 for:

- Vibrating systems
- Structure analysis
- Electric circuit system analysis
- Solution and stability analysis of ordinary differential equations

System of equations has the form: $A\,x = \lambda\,x$

λ: Eigenvalue (characteristic value) of matrix A

x: Eigenvector (characteristic vector) corresponding to λ

Linear Differential Equations x' = Ax

A : coefficient matrix with n distinct real eigenvalues

x : column vector of unknown functions of time

x' : dx/dt

x' = Ax+f , x' = Ax+f(t)

Applications: Time dependent mixing, heat conduction

There should be n linearly independent homogeneous solutions formed by the eigenvectors x_j and eigenvalues λ_j

Matrix Eigenvalues and Eigenvectors

- If $(A- \lambda)x=0$ then let $Ax = \lambda x$, where

A : n-by-n square matrix

x : column vector of size n

λ : a number

Then both the LHS and RHS of $Ax = \lambda x$ is a column vector of size n.

- Let **$Ax = \lambda x = 0$** for **$A \neq 0$**
- Which x would satisfy $Ax = 0$ for **$\lambda \neq 0$** ?

Matrix Eigenvalues and Eigenvectors

- Let **$Ax = \lambda x = 0$** for **$A \neq 0$** and **$\lambda \neq 0$**

Eigenvalues and **Eigenvectors** of A

- Rearranging $Ax = \lambda x = 0$ gives:

  $(A - \lambda I)x = 0$

  where I is an n-by-n identity matrix and $\lambda \cdot I$ makes a matrix the same size as A.

- $(A - \lambda I)x = 0$ is a system of homogeneous equations where

$\det(A - \lambda I) = 0$ for $\lambda \neq 0$

Characteristic Matrix and Characteristic Equation

For a linearly independent system

$$
\begin{aligned}
&(A_{11}-\lambda)x_1 + A_{12}x_2 + \ldots + A_{1n}x_n = 0\\
&A_{21}x_1 + (A_{22}-\lambda)x_2 + \ldots + A_{2n}x_n = 0\\
&\ldots \quad \ldots \quad \ldots \quad \ldots \quad \ldots\\
&A_{n1}x_1 + A_{n2}x_2 + \ldots + (A_{nn}-\lambda)x_n = 0
\end{aligned}
\qquad \longrightarrow \qquad
\begin{aligned}
&\big[[A]-\lambda[I]\big][x] = 0\\
&\textit{where}\\
&\det\Big(\big[[A]-\lambda[I]\big]\Big) = 0
\end{aligned}
$$

The nonsingular characteristic matrix A can be represented as

$$
|A-\lambda I| = \det\begin{pmatrix} A_{11}-\lambda & A_{12} & \ldots & A_{1n} \\ A_{21} & A_{22}-\lambda & \ldots & A_{2n} \\ \ldots & \ldots & \ldots & \ldots \\ A_{n1} & A_{n2} & \ldots & A_{nn}-\lambda \end{pmatrix} = 0
$$

The determiant can be expanded to yield a polynomial of n^{th} degree

$$
\lambda^n - \alpha_1\lambda^{n-1} - \alpha_2\lambda^{n-2} - \ldots - \alpha_n = 0\,, \qquad \text{where } \alpha_i = f(A_{ij})
$$

Characteristic Equation (Polynomial)

$$f(\lambda) = \lambda^n - \alpha_1\lambda^{n-1} - \alpha_2\lambda^{n-2} - \ldots - \alpha_n = 0\,, \text{ where } \alpha_i = f(A_{ij})$$

- **f(λ) = det(A- I)** →Characteristic polynomial of matrix A

  f(λ) : polynomial of **λ** of order n
- Has n roots → eigenvalues of A
- Roots may be real distinct, real repeated, or complex depending on A
- A nonsingular real symmetric matrix of order n has n real nonzero eigen-values and n linearly independent eigenvectors
- Eigenvectors of a real symmetric matrix are orthogonal to each other

Ex: Singular real symmetric matrix

Find the eigenvalues and eigenvectors for Ax=0, where $A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$

Calculating eigenvalues: $\det(A-\lambda I) = 0 = \det\begin{pmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{pmatrix} = \lambda^2 - 4\lambda + 3,$ where $\lambda_1 = 1$ and $\lambda_2 = 3$

Inserting $\lambda_1 = 1$ in $\det(A-\lambda I)$ gives:

$\begin{pmatrix} 2-1 & 1 \\ 1 & 2-1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ which can be written as $1*x_1 + 1*x_2 = 0,$ *i.e.*, $x_1 = -x_2$

Infinite number of eigenvectors would satisfy this condition, such as $\begin{pmatrix} 5 \\ -5 \end{pmatrix}, \begin{pmatrix} 0.1 \\ -0.1 \end{pmatrix}, \ldots$

