Numerical Analysis and Optimization

Lecture 10 – Monday May 8, 2017

Outline

- Polynomial Functions
- Solutions to Systems of Linear Equations
- Optimization Problem
- Matlab Optimization Toolbox
- Design Problems

Outline

- **Polynomial Functions**
- Solutions to Systems of Linear Equations
- Optimization Problem
- Matlab Optimization Toolbox
- Design Problems

Polynomial Functions

A polynomial function is a function of a single variable that can be expressed in the general form:

$$f(x) = a_0 x^N + a_1 x^{N-1} + a_2 x^{N-2} + \cdots + a_{N-2} x^2 + a_{N-1} x + a_N$$

Where the variable is x and the polynomial coefficient are represented by the values a_o, $a_1 \ldots . a_n$.

The degree of a polynomial is equal to the largest value used as an exponent.

$g(x) = a_0 x^3 + a_1 x^2 + a_2 x + a_3$ ⇨ Cubic (degree 3) polynomial

$h(x) = x^3 - 2x^2 + 0.5x - 6.5.$ ⇨ Example for Cubic polynomial

Polynomial Functions

There are several ways to evaluate a polynomial using MATLAB

Example: $f(x) = 3x^4 - 0.5x^3 + x - 5.2$

```
f = 3*x^4 - 0.5*x^3 + x - 5.2;
```

```
f = 3*x.^4 - 0.5*x.^3 + x - 5.2;
```

If x is a vector or a matrix

`polyval(a,x)`	Evaluates a polynomial with coefficients `a` for the values in `x`. The result is a matrix the same size as `x`.

```
a = [3,-0.5,0,1,-5.2];
f = polyval(a,x);
```

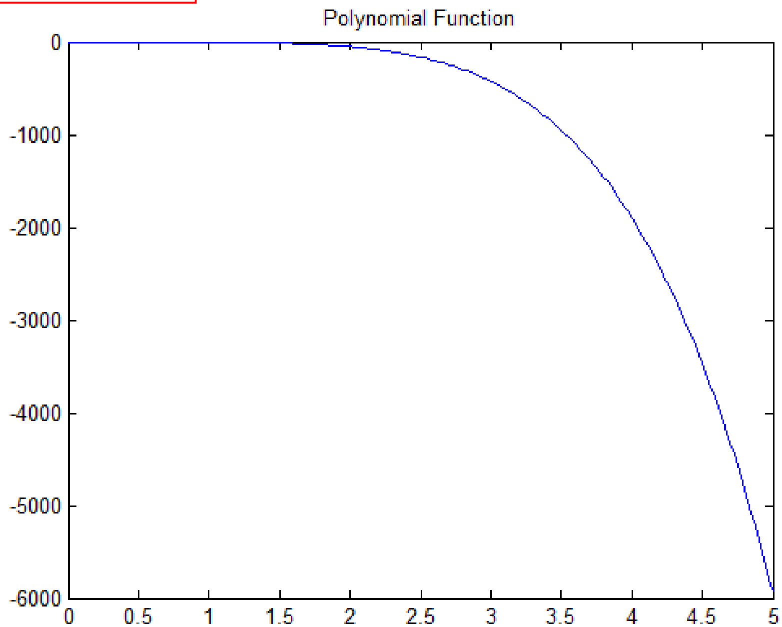
or

```
f = polyval([3,-0.5,0,1,-5.2],x);
```

Polynomial Functions

This code will generate 201 points of the polynomial over the desired interval.

```
x=0:5/200:5;
a=[-2,0,3,-2.5,0,-2.5];
g=polyval(a,x);
plot(x,g),title('Polynomial Function')
set(gcf,'color','w');
```



Polynomial Functions

- **Polynomial Operations**

$$g(x) = x^4 - 3x^2 - x + 2.4$$

$$h(x) = 4x^3 - 2x^2 + 5x - 16$$

$$s(x) = g(x) + h(x)$$

MATLAB statements to perform this polynomial addition are

```
g = [1,0,-3,-1,2.4];
h = [0,4,-2,5,-16];
s = g + h;
```

Polynomial Functions

MATLAB contains functions to perform polynomial multiplication and division:

`conv(a,b)`	Computes a coefficient vector that contains the coefficients of the product of polynomials represented by the coefficients in `a` and `b`. The vectors `a` and `b` do not have to be the same size.
`[q,r] = deconv(n,d)`	Returns two vectors. The first vector contains the coefficients of the quotient and the second vector contains the coefficients of the remainder polynomial.

Polynomial Functions

- **Example:**

$$g(x) = (3x^3 - 5x^2 + 6x - 2)(x^5 + 3x^4 - x^2 + 2.5)$$

```
a = [3,-5,6,-2];
b = [1,3,0,-1,0,2.5];
g = conv(a,b);
```

$$g(x) = 3x^8 + 4x^7 - 9x^6 + 13x^5 - x^4 + 1.5x^3 - 10.5x^2 + 15x - 5$$

Polynomial Functions

$$g(x) = 3x^8 + 4x^7 - 9x^6 + 13x^5 - x^4 + 1.5x^3 - 10.5x^2 + 15x - 5$$

$$h(x) = \frac{3x^8 + 4x^7 - 9x^6 + 13x^5 - x^4 + 1.5x^3 - 10.5x^2 + 15x - 5}{x^5 + 3x^4 - x^2 + 2.5}$$

```
g = [3,4,-9,13,-1,1.5,-10.5,15,-5];
b = [1,3,0,-1,0,2.5];
[q,r] = deconv(g,b);
```

As expected, the quotient coefficient vector is [3,-5,6,-2], which represents a quotient polynomial of $3x^3-5x^2+6x-2$, the remainder vector contains zeros.

