Text reference page 184.

Lagrange Interpolation

Purpose
To use MATLAB to find and plot the Lagrange interpolating polynomial and study how well it serves as an approximation to a function.

MATLAB Functions
`plot, polyval, A\b, polyfit, max, abs`

Suppose we have data points $(x_1, y_1), (x_2, y_2), \cdots (x_n, y_n)$ and we would like to find the *Lagrange interpolating polynomial*

$$p(t) = c_0 + c_1 t + c_2 t^2 + \cdots + c_{n-1} t^{n-1}$$

that passes through these points, that is,

$$\begin{aligned} p(x_1) &= y_1 \\ p(x_2) &= y_2 \\ &\vdots \\ p(x_n) &= y_n \end{aligned}$$

These equations become the linear system

$$\begin{aligned} c_0 + c_2 x_1^1 + c_2 x_1^2 + \cdots + c_{n-1} x_1^{n-1} &= y_1 \\ c_0 + c_1 x_2^1 + c_2 x_2^2 + \cdots + c_{n-1} x_2^{n-1} &= y_2 \\ &\vdots \\ c_0 + c_1 x_n^1 + c_2 x_n^2 + \cdots + c_{n-1} x_n^{n-1} &= y_n \end{aligned}$$

This system gives rise to the matrix equation (see Exercise 11 on page 184 of the text) $A\mathbf{c} = \mathbf{y}$, where

$$A = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \cdots & x_2^{n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{n-1} \end{bmatrix} \qquad \mathbf{c} = \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \end{bmatrix} \qquad \mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

Notice how the coefficient matrix depends only on the x-coordinates $x_1, x_2, \ldots, x_n$. In MATLAB we let

```
x = (-1:2)', A = [ones(4,1), x, x.^ 2, x.^ 3];
```

Next, we set up the y-coordinates

```
y = [1; 1; -1; 0];
```

The data should be in the form of 4×1 column vectors. Now we can use MATLAB to solve for $\mathbf{c}$:

```
c = A\y
```

The entries of the vector $\mathbf{c}$ are the coefficients of the polynomial $p(t) = c(1) + c(2)t + c(3)t^2 + c(4)t^3$. We can plot this vector together with the data points to see how the polynomial passes through them. First, we plot

```
plot(x,y,'o'), hold
```

The `hold` keeps this plot in place so that we can plot $p(t)$ over the data points. MATLAB represents polynomials in the reverse order from the way they are conventionally written (as above), so first we need to reverse the coefficients and set up the sample vectors for plotting:

```
c = c(4:-1:1)
xs = -1:.1:2;
ys = polyval(c,xs);
plot(xs,ys)
```

MATLAB Exercises

1. Solve Exercise 33 on page 27 of the text. Plot the data and the interpolating polynomial.

2. Solve Exercise 34 on page 27 of the text. Plot the data and the interpolating