Inserting $\lambda_2 = 3$ in $\det(A-\lambda I)$ gives:

$\begin{pmatrix} 2-3 & 1 \\ 1 & 2-3 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ which can be written as $-1*x_1 + 1*x_2 = 0$ and $1*x_1 - 1*x_2 = 0,$ *i.e.*, $x_1 = x_2$

Infinite number of eigenvectors would satisfy this condition, such as $\begin{pmatrix} 5 \\ 5 \end{pmatrix}, \begin{pmatrix} 0.1 \\ 0.1 \end{pmatrix}, \ldots$

Ex Contd.: Orthogonal Eigenvectors

For eigenvectors to have the orthonormal property, they must satisfy QQ'=I

Remember: An orthogonal matrix is whose transpose is its inverse, i.e., $Q'=Q^{-1}$

For example:

$$QQ' = \begin{pmatrix} c_1 & c_2 \\ -c_1 & c_2 \end{pmatrix}\begin{pmatrix} c_1 & -c_1 \\ c_2 & c_2 \end{pmatrix} = \begin{pmatrix} c_1^2 + c_2^2 & -c_1^2 + c_2^2 \\ -c_1^2 + c_2^2 & c_1^2 + c_2^2 \end{pmatrix} = I = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$c_1^2 + c_2^2 = 1$ and $-c_1^2 + c_2^2 = 0$ gives $c_1^2 = c_2^2 = 0.5$

Accordingly, $c_1 = \frac{1}{\sqrt{2}}$ or $c_1 = -\frac{1}{\sqrt{2}}$ and $c_2 = \frac{1}{\sqrt{2}}$ or $c_2 = -\frac{1}{\sqrt{2}}$

Consequently, for this problem, various Q's can be obtained, such as

$$Q = \begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{pmatrix}$$

Ex: Eigenvalues in MATLAB®

```
>> A=[2 1; 1 2]
A =
     2     1
     1     2
>> det(A)
ans =     3
>> eig(A)
ans =
     1
     3
>> [Q,d]=eig(A)
Q =
  -0.70710678118655   0.70710678118655
   0.70710678118655   0.70710678118655
d =
     1     0
     0     3
```

eig(A) : Gives the eigenvalues of matrix A in vector

[Q,d]=eig(A): Gives Q and d

Q : Square matrix of the eigenvectors of A

d : Square diagonal matrix whose diagonal has the eigenvalues of A

Ex: Electric circuit system analysis

Use eig(A) in MATLAB® to determine the stability of electric circuits. For the stability of an electric circuit, the necessary and sufficient condition is: eigenvalues of the system should be on the left hand side of the complex number axis and, if an imaginary axis exists, then the eigenvalues on the imaginary axis should not have any multiplicates.

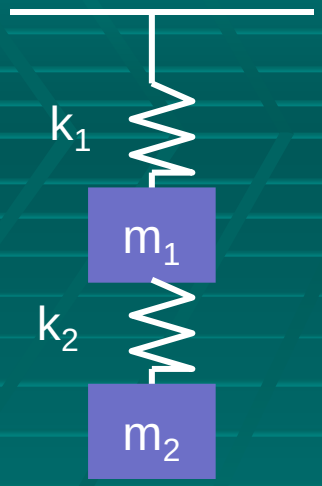
Determine the stability of an electric circuit with the following equation:

$$\frac{d}{dt}\begin{pmatrix} V_1(t) \\ i_2(t) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ -1 & -2 \end{pmatrix}\begin{pmatrix} V_1(t) \\ i_2(t) \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \end{pmatrix} U(t)$$

```
>> A=[1 0; -1 -2]
A =
     1     0
    -1    -2
>> eig(A)
ans =
    -2
     1

% This circuit is not stable.
```

Ex: Vibrating Spring-Mass System



Governing equations:

$$\left.\begin{aligned} m_1 \frac{d^2 y_1}{dt^2} &= -k_1 y_1 + k_2 (y_2 - y_1) \\ m_2 \frac{d^2 y_2}{dt^2} &= -k_2 (y_2 - y_1) \end{aligned}\right\} \rightarrow \begin{pmatrix} -\frac{k_2 + k_1}{m_1} & \frac{k_2}{m_1} \\ \frac{k_2}{m_2} & -\frac{k_2}{m_2} \end{pmatrix} \begin{pmatrix} y_1(t) \\ y_2(t) \end{pmatrix} = \begin{pmatrix} y_1'' \\ y_2'' \end{pmatrix}$$

What is $y_1(t)$ and $y_2(t)$?