Polynomial Functions

- **Roots of Polynomial:**

The solution of many engineering problems involve finding the roots of an equation of the form

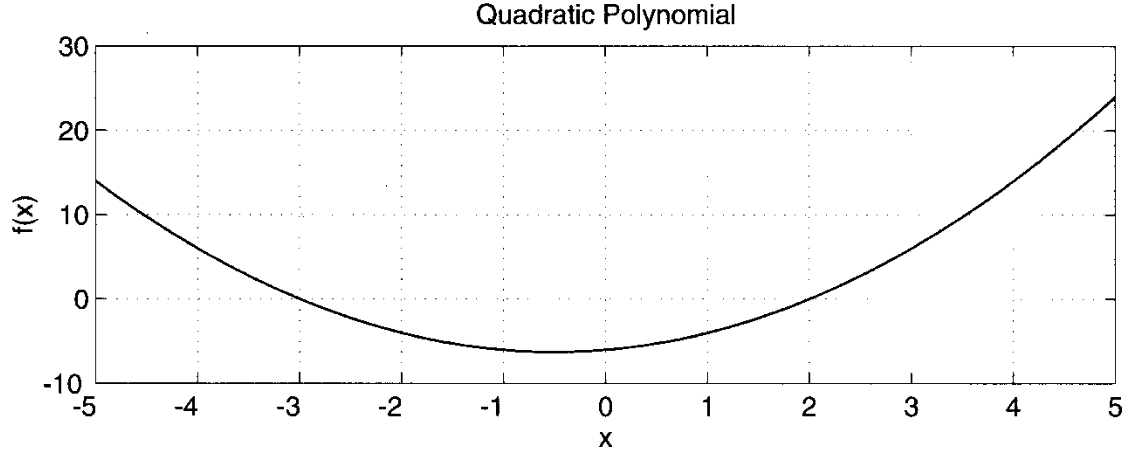
$$y = f(x)$$

Where the roots are the values of x for which y is equal to 0.

$$f(x) = x^2 + x - 6$$

$$= (x-2)(x+3)$$

Roots of this polynomial are 2 and -3



Polynomial with two real roots

Polynomial Functions

- **Cubic polynomial:**

$$f(x) = a_0x^3 + a_1x^2 + a_2x + a_3$$

3 real distinct roots

3 real multiple roots

1 distinct real root and 2 multiple real roots

1 real root and a complex conjugate pair of roots

Examples of functions:

$$\begin{aligned} f_1(x) &= (x-3)(x+1)(x-1) \\ &= x^3 - 3x^2 - x + 3 \end{aligned}$$

$$\begin{aligned} f_2(x) &= (x-2)^3 \\ &= x^3 - 6x^2 + 12x - 8 \end{aligned}$$

$$\begin{aligned} f_3(x) &= (x+4)(x-2)^2 \\ &= x^3 - 12x + 16 \end{aligned}$$

$$\begin{aligned} f_4(x) &= (x+2)(x-(2+i))(x-(2-i)) \\ &= x^3 - 2x^2 - 3x + 10 \end{aligned}$$

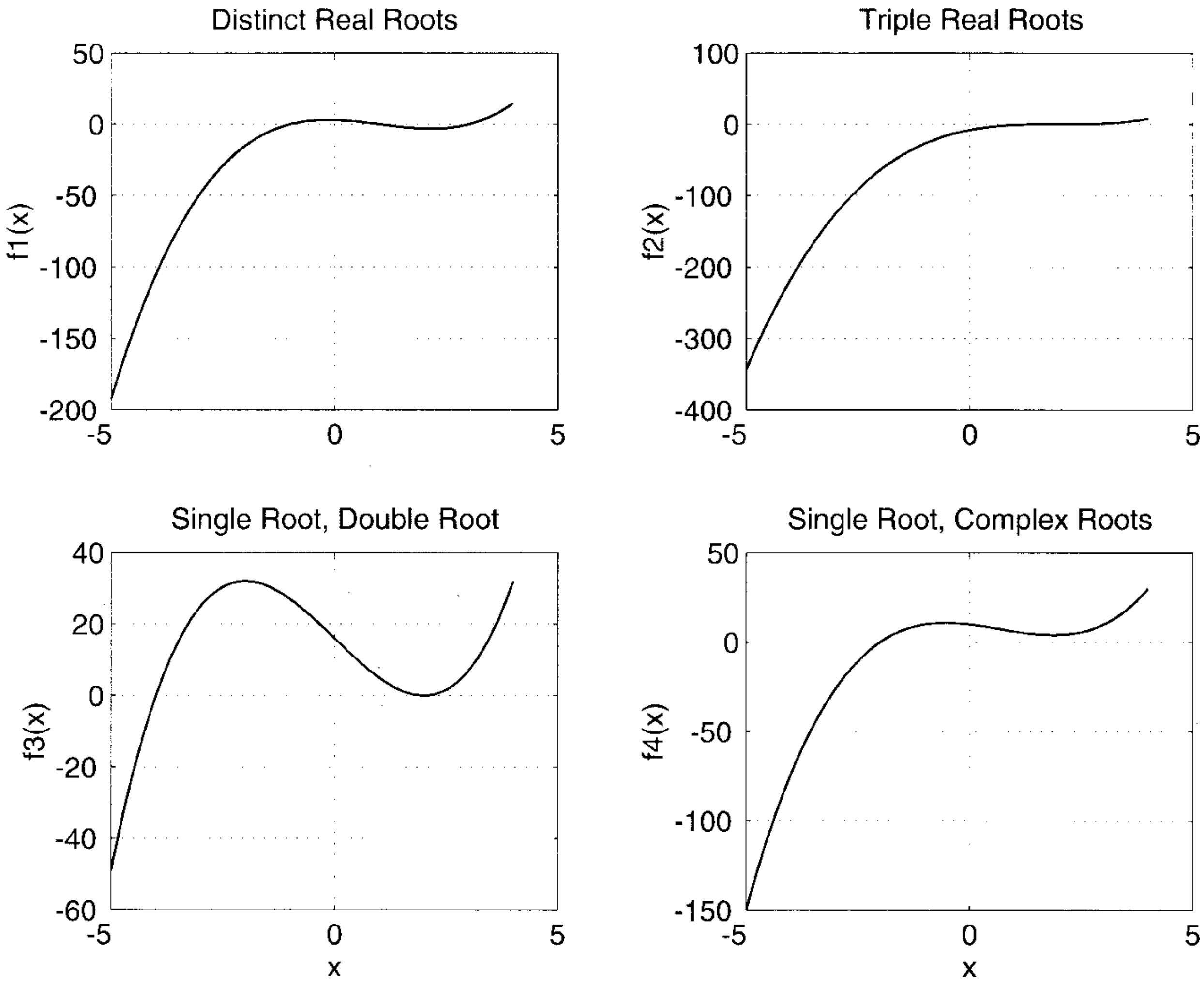
Polynomial Functions

$$f_1(x) = (x-3)(x+1)(x-1) = x^3 - 3x^2 - x + 3$$

$$f_2(x) = (x-2)^3 = x^3 - 6x^2 + 12x - 8$$

$$f_3(x) = (x+4)(x-2)^2 = x^3 - 12x + 16$$

$$f_4(x) = (x+2)(x-(2+i))(x-(2-i)) = x^3 - 2x^2 - 3x + 10$$



Cubic polynomials

Polynomial Functions

- **Roots of Polynomial:**

`roots(a)`	Determines the roots of the polynomial represented by the coefficient vector `a`.

