Lecture 18

Polynomial

- Polynomials are a special type of nonlinear algebraic function of the general form

$$f_n(x) = a_1x^n + a_2x^{n-1} + \cdots + a_{n-1}x^2 + a_nx + a_{n+1}$$

  where n is the order of the polynomial, and the a's are constant coefficients. In many (but not all) cases, the coefficients will be real. For such cases, the roots can be real and/or complex.

- We generally do NOT write the polynomial as

$$f_n(x) = a_nx^n + a_{n-1}x^{n-1} + \cdots + a_2x^2 + a_1x + a_0$$

  because the coefficient of x^n is generally the first element in the coefficient vector, and coefficient of x^{n-1} the second …

- In general, an n th order polynomial will has n roots, and has $n + 1$ coefficients.

  Example: ax2 + bx +c =2 has two roots

Polynomial

- If we want to find a particular root of a polynomial, the bracketing techniques and open methods can be used.
- MATLAB has a built-in program called roots to determine all the roots of a polynomial – including real and imaginary components.
- The syntax of the roots function

  x = roots(c)

  x is a column vector containing the roots

  c is a row vector containing the polynomial coefficients (in the new version of MATLAB, c does not have to be a row vector). The sequence of the coefficients is from the coefficient of x^n to $x^0 = 1$

  Example: $f_n(x) = a_1 x^n + a_2 x^{n-1} + \cdots + a_{n-1} x^2 + a_n x + a_{n+1}$

  $$c = (a_1 \quad a_2 \quad \cdots \quad a_n \quad a_{n+1})$$

Polynomial

- Example 1: Find the roots of

  $f(x) = x^5 - 3.5x^4 + 2.75x^3 + 2.125x^2 - 3.875x + 1.25$

  (the vector of coefficients is [1 -3.5 2.75 2.125 -3.875 1.25])

  ```
  >> x = roots([1  -3.5  2.75  2.125  -3.875  1.25])
  ```

  or

  ```
  >> c = [1  -3.5  2.75  2.125  -3.875  1.25];
  >> x = roots(c)
  x =
     2.0000 + 0.0000i
    -1.0000 + 0.0000i
     1.0000 + 0.5000i
     1.0000 - 0.5000i
     0.5000 + 0.0000i
  ```

  The third and fourth roots are complex

Polynomial

- Example 2: Find the roots of $f(x) = x^2 + 4x + 2$

```
>> c = [1  4  2];
>> xr = roots(c)     % or   xr = roots([1 4 2])
>> xr =
       -3.4142
       -0.5858
```

When c is a column vector, the roots function still works

```
>> c = [1;  4;  2];
>> xr = roots(c);
       -3.4142
       -0.5858
```

‘c’ cannot be a matrix

Polynomial

- roots function has an inverse function called poly
- MATLAB's poly function can be used to determine polynomial coefficients if roots are given.

Syntax: c = poly(x), where x is the vector of roots, and c is the vector of coefficients of a polynomial

Example 1 : >> xr =

-3.4142

-0.5858

>> c = poly(xr)

>> c =

1 4 2

The polynomial is: $f(x) = x^2 + 4x + 2$

when xr is a row vector, poly function still works

Can we use c = poly(-3.4142 -0.5858)?

Example 2 : >> b=poly([0.5 -1])

>> b=

1.0000 0.5000 -0.5000

The polynomial is: $f(x) = x^2 + 0.5x - 0.5$

Polynomial

- MATLAB's polyval function can be used to evaluate a polynomial at one or more points:

  b = polyval(c,x)

  'c' can be either a row or a column vector

  where c is the vector of the coefficients of the polynomial, x is the x values of the points.

  Example: : $f(x) = x^2 + 4x + 2$

```
>> c =  [1   4   2];
>> xr = roots(c)
>> xr =
    -3.4142
    -0.5858
>> b = polyval(c,xr(1))
b =
  -4.4409e-16
```

Polynomial

- MATLAB's polyval function can be used to evaluate a polynomial at one or more points:

Example: : $f(x) = x^2 + 4x + 2$

```
>> c =  [1   4   2];
>> a =  [1   2  3];
>> b = polyval(c,a)
b =
   7   14   23
```

```
>> c =  [1   4   2];
>> a =  [1;  2; 3];
>> b = polyval(c,a)
b =
     7
    14
    23
```

Is there any other way to find the values of a function? Define a function handle like f=@(x) x^2 + …
f(a)

Polynomial

- MATLAB's polyfit :

  MATLAB has a built-in function polyfit that fits a least-squares *n*th-order polynomial to data. The syntax is

  p = polyfit(x,y,N)

  where x and y are the vectors of the independent and dependent variables, respectively, N is the order of the polynomial.

  polyfit finds the coefficients of a polynomial p(x) of degree N that fits the y-data best in a least-squares sense.

  Example:

```
>> x = [10  20  30  40  50  60  70  80];
>> y = [25  70  380  550  610  1220  830  1450];
>> a = polyfit(x,y,1)
a =
          19.4702 -234.2857
>> xx = linspace(10,80,10);
>> plot(x,y,'bo',xx,polyval(a,xx))
```

First order polynomial is a straight line

Multiple Roots

- **Homework on Canvas**