Assume a solution of the form: $y_1(t) = x_1 e^{wt}$ and $y_2(t) = x_2 e^{wt}$,

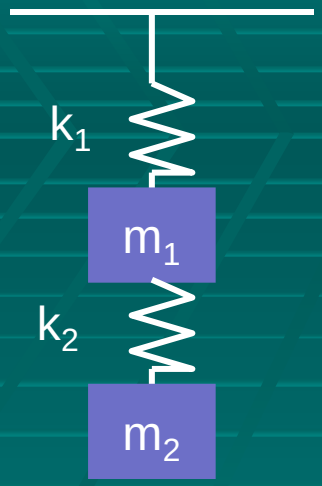
where x_1 and x_2 are independent of t.

Then the governing quations become:

$$\begin{pmatrix} -\frac{k_2 + k_1}{m_1} & \frac{k_2}{m_1} \\ \frac{k_2}{m_2} & -\frac{k_2}{m_2} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = w^2 \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

Note that this is in the same form as $Ax = \lambda x$, where $\lambda = w^2$

Ex: Vibrating Spring-Mass System



Let $m_1 = m_2 = 1$ and $k_1 = 3$, $k_2 = 2$. Then the matrix becomes:

$$\begin{pmatrix} -5 & 2 \\ 2 & -2 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \lambda \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \quad \text{or} \quad \begin{pmatrix} -5-\lambda & 2 \\ 2 & -2-\lambda \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0$$

For a nontrivial solution $\rightarrow$ $\det(A - \lambda I) = 0$

$$\det(A - \lambda I) = (-5-\lambda)(-2-\lambda) - 4 = \lambda^2 + 7\lambda + 6 = 0$$

This gives two solutions: $\lambda_1 = -6$ and $\lambda_2 = -1$

$$\text{For } \lambda_1 = -6 \rightarrow \begin{pmatrix} -5+6 & 2 \\ 2 & -2+6 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0 \rightarrow x_1 = -2x_2 \rightarrow X_1 = C_1\begin{pmatrix} -2 \\ 1 \end{pmatrix}$$

$$\text{For } \lambda_2 = -1 \rightarrow \begin{pmatrix} -5+1 & 2 \\ 2 & -2+1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0 \rightarrow x_1 = \tfrac{1}{2}x_2 \rightarrow X_2 = C_2\begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

where X_1 and X_2 are eigenvectors and C_1 and C_2 are arbitrary numbers.

General solutions:

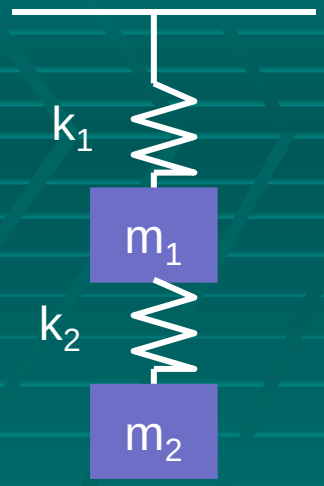
$$\text{For } \lambda_1 = -6 \rightarrow y_1(t) = -2C_1 e^{\sqrt{-6}\,t} = -2C_1 e^{i\sqrt{6}\,t}$$

$$y_2(t) = C_1 e^{\sqrt{-6}\,t} = C_1 e^{i\sqrt{6}\,t}$$

$$\text{For } \lambda_2 = -1 \rightarrow y_1(t) = C_2 e^{\sqrt{-1}\,t} = C_2 e^{it}$$

$$y_2(t) = 2C_2 e^{\sqrt{-1}\,t} = 2C_2 e^{it}$$

Ex: Vibrating Spring-Mass System



Converting the general solution to matrix notation:

Using Euler's equation, i.e., $e^{iz} = \cos(z) + i\sin(z)$

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} -2 \\ 1 \end{pmatrix}\left[a\cos(\sqrt{6}t) + b\sin(\sqrt{6}t)\right] + \begin{pmatrix} 1 \\ 2 \end{pmatrix}\left[c\cos(t) + d\sin(t)\right]$$

There are four unknown constants a, b, c and d, which are complex numbers

Therefore, we need four conditions (initial or boundary conditions).

Suppose the following initial conditions are given:

Displacements	Velocities
$y_1(0) = 1$	$y_1'(0) = -2\sqrt{6}$
$y_2(0) = 2$	$y_2'(0) = \sqrt{6}$

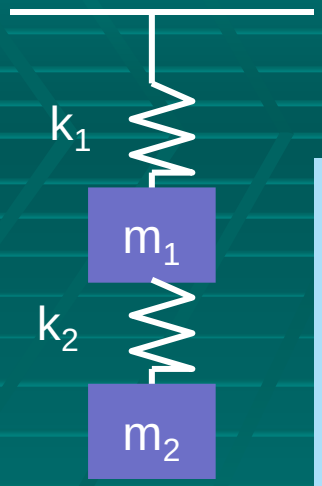
The solution in matrix notation is given as:

$$\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} -2 \\ 1 \end{pmatrix}\left[\sin(\sqrt{6}t)\right] + \begin{pmatrix} 1 \\ 2 \end{pmatrix}\left[\cos(t)\right]$$