Example: $f(x) = x^3 - 2x^2 - 3x + 10$

```
p = [1,-2,-3,10];
r = roots(p)
```

```
r = roots([1,-2,-3,10])
```

```
polyval([1,-2,-3,10],r)
```

Value of the polynomial at the roots

`poly(r)`	Determines the coefficients of the polynomial whose roots are contained in the vector `r`.

```
a = poly([-1,1,3]);
```

The result

Outline

- Polynomial Functions
- **Solutions to Systems of Linear Equations**
- Optimization Problem
- Matlab Optimization Toolbox
- Design Problems

Solutions to Systems of Linear Equations

Consider the following system of three equations with three unknowns:

3x+2y-z=10

-x+3y+2z=5

x-y-z=-1

$$AX = B$$

$$A = \begin{bmatrix} 3 & 2 & -1 \\ -1 & 3 & 2 \\ 1 & -1 & -1 \end{bmatrix} \qquad X = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \qquad B = \begin{bmatrix} 10 \\ 5 \\ -1 \end{bmatrix}$$

Solutions to Systems of Linear Equations

Consider the following system of three equations with three unknowns:

3x + 2y – x = 10

-x + 3y + 2z = 5

x - y - z = -1

$$AC = B$$

A=[3 , 2 , -1 ; -1 , 3 , 2 ; 1 , -1 , -1];

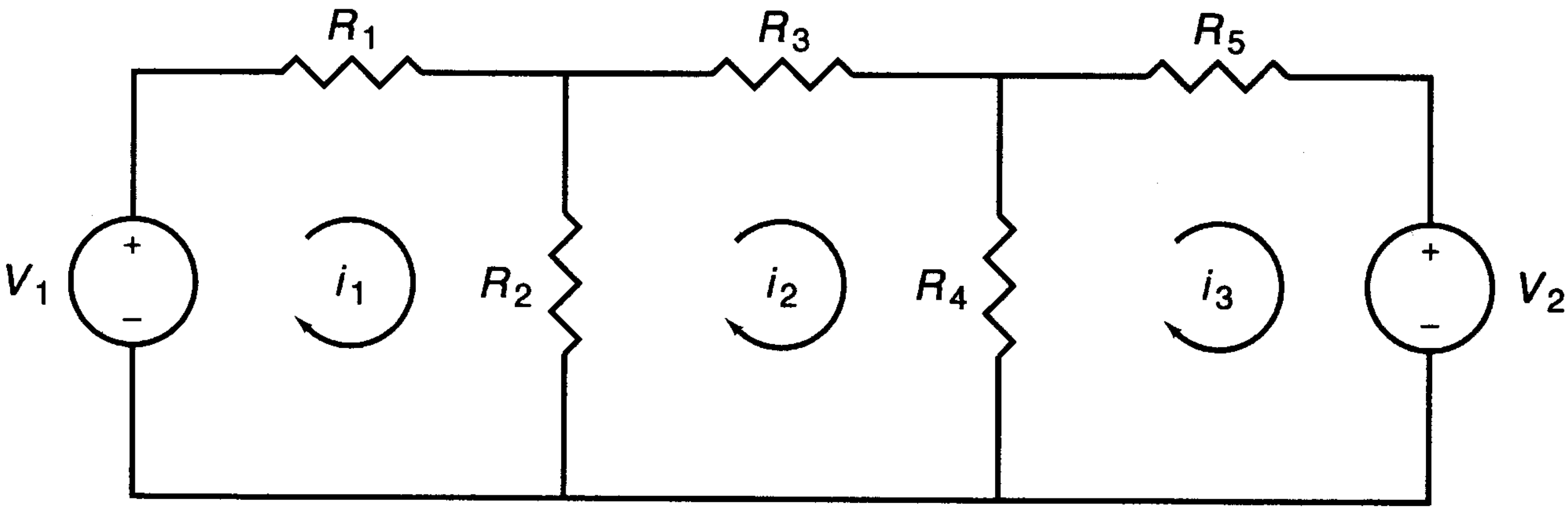
B=[10 ; 5 ; -1];

C=A\B;