The solution in open form is given as:

$$y_1(t) = -2\sin(\sqrt{6}\ t) + \cos(t) \quad \text{and} \quad y_2(t) = \sin(\sqrt{6}\ t) + 2\cos(t)$$

Ex: Vibrating Spring-Mass System



Note the following:

The coefficient matrix $A = \begin{pmatrix} -5 & 2 \\ 2 & -2 \end{pmatrix}$ is a symmetrical matrix

Eigenvectors $X_1 = \begin{pmatrix} -2 \\ 1 \end{pmatrix}$ and $X_2 = \begin{pmatrix} 1 \\ 2 \end{pmatrix}$ are othogonal to each other,

i.e., $X_1^T X_2 = 0$

Eigenvalues $\lambda_1 = -6$ and $\lambda_2 = -1$ are real

Trace of A is $tr(A) = \sum a_{ii} = \sum \lambda_i$, i.e., the sum of eigenvalues

Roots of Characteristic Polynomial-Eigenvalues

Matrix	Eigenvalue
Singular, det(A)=0	At least one zero eigenvalue
Nonsingular, det(A)≠0	No zero eigenvalues
Symmetric, A=A’	All real eigenvalues
Hermitian (*see next slide*)	All real eigenvalues
Zero matrix, A=0	All zero eigenvalues
Identity, A= I	All unity eigenvalues
Diagonal, A=D	Equal to diagonal elements of A
Inverse, A^{-1}	Inverse eigenvalues of A
Transformed, $B=Q^{-1}AQ$	Eigenvalues of B = eigenvalues of A

Ref: A. Constantinides and N. Moustoufi, Numerical Methods for Chemical Engineers with MATLAB® Applications, 1999, Prentice Hall PTR, p.123

What is a Hermitian (Self-Adjoint) Matrix

A square matrix with complex entries which is equal to its own conjugate transpose

This means the element in the *i*th row and *j*th column is equal to the complex conjugate of the element in the *j*th row and *i*th column, for all indices *i* and *j*

Ex: If the conjugate transpose of a matrix A is shown by A', then A=A'

$$A = \begin{pmatrix} 3 & 2+i \\ 2-i & 1 \end{pmatrix}$$

Every Hermitian matrix is normal.

It can be diagonalized by a unitary matrix, and the resulting diagonal matrix has only real entries. This means that all eigenvalues of a Hermitian matrix are real, and, moreover, eigenvectors with distinct eigenvalues are orthogonal

Methods for Finding Eigenvalues

- Power Method: Iterative procedure for finding a specific eigenvalue and its associated eigenvector for a given matrix A

The basic power method finds the dominant (largest magnitude) eigenvalue

The inverse power method finds the smallest eigenvalue

Methods for Finding Eigenvalues

- QR Method: Uses QR factorization to find all the eigenvalues for a given matrix A
  - Produces a sequence of matrices that converge to a similar matrix for which eigenvalues are easy to find
  - Does not directly find the eigenvectors, therefore inverse iteration can be used to find the eigenvector associated with any specified eigenvalue

Power Method

If x is an eigenvector of A:

Convert λx=Ax into a sequence of approximations of λ and x

- Start with an initial guess z for the eigenvector x: w=Az
- If z is an eigenvector, for any component of z: $\lambda z_k = w_k$
- If z is not an eigenvector, normalize w and use it as the next approximation:

$$z^{(1)} = [1,1,\ldots,1]', \quad w^{(1)} = Az^{(1)}$$

$$z^{(2)} = \frac{1}{w_k^{(1)}} w^{(1)}, \quad w^{(2)} = Az^{(2)}$$

$$z^{(3)} = \frac{1}{w_k^{(2)}} w^{(2)}, \quad w^{(3)} = Az^{(3)}, \ldots$$

$$z^{(1)} = [1,1,\ldots,1]', \quad w^{(1)} = Az^{(1)}$$

$$z^{(2)} = \frac{1}{w_k^{(1)}} w^{(1)} = \frac{1}{w_k^{(1)}} Az^{(1)}$$

$$w^{(2)} = Az^{(2)} = A\frac{1}{w_k^{(1)}} Az^{(1)} = \frac{1}{w_k^{(1)}} A^2 z^{(1)}$$

$$z^{(3)} = \frac{1}{w_k^{(2)}} w^{(2)} = \frac{1}{w_k^{(2)}} \frac{1}{w_k^{(1)}} A^2 z^{(1)}$$

$$w^{(3)} = Az^{(3)} = A\frac{1}{w_k^{(2)}} \frac{1}{w_k^{(1)}} A^2 z^{(1)}$$

$$= \frac{1}{w_k^{(2)}} \frac{1}{w_k^{(1)}} A^3 z^{(1)}$$

With each iteration, the original estimate for z is multiplied with ***powers*** of A → ***Power method***