Try this: **C=B/A;**

And try this also: **C=inv(A)*B;**

Example: Electrical Circuit Analysis



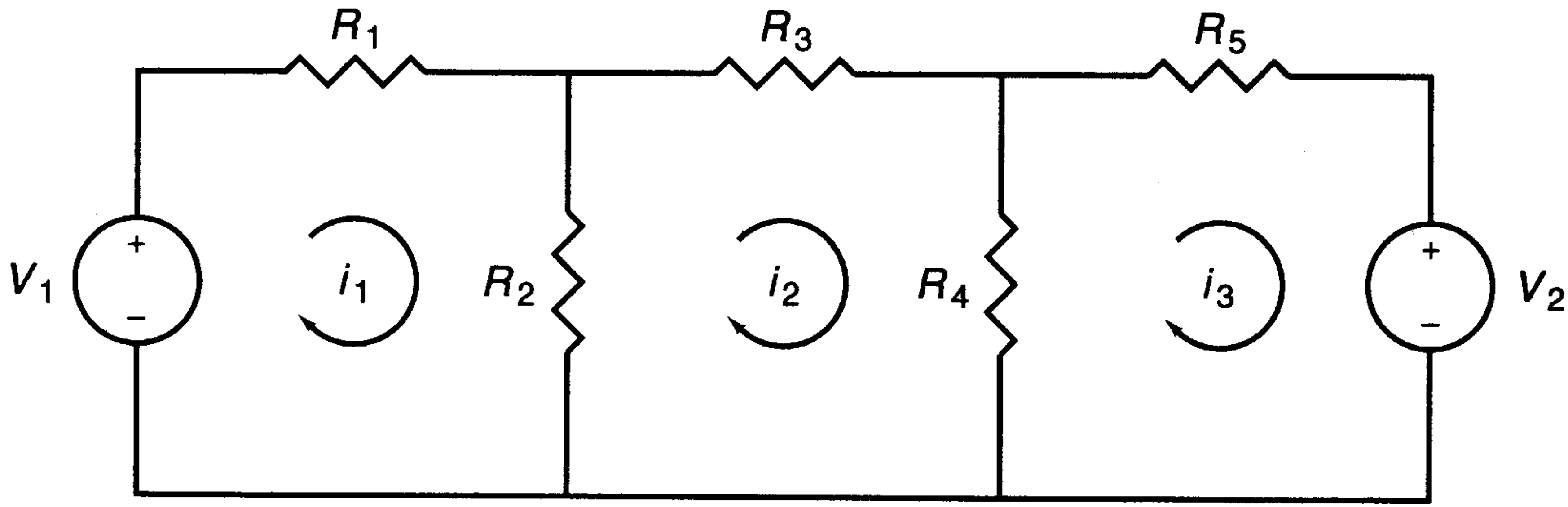
$$(R_1+R_2)i_1-R_2i_2+0i_3=V_1$$

$$-R_2i_1+(R_2+R_3+R_4)i_2-R_4i_3=0$$

$$0i_1-R_4i_2+(R_4+R_5)i_3=-V_2$$

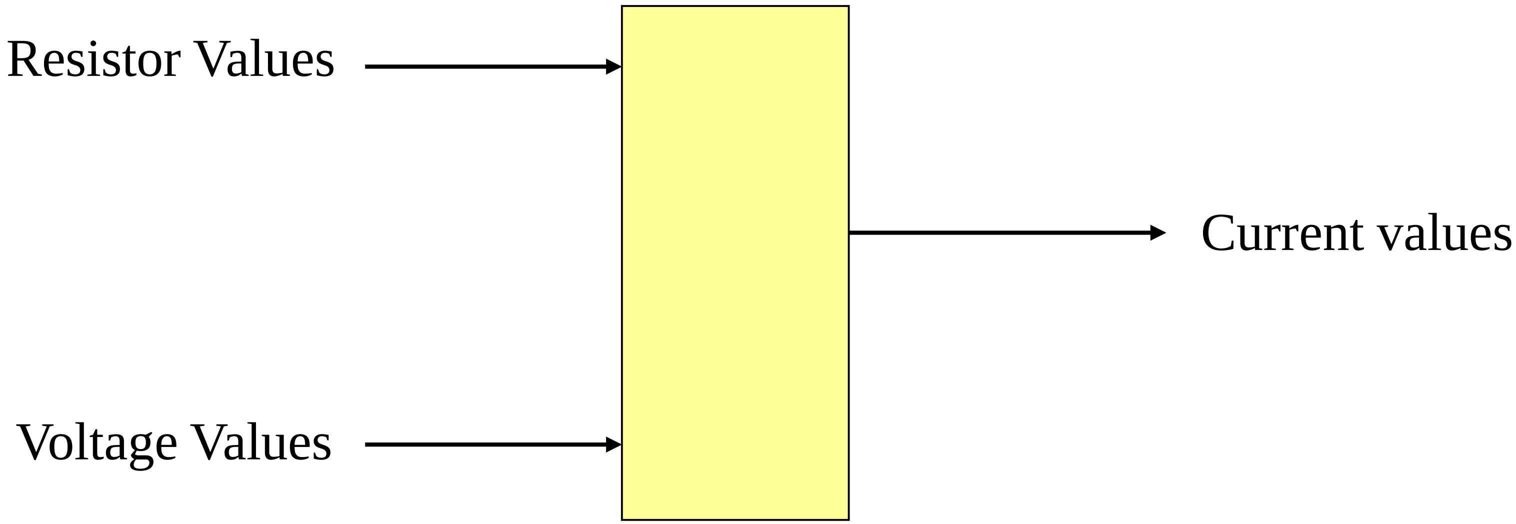
Example: Electrical Circuit Analysis

Using input values for the resistors and voltage sources, compute the three mesh currents in the circuit shown in the figure.



Example: Electrical Circuit Analysis

The following I/O diagram shows the five resistor value inputs and two voltage source inputs to the program. The output consists of the three mesh currents.



Example: Electrical Circuit Analysis

For a hand example, we use the following values:

$R_1=R_2=R_3=R_4=R_5=1$ ohm

$V_1=V_2=5$ volts.

The corresponding set of equations is then

$$2i_1 - i_2 + 0i_3 = 5$$
$$-i_1 + 3i_2 - i_3 = 0$$
$$0i_1 - i_2 + 2i_3 = -5$$

We use MATLAB to compute the solution

```
A=[2,-1,0;-1,3,-1;0,-1,2];
B=[5;,0;-5];
X=A\B;
ERR=sum(A*X-B)
```

This gives:

X=

2.5000

0

-2.5000

ERR=

0

Example: Electrical Circuit Analysis

```
%This program is for electrical circuit analysis
%Read resistor and voltage values
R=input('Enter resistor values in ohms, [R1…R5]');
V=input('Enter voltage values in volts, [V1 V2]');
%Initialize matrix A and vector B using AX=B form
A=[R(1)+R(2),-R(2), 0;-R(2), R(2)+R(3)+R(4),-R(4); 0,-R(4), R(4)+R(5)];
B=[V(1);0;-V(2)];
if rank(A)==3
    fprintf('Mesh Currents \n');
    i=A\B;
else
    fprintf('No Unique Solution');
end
```

Example: Electrical Circuit Analysis

```
Enter resistor values in ohms,  [R1...R5]  [1 1 1 1 1]
Enter voltage values in volts, [V1 V2]   [5 5]
Mesh Currents
i=
           2.5000
              0
          -2.5000
```

```
Enter resistor values in ohms,  [R1...R5]  [2 8 6 6 4]
Enter voltage values in volts, [V1 V2]   [40 20]
Mesh Currents
i=
           5.6000
           2.0000
          -0.8000
```

Outline

- Polynomial Functions
- Solutions to Systems of Linear Equations
- **Optimization Problem**
- Matlab Optimization Toolbox
- Design Problems

Optimization Problem

- **Statement of an Optimization Problem**

  An optimization or a mathematical programming problem can be stated as follows.

$$\text{Find}\quad \mathbf{X} = \begin{Bmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{Bmatrix} \quad \text{which minimize / maximize} \quad f(\mathbf{X})$$

subject to the constraints

$$g_j(\mathbf{X}) \le 0, \qquad j = 1,2,\ldots,m \qquad \text{Inequality constraints}$$

$$l_j(\mathbf{X}) = 0, \qquad j = 1,2,\ldots,p \qquad \text{Equality constraints}$$

Optimization Problem

Optimization Problem

Design vector (Set of unknowns or decision variables)

$$\mathbf{X} = \{x_1 \quad x_2 \quad . \quad . \quad . \quad x_n\}^T$$

+

Objective Function

$$f(\mathbf{X})$$

+

Set of inequality and/or equality constraints

$$g_j(\mathbf{X}),\ l_j(\mathbf{X})$$

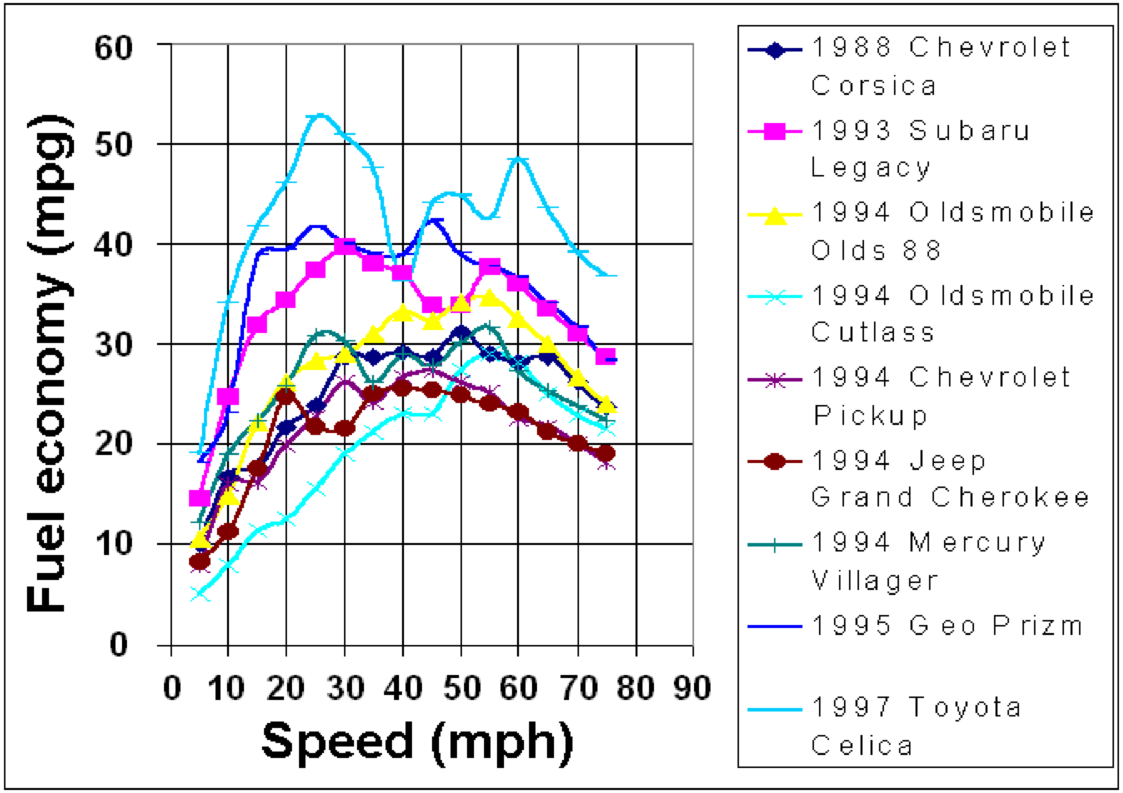
Optimization Problem

- ***Example-1:***

◇ **objective function**

f=fuel economy

distance traveled per unit of fuel used; in miles per gallon (mpg) or kilometres per litre (km/L),



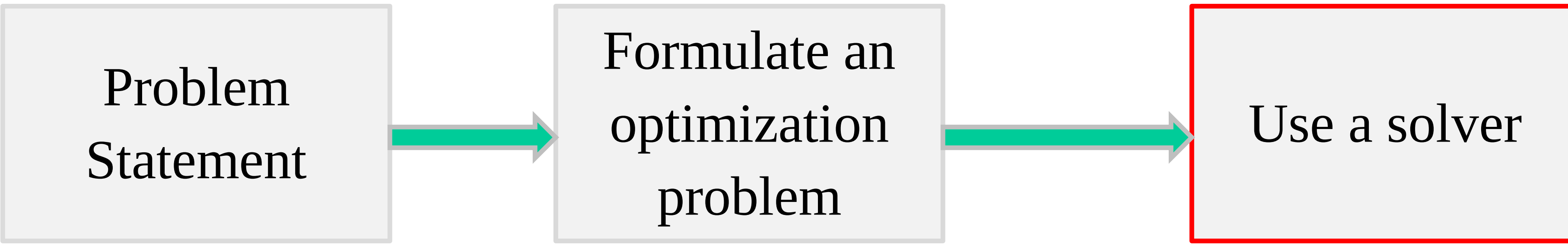
◇ **Design vector or unknown**

$$\mathbf{X} = \{x_1\} = \{\text{car speed in mph or Km/h}\}$$

◇ **Constraints**

$$g_1(\mathbf{X}) = x_1 \leq \text{speed limit}$$

Optimization Problem



Outline

- Polynomial Functions
- Solutions to Systems of Linear Equations
- Optimization Problem
- **Matlab Optimization Toolbox**
- Design Problems