Ex: Basic Power Method

$$A = \begin{pmatrix} 23 & 9 & -3 \\ 11 & 10 & 2 \\ 5 & -4 & 16 \end{pmatrix}$$

Find the largest eigenvalue and the associated eigenvector of

```
>> A=[23 9 -3; 11 10 2; 5 -4 16];
>> z=[1;1;1];                    %Start with the initial x vector
>> w=A*z; w‘                     %Compute w=A*z
ans =    29    23    17
>> z=w/w(1); z‘                  %Normalize w using the largest value and use as a guess
ans =    1.00000000000000   0.79310344827586   0.58620689655172
>> w=A*z; w'
ans =   28.37931034482759  20.10344827586207  11.20689655172414
>> z=w/w(1); z'
ans =    1.00000000000000   0.70838396111786   0.39489671931956
>> w=A*z; w'
ans =  28.19076549210207  18.87363304981774   8.48481166464155
>> z=w/w(1); z'
ans =   1.00000000000000   0.66949700443946   0.30097840610319
>> w=A*z; w'
ans =  28.12253782164562  18.29692685660101   7.13766647989311
…
ans =  28.06845698567594  17.72557765083737   5.75573908993695
```

Ex: Basic Power Method

$$A = \begin{pmatrix} 23 & 9 & -3 \\ 11 & 10 & 2 \\ 5 & -4 & 16 \end{pmatrix}$$

Find the largest eigenvalue and the associated eigenvector of

```
>> A=[23 9 -3; 11 10 2; 5 -4 16];
>> z=[1;1;1];                    %Start with the initial vector
>> w=A*z; w'                     %Compute w=A*z
ans =    29    23    17
>> z=w/w(1); z'                  %Normalize w using the largest value and use as a guess
ans =    1.00000000000000   0.79310344827586   0.58620689655172
```

…

After 15 iterations the estimate of the

largest eigenvalue: $\lambda \approx 28.06845698567594$

eigenvector z': $\approx$ 1.00000000000000 0.63153431091383 0.20511727084952

error : A*z-w(1)*z = 0 -0.00061599002606 -0.00158620392198

max norm of the error: $||Az - \lambda z||_\infty = 0.00158620392198$

In MATLAB® :

```
>> [eig(A)]'
ans = 28.06839976386356     6.58643222669528    14.34516800944117
```

The largest eigenvalue is 28.06839976 and successive approximations of the power method converge towards this value

Ex: Basic Power Method

```
function [z,wmax]=BasicPower(A,imax,tol)
[m,n]=size(A);              % Get the dimensions of A
w=ones(m,1);                % Start with initial guess eigenvalues
disp('iter    wmax    r      z(1)      z(2)      z(3)       z(4)')
for i=1:imax
   [kk,k]=max(abs(w)); % Returns the indices of the maximum w values in vector k
   z=w/w(k);                % Normalize w with respect to maxw=w(k)
   w=A*z;                   % Calculate w again
   wmax=w(k);               % z(k)=1
   r=norm(wmax*z-w);  % Use Euclidean form (norm(w,p) = sum(abs(w).^p)^(1/p))
   final=[i,wmax,r,z'];
   disp(final)
   if r<tol, disp('Power method has converged'), break
   end
end
```

Ex: Basic Power Method in MATLAB®

```
>> A=[23 9 -3; 11 10 2; 5 -4 16]; [z,wmax]=BasicPower(A,20,.001);
```

iter	wmax	r	z(1)	z(2)	z(3)	z(4)
1.0000	29.0000	13.4164	1.0000	1.0000	1.0000	
2.0000	28.3793	5.9378	1.0000	0.7931	0.5862	
3.0000	28.1908	2.8656	1.0000	0.7084	0.3949	
4.0000	28.1225	1.4289	1.0000	0.6695	0.3010	
5.0000	28.0941	0.7223	1.0000	0.6506	0.2538	
6.0000	28.0811	0.3672	1.0000	0.6412	0.2299	
7.0000	28.0748	0.1872	1.0000	0.6364	0.2177	
8.0000	28.0716	0.0956	1.0000	0.6340	0.2115	
9.0000	28.0700	0.0488	1.0000	0.6328	0.2083	
10.0000	28.0692	0.0249	1.0000	0.6321	0.2067	
11.0000	28.0688	0.0127	1.0000	0.6318	0.2059	
12.0000	28.0686	0.0065	1.0000	0.6317	0.2054	
13.0000	28.0685	0.0033	1.0000	0.6316	0.2052	
14.0000	28.0685	0.0017	1.0000	0.6315	0.2051	
15.0000	28.0684	0.0009	1.0000	0.6315	0.2051	