Matlab Optimization Toolbox

- **What Is the Optimization Toolbox?**
  - Collection of functions that extends Matlab capabilities
  - Includes functions to solve difficult optimization problems
    - Minimization and Maximization Problems
    - Multi-variable objective function optimization
    - Linear and Quadratic optimization
    - Nonlinear system of equation solving
    - Nonlinear least squares and curve-fitting

Matlab Optimization Toolbox

- **Minimization**
  - Unconstrained minimization function $\min_x f(x)$
    - fminunc (fun,X_0)
  - Constrained minimization function $\min_x f(x)$ such that

$$\begin{cases} c(x) \leq 0 & \text{Non-linear inequality} \\ ceq(x) = 0 & \text{Non-linear equality} \\ A \cdot x \leq b & \text{Linear inequality} \\ Aeq \cdot x = beq & \text{Linear equality} \\ lb \leq x \leq ub & \text{Bounds} \end{cases}$$

    - fmincon(fun, X_0, A, b, Aeq, Beq, lb, ub, nonlcon)

Matlab Optimization Toolbox

- **Equation Solving**
  - Nonlinear equation solving
    - fslove(fun,X_0)
- **Curve Fitting**
  - Nonlinear curve fitting
    - lsqcurvefit(fun,x0,xdata,ydata)

Matlab Optimization Toolbox

- Examples using the Matlab Optimization Toolbox
  - Unconstrained Example
  - Constrained Example
  - Constrained With Bounds Example

Matlab Optimization Toolbox

- **Unconstrained Example**

```
function unconstrainedOptimizationExample

%clearing all previous data
clc,clear,close all

% Make a starting guess at the solution
x0 = [0,0];

% Overwritting the default optimization options
options = optimset('Display','none','LargeScale','off');

% Calling the constrained minmization function
[x,fval,exitflag,output] = fminunc(@objfun,x0,options)

%Defining the objective function
function f = objfun(x)
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
```

$$\min_{x} e^{x_1}[4\,x_1^2 + 2\,x_2^2 + 4\,x_1 x_2 + 2x_2 + 1]$$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

Matlab Optimization Toolbox

- **Unconstrained Example**

```
function unconstrainedOptimizationExample

%clearing all previous data
clc,clear,close all

% Make a starting guess at the solution
x0 = [0,0];

% Overwritting the default optimization options
options = optimset('Display','none','LargeScale','off');

% Calling the constrained minmization function
[x,fval,exitflag,output] = fminunc(@objfun,x0,options)

%Defining the objective function
function f = objfun(x)
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
```

```
x =

    0.5000   -1.0000


fval =

  1.8304e-015


exitflag =

     1


output =

       iterations: 7
        funcCount: 33
         stepsize: 1
    firstorderopt: 2.4568e-008
        algorithm: 'medium-scale: Quasi-Newton line search'
```

Matlab Optimization Toolbox

- **Constrained Example**

```
function constrainedOptimizationExample

%clearing all previous data
clc,clear,close all

% Make a starting guess at the solution
x0 = [-1,1];

% Overwritting the default optimization options
options = optimset('Display','none','LargeScale','off');
% Calling the constrained minmization function
[x, fval] = fmincon(@objfun,x0,[],[],[],[],[],[],@confun,options)
confun(x)

% Definig the set of Non-linear inequalities and equalities constraints
function [c, ceq] = confun(x)
% Nonlinear inequality constraints
c = [1.5 + x(1)*x(2) - x(1) - x(2);
-x(1)*x(2) - 10];
% Nonlinear equality constraints
ceq = [];

%Defining the objective function
function f = objfun(x)
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
```

$$\min_{x} e^{x_1}[4\,x_1^2 + 2\,x_2^2 + 4\,x_1 x_2 + 2x_2 + 1]$$

$$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

<u>Such that</u>

$$\left.\begin{aligned} 1.5\,x_1 x_2 - x_1 - x_2 &\le 0 \\ -x_1 \times x_2 &\le 10 \end{aligned}\right\} c(x) \le 0$$

Matlab Optimization Toolbox

- **Constrained Example**

```
function constrainedOptimizationExample

%clearing all previous data
clc,clear,close all

% Make a starting guess at the solution
x0 = [-1,1];

% Overwritting the default optimization options
options = optimset('Display','none','LargeScale','off');
% Calling the constrained minmization function
[x, fval] = fmincon(@objfun,x0,[],[],[],[],[],[],@confun,options)
confun(x)

% Definig the set of Non-linear inequalities and equalities constraints
function [c, ceq] = confun(x)
% Nonlinear inequality constraints
c = [1.5 + x(1)*x(2) - x(1) - x(2);
-x(1)*x(2) - 10];
% Nonlinear equality constraints
ceq = [];

%Defining the objective function
function f = objfun(x)
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
```

```
x =

   -9.5474    1.0474


fval =

    0.0236


exitflag =

     1


output =

       iterations: 8
        funcCount: 28
     lssteplength: 1
         stepsize: 4.2503e-004
        algorithm: 'medium-scale: SQP, Quasi-Newton, line-search'
    firstorderopt: 8.5125e-007
```

Matlab Optimization Toolbox

- **Constrained With Bounds Example**

```
function constrainedWithBounds

%clearing all previous data
clc,clear,close all

lb = [0,0]; % Set lower bounds
ub = [ ]; % No upper bounds
x0 = [-1,1]; % Initial guess

% Overwritting the default optimization options
options = optimset('Display','none','LargeScale','off');
% Calling the constrained minmization function
[x, fval] = fmincon(@objfun,x0,[],[],[],[],lb,ub,@confun,options)


% Definig the set of Non-linear inequalities and equalities constraints
function [c, ceq] = confun(x)
% Nonlinear inequality constraints
c = [1.5 + x(1)*x(2) - x(1) - x(2);
-x(1)*x(2) - 10];
% Nonlinear equality constraints
ceq = x(1)^2 + x(2) - 1;

%Defining the objective function
function f = objfun(x)
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
```

$$\min_{x} e^{x_1}[4\,x_1^2 + 2\,x_2^2 + 4\,x_1 x_2 + 2x_2 + 1]$$

Such that

$$\left.\begin{array}{l} 1.5\,x_1 \times x_2 \ - x_1 \ - x_2 \leq 0 \\ -x_1 \times x_2 \ \leq 10 \end{array}\right\} c(x) \leq 0$$

$$\left.x_1^2 + x_2 - 1 = 0\right\} ceq(x) \leq 0$$

$$\left.\begin{array}{l} x_1 \leq 0 \\ x_2 \leq 0 \end{array}\right\} lb \leq x$$

Matlab Optimization Toolbox

- **Constrained With Bounds Example**

```
function constrainedWithBounds

%clearing all previous data
clc,clear,close all

lb = [0,0]; % Set lower bounds
ub = [ ]; % No upper bounds
x0 = [-1,1]; % Initial guess

% Overwritting the default optimization options
options = optimset('Display','none','LargeScale','off');
% Calling the constrained minmization function
[x, fval] = fmincon(@objfun,x0,[],[],[],[],lb,ub,@confun,options)


% Definig the set of Non-linear inequalities and equalities constraints
function [c, ceq] = confun(x)
% Nonlinear inequality constraints
c = [1.5 + x(1)*x(2) - x(1) - x(2);
-x(1)*x(2) - 10];
% Nonlinear equality constraints
ceq = x(1)^2 + x(2) - 1;

%Defining the objective function
function f = objfun(x)
f = exp(x(1))*(4*x(1)^2+2*x(2)^2+4*x(1)*x(2)+2*x(2)+1);
```

```
x =

   -0.1429    1.2563


fval =

    5.2296


exitflag =

     0


output =

       iterations: 41
        funcCount: 201
     lssteplength: 0.0039
         stepsize: 0.0300
        algorithm: 'medium-scale: SQP, Quasi-Newton, line-search'
    firstorderopt: 723.1771
```

Outline

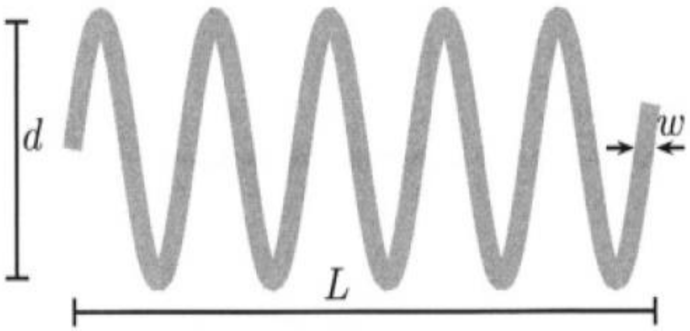
- Polynomial Functions
- Solutions to Systems of Linear Equations
- Optimization Problem
- Matlab Optimization Toolbox
- **Design Problems**

Design Problem

- **Spring Design**

A very simple design problem in engineering is to design a spring under tension and/or compression for given requirements or constraints including **minimum deflection**, **outer diameter**, **frequency**, and **maximum shear stress**.

For a given problem such as forming a spring using a wire, the adjustable parameters or design variables are the **wire diameter** w (thickness), the **coil diameter** d, and the **length** L (or equivalently, the number of active coils).



[1]

Design Problem

- **Spring Design**

This design optimization problem can be formulated as follows:

$$\text{minimize} \quad f(\boldsymbol{x}) = (2+L)dw^2,$$

$$\text{subject to} \quad g_1(\boldsymbol{x}) = 1 - \frac{d^2 L}{7178 w^4} \le 0$$

$$g_2(\boldsymbol{x}) = \frac{4d^2 - wd}{12566 dw^3 - w^4} + \frac{1}{5108 w^2} - 1 \le 0$$

$$g_3(\boldsymbol{x}) = 1 - \frac{140.45 w}{d^2 L} \le 0$$

$$g_4(\boldsymbol{x}) = \frac{w+L}{1.5} - 1 \le 0$$

The simple bounds are

$$0.05 \le w \le 2.0, \qquad 0.25 \le d \le 1.3, \qquad 2.0 \le L \le 15.0$$

If we use the Matlab program, we can find the following best solution:

$$f_* \approx 0.0075$$

$$\boldsymbol{x}_* \approx (0.0500,\ 0.2500,\ 9.9877)$$

[1]

Design Problem

- **Spring Design**

$$\mathbf{X}=\begin{Bmatrix} w \\ d \\ L \end{Bmatrix}=\begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} \qquad f(\mathbf{X})=(2+x_3)x_2x_1^2$$

$$g_1(\mathbf{X})=1-\frac{x_2^3x_3}{7178\,x_1^4}\le 0$$

$$g_2(\mathbf{X})=\frac{4x_2^2-x_1x_2}{12566\,x_2x_1^3-x_1^4}+\frac{1}{5108\,x_1^2}-1\le 0$$

$$g_3(\mathbf{X})=1-\frac{140.45\,x_1}{x_2^2x_3}\le 0$$

$$g_4(\mathbf{X})=\frac{x_1+x_2}{1.5}-1\le 0$$

$$\text{find} \quad \mathbf{X}^*=\arg\min_{\mathbf{X}} \; f(\mathbf{X})$$

$$\text{s.t.} \quad g_i(\mathbf{X}) \qquad i=1,2,3,4$$

Design Problem

- **Spring Design**

The bounds can be written as $Lb \le x \le Ub$

with the lower bound $Lb = [0.05; 0.25; 2.0]$

and the upper bound $Ub = [2.0; 1.3; 15.0]$

Starting from an educated guess $x0 = [0.1; 0.5; 10]$

Design Problem

- **Spring Design**

```
% Spring Design Optimization using Matlab fmincon
function spring
x0=[0.1; 0.5; 10];
Lb=[0.05; 0.25; 2.0]; Ub=[2.0; 1.3; 15.0];

% call matlab optimization toolbox
[x,fval]=fmincon(@objfun,x0, [],[],[], [] ,Lb,Ub,@nonfun)

% Objective function
function f=objfun(x)
f=(2+x(3))*x(1)^2*x(2);

% Nonlinear constraints
function [g,geq]=nonfun(x)

% Inequality constraints
g(1)=1-x(2)^3*x(3)/(7178*x(1)^4);
gtmp=(4*x(2)^2-x(1)*x(2))/(12566*x(2)*x(1)^3-x(1)^4);
g(2)=gtmp+1/(5108*x(1)^2)-1;
g(3)=1-140.45*x(1)/(x(2)^2*x(3));
g(4)=x(1)+x(2)-1.5;

% Equality constraints [none]
geq= [] ;
```