Power method has converged

$$A = \begin{pmatrix} 23 & 9 & -3 \\ 11 & 10 & 2 \\ 5 & -4 & 16 \end{pmatrix}$$

Maximum eigenvalue

$$\lambda_{\max} = 28.0684$$

Associated eigenvector

$$z = [1.0000 \; 0.6315 \; 0.2051]'$$

Inverse Power Method

Gives the smallest eigenvalue of A and is based on the fact that:

Eigenvalues of $B=A^{-1}$ are the reciprocals of the eigenvalues of A

To find the smallest λ of A (by avoiding computation of A^{-1}) first find w_{max} (dominant λ of B):

Apply the power method to $B=A^{-1}$ to find w_{max}

Ex: Inverse Power Method in MATLAB®

```
function [z,wmin]=PowerInverse(A,imax,tol)
[m,n]=size(A);              % Get the dimensions of A
L=eye(m);                   % Create an identity matrix
U=A;                        % Set U equal to A o start with
for j=1:m
   for i=j+1:m,          L(i,j)=U(i,j)/U(j,j);
                         U(i,:)=U(i,:)-L(i,j)*U(j,:);
   end
end,  A,U,L,
w=ones(m,1);
disp('iter  wmax  r   z(1)    z(2)    z(3)    z(4)')
for i=1:imax
z=w/norm(w);    w=LUsolve(L,U,z);  %Uses LU decomposition
wmax=z'*w;     r=norm(wmax*z-w);
final=[i, wmax, r, z'];       disp(final)
    if r<tol, disp('Inverse Power Method has converged'), break, end
end
wmin=1/wmax,  % Smallest eigenvalue is the reciprocal of wmax of
```

Ex: Inverse Power Method in MATLAB®

```
>> [z,wmin]=PowerInverse(A,20,0.001)

A =
    23     9    -3
    11    10     2
     5    -4    16
U =
   23.0000    9.0000   -3.0000
         0    5.6957    3.4348
         0         0   20.2443
L =
    1.0000         0         0
    0.4783    1.0000         0
    0.2174   -1.0458    1.0000
```

Ex: Inverse Power Method in MATLAB®

iter	wmax	r	z(1)	z(2)	z(3)	z(4)
1.0000	0.0498	0.0139	0.5774	0.5774	0.5774	
2.0000	0.0650	0.0175	0.3538	0.5812	0.7328	
3.0000	0.0854	0.0181	0.1003	0.5941	0.7981	
4.0000	0.1075	0.0172	-0.1005	0.6388	0.7627	
5.0000	0.1267	0.0134	-0.2237	0.6965	0.6818	
6.0000	0.1392	0.0084	-0.2864	0.7398	0.6088	
7.0000	0.1459	0.0045	-0.3151	0.7645	0.5624	
8.0000	0.1491	0.0022	-0.3278	0.7769	0.5375	
9.0000	0.1506	0.0010	-0.3334	0.7829	0.5253	
10.0000	0.1512	0.0005	-0.3360	0.7857	0.5195	

Inverse Power Method has converged

wmin = 6.6120

z =

-0.3360

0.7857

0.5195

QR Method

To find eigenvalues of a real matrix A:

- Generate a sequence of matrices **$A^{(m)}$ that are orthogonally similar to A**, i.e., have the same eigenvalues as A
- Find the eigenvalues from the matrix to which the $A^{(m)}$ sequence converges

QR Factorization Using Householder Transformations

- Suppose that **A** is a real symmetric matrix.
- Householder's method is used to construct a similar symmetric **tridiagonal matrix**. Then the QR Method can be used to find all eigenvalues of the tridiagonal matrix. A=QR where Q is orthogonal and R is upper triangular

$$
\begin{aligned}
A &= (H_1)(H_1A) && = Q_1R_1 \\
&= (Q_1H_2)(H_2R_1) && = Q_2R_2 \\
&= (Q_2H_3)(H_3R_2) && = Q_3R_3 \\
&= \ldots \\
&= (Q_{n-2}H_{n-1})(H_{n-1}R_{n-2}) && = QR
\end{aligned}
$$

H_k : Householder matrix that zeros out the k^{th} column, below the diagonal, of the matrix it multiplies

QR Factorization Using Householder Transformations

```
%QR Factorization using Householder transformations
%Ref: L.V.Fausett, Applied Numerical Analysis Using Matlab, 2nd ed.,p.164
 function [Q,R]=QRfactor(A)
[m,n]=size(A);
R=A;              %Start with R=A
Q=eye(m);       %Set Q as the identity matrix
for k=1:m-1
    x=zeros(m,1);
    x(k:m,1)=R(k:m,k);
    g=norm(x);
    v=x;        v(k)=x(k)+g;
    %Orthogonal transformation matrix that eliminates one element
    %below the diagonal of the matrix it is post-multiplying:
    s=norm(v);
        if s~=0, w=v/s; u=2*R'*w;
        R=R-w*u';          %Product HR
        Q=Q-2*Q*w*w';  %Product QR
    end
end
```

QR Method

To see that **$A^{(m)}$ (that are orthogonally similar to A)** in the sequence are similar:

1. $A=QR \rightarrow R=Q'A$
2. $A^{(1)} \equiv RQ \rightarrow A^{(1)}=Q'AQ$

Thus, $A^{(1)}$ is similar to A

Factor $A^{(1)}= Q^{(1)}R^{(1)}$ and form $A^{(2)}= Q^{(1)}R^{(1)}$

R: Upper triangular matrix

Q: Unitary matrix

Ref: L. V. Fausett, Applied Numerical Analysis Using MATLAB, 2nd Ed, Pearson Prentice Hall 2008, Chap.5

QR Method: Outline

To find all eigenvalues of matrix A:

Define A(1)=A

for k=1:kmax

  Factor A(k)=Q(k)*R(k)

  Define A(k+1)=R(k)*Q(k)

end

Ref: L. V. Fausett, Applied Numerical Analysis Using MATLAB, 2nd Ed, Pearson Prentice Hall 2008, Chap.5

Ex: QR Method in MATLAB®

```
%QR method to find eigenvalues
%Using QRfactorization
%A=QR => R=Q'A where Q'=inv(Q)
function eigen=QReigen(A,imax)
for i=1:imax,
    [Q,R]=QRfactor(A);    %Personally created function
    A=R*Q;                %Defining new A=RQ=Q'AQ
end
eigen=diag(A)
```

Ex: Orthogonal-Triangular Decomposition in MATLAB®

```
>> A=[-2 2 3; 1 3 5; -3 -1 2]
A =
    -2     2     3
     1     3     5
    -3    -1     2

>> [Q,R]=qr(A)
Q =
   -0.5345   -0.6172   -0.5774
    0.2673   -0.7715    0.5774
   -0.8018    0.1543    0.5774
R =
    3.7417    0.5345   -1.8708
         0   -3.7033   -5.4006
         0         0    2.3094
>> Q*R
ans =
   -2.0000    2.0000    3.0000
    1.0000    3.0000    5.0000
   -3.0000   -1.0000    2.0000
```

QR Orthogonal-triangular decomposition

[Q,R] = QR(A), where A is m-by-n, produces an m-by-n upper triangular matrix R and an m-by-m unitary matrix Q so that A = Q*R.

[Q,R] = QR(A,0) produces the "economy size" decomposition. If m>n, only the first n columns of Q and the first n rows of R are computed. If m<=n, this is the same as [Q,R] = QR(A)

Example: The least squares approximate solution to A*x = b, i.e. min|A*x-b|, can be found with the Q-less QR decomposition and one step of iterative refinement:

```
if issparse(A), R = qr(A); else R = triu(qr(A)); end
x = R\(R'\(A'*b));
r = b - A*x;
e = R\(R'\(A'*r));
x = x + e;
```

Ex1: QR Decomposition

```
>> A =  [ 1 2 3; 4 5 6; 7 8 9; 10 11 12 ]
A =
     1     2     3
     4     5     6
     7     8     9
    10    11    12
>> [Q,R]=qr(A)
Q =
   -0.0776   -0.8331    0.5473   -0.0221
   -0.3105   -0.4512   -0.7133    0.4373
   -0.5433   -0.0694   -0.2153   -0.8085
   -0.7762    0.3124    0.3813    0.3932
R =
  -12.8841  -14.5916  -16.2992
         0   -1.0413   -2.0826
         0         0   -0.0000
         0         0         0
```

- This is a rank-deficient matrix; the middle column is the average of the other two columns. The rank deficiency is revealed by the factorization:

- The triangular structure of R gives it zeros below the diagonal; the zero on the diagonal in R(3,3) implies that R, and consequently A, does not have full rank

```
>> A =  [ 1 2 3; 4 5 6; 7 8 9; 10 11 12 ];   y=[2; 4; 6; 8];
>> x=A\y
Warning:Rank deficient, rank=2, tol=1.4594e-014.
x = [ -0.0000;  0;   0.6667]
>> [Q,R,E] = qr(A)
Q =
   -0.1826   -0.8165    0.5465    0.0363
   -0.3651   -0.4082   -0.7558    0.3589
   -0.5477   -0.0000   -0.1280   -0.8268
   -0.7303    0.4082    0.3373    0.4316
R =
  -16.4317  -12.7802  -14.6059
         0    1.6330    0.8165
         0         0    0.0000
         0         0         0
E =
     0     1     0
     0     0     1
     1     0     0
>> tol = max(size(A))*eps*abs(R(1,1))
tol =  1.4594e-014
>> z=Q'*y;    x = R\z
Warning:Rank deficient,rank=2, tol=1.4594e-014.
x = [ -0.0000;  0;   0.6667]
```

Ex2: QR Decomposition

- The QR factorization is used to solve linear systems with more equations than unknowns
- Linear system Ax=y represents 4 equations in only 3 unknowns. The best solution in a least squares sense is computed by x=A\y
- The quantity tol is a tolerance used to decide if a diagonal element of R is negligible. If [Q,R,E] = qr(A), then

  tol=max(size(A))*eps*abs(R(1,1))
- [Q,R,E] = qr(A) for full matrix A, produces a permutation matrix E, an upper triangular matrix R with decreasing diagonal elements, and a unitary matrix Q so that A*E = Q*R. The column permutation E is chosen so that abs(diag(R)) is decreasing.
- The solution x is computed using the factorization and the two steps

  z = Q'*b; x = R\z
- Ax equals y to within roundoff error, which indicates that even though the simultaneous equations Ax=y are overdetermined and rank deficient, they happen to be consistent. There are infinitely many solution vectors x; the QR factorization has found just one of them.

Ex: Eigenvalues in MATLAB®

```
>> [u,s,v]=svd(A)
u =
   -0.5476    0.3027   -0.7800
   -0.7997   -0.4636    0.3815
   -0.2462    0.8327    0.4960
s =
    7.0172         0         0
         0    3.9254         0
         0         0    1.1617
v =
    0.1473   -0.9088    0.3904
   -0.4629   -0.4122   -0.7847
   -0.8741    0.0651    0.4814
>> u*s*v'
ans =
   -2.0000    2.0000    3.0000
    1.0000    3.0000    5.0000
   -3.0000   -1.0000    2.0000
```

SVD Singular value decomposition

[U,S,V] = SVD(X) produces a diagonal matrix S, of the same dimension as X and with nonnegative diagonal elements in decreasing order, and unitary matrices U and V so that X = U*S*V'.

S = SVD(X) returns a vector containing the singular values.

[U,S,V] = SVD(X,0) produces the “economy size” decomposition. If X is m-by-n with m > n, then only the first n columns of U are computed and S is n-by-n.

For m <= n, SVD(X,0) is equivalent to SVD(X).

When the eigenvalues (and eigenvectors) of a symmetric matrix are known, the following values are easily calculated:

- **Condition number**: For a nonsingular matrix *A,* it is defined as cond(*A*) = $||A||_2 * ||A-1||_2$. In case of a symmetric matrix, it is the absolute value of the quotient of the largest and smallest eigenvalue. Matrices with large condition numbers can cause numerically unstable results, i.e., small perturbation can result in large errors. Hilbert matrices are the most famous ill-conditioned matrices. For ex: Fourth order Hilbert matrix has a condition of 15514, while for order 8 it is $2.7 * 10^8$.
- **Rank:** A matrix *A* has rank r if it has r columns which are linearily independent while the remaining columns are linearly dependent on these.

  Equivalently, r is the dimension of the range of A. Furthermore it is the number of nonzero singular values. In case of a symmetric matrix r is the number of nonzero eigenvalues. Due to rounding errors, numerical approximations of zero eigenvalues may not be zero (it may also happen that a numerical approximation is zero while the true value is not). Thus, one can only calculate the *numerical* rank by making a decision which of the eigenvalues are close enough to zero.
- **Singular values**: For a (square) matrix *A,* these are the square roots of the (non negative) eigenvalues of A^*A. In case of a symmetric matrix $S \rightarrow S'S = S^2$, and the singular values of *S* are the absolute values of the eigenvalues of *S*
- **2-Norm and spectral radius:** For a matrix *A,* it is the norm based on the euclidian vectornorm, i.e. the largest value $||Ax||_2$ when x runs through all vectors with $||x||_2=1$. It is the largest singular value of *A*. In case of a symmetric matrix, it is the largest absolute value of its eigenvectors and thus equal to its spectral radius.

Ref: http://en.wikipedia.org/wiki/Jacobi_eigenvalue_algorithm

For More Online Information

- http://www.sosmath.com/diffeq/system/linear/eigenvalue/eigenvalue.html
- http://mathworld.wolfram.com/Eigenvalue.html
- http://www.miislita.com/information-retrieval-tutorial/matrix-tutorial-3-eigenvalues-eigenvectors.html
- http://math.fullerton.edu/mathews/n2003/EigenvaluesMod.html
- http://math.fullerton.edu/mathews/n2003/HessenbergMod.html
- http://math.fullerton.edu/mathews/n2003/HouseholderMod.html