Run

```
x =
    0.0500
    0.2845
    2.0000
fval =
    0.0028
```

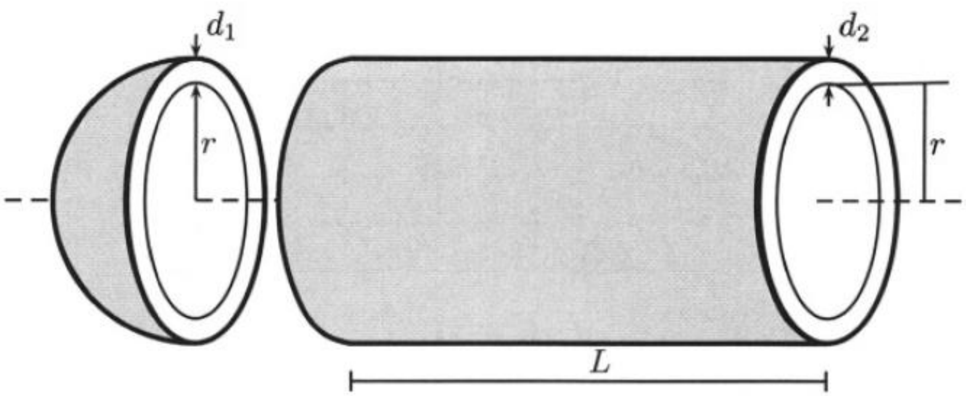
Design Problem

- **Pressure Vessels**

Pressure vessels are literarily everywhere such as bottles of sparkling drink, and gas tanks. For a given volume and working pressure, the basic aim of designing a cylindrical vessel is to **minimize the total cost**.

Typically, the design variables are:

◇ the thickness d_1 of the head,
◇ the thickness d_2 of the body,
◇ the inner radius r, and
◇ the length L of the cylindrical section.



Design Problem

- **Pressure Vessels**

This is a well-known test problem for and it can be written as

$$\text{minimize} \quad f(\boldsymbol{x}) = 0.6224 d_1 r L + 1.7781 d_2 r^2 + 3.1661 d_1^2 L + 19.84 d_1^2 r,$$

$$\begin{aligned} \text{subject to} \quad & g_1(\boldsymbol{x}) = -d_1 + 0.0193 r \leq 0 \\ & g_2(\boldsymbol{x}) = -d_2 + 0.00954 r \leq 0 \\ & g_3(\boldsymbol{x}) = -\pi r^2 L - \frac{4\pi}{3} r^3 + 1296000 \leq 0 \\ & g_4(\boldsymbol{x}) = L - 240 \leq 0. \end{aligned}$$

The simple bounds are $0.0625 \leq d_1, d_2 \leq 99 \times 0.0625,$

$$10.0 \leq r, \qquad L \leq 200.0.$$

It is worth pointing out that d_1 and d_2 should only take discrete values of integer multiples of 0.0625.

Solution: $f_* = 6059.714,$

$$\boldsymbol{x}_* = (0.8125,\ 0.4375,\ 42.0984,\ 176.6366)$$

Design Problem

- **Pressure Vessels**

$$\mathbf{X} = (d_1 \quad d_2 \quad r \quad L)^T = (x_1 \quad x_2 \quad x_3 \quad x_4)^T$$

minimize

$$f(\mathbf{X}) = 0.6224\, x_1 x_3 x_4 + 1.7781\, x_2 x_3^2 + 3.1661\, x_1^2 x_4 + 19.84\, x_1^2 x_3$$

subject to

$$g_1(\mathbf{X}) = -x_1 + 0.0193\, x_3 \le 0$$

$$g_2(\mathbf{X}) = -x_2 + 0.00954\, x_3 \le 0$$

$$g_3(\mathbf{X}) = x_4 - 240 \le 0$$

$$g_4(\mathbf{X}) = -\pi x_3^2 x_4 - \frac{4\pi}{3} x_3^3 + 1296000 \le 0$$

Design Problem

- **Pressure Vessels**

The first three inequalities g_1, g_2, g_3 *can be written as*

$$\mathbf{A}x \le b$$

$$\mathbf{A} = \begin{pmatrix} -1 & 0 & 0.0193 & 0 \\ 0 & -1 & 0.00954 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad b = \begin{pmatrix} 0 \\ 0 \\ 240 \end{pmatrix}$$

The simple bounds can be rewritten as

$$Lb = [d; d; 10; 10];\ Ub = [99 \times d;\ 99 \times d;\ 200;\ 200],$$

where $d = 0.0625$.

Design Problem

- **Pressure Vessels**

```
% Design Optimization of a Pressure Vessel
function pressure_vessel
x0=[1; 1; 20; 50];
d=0.0625;

% Linear inequality constraints
A=[-1 0 0.0193 0;0 -1 0.00954 0;0 0 0 1];
b=[0; 0; 240];

% Simple bounds
Lb=[d; d; 10; 10]; Ub=[99*d; 99*d; 200; 200];
options=optimset('Display','iter','TolFun',1e-08);
[x,fval]=fmincon(@objfun,x0,A,b, [] , [] ,Lb,Ub,@nonfun,options)

% The objective function
function f=objfun(x)
f=0.6224*x(1)*x(3)*x(4)+1.7781*x(2)*x(3)^2+3.1661*x(1)^2*x(4)+19.84*x(1)^2*x(3)

% Nonlinear constraints
function [g,geq]=nonfun(x)

% Nonlinear inequality
g=-pi*x(3)^2*x(4)-4*pi/3*x(3)^3+1296000;

% Equality constraint [none]
geq= [];
```

Run

```
x =
    0.7782
    0.3846
   40.3196
  200.0000
fval =
  5885.3328
```

Conclusion

- Matlab is a high-level and interactive environment used by millions of engineers and scientists worldwide.
- It enables rapid development of prototypes.
- Mathematical optimization problems have three components.
    - Decision variables.
    - Objective function.
    - Constraints.
- Matlab optimization toolbox can be used to solve these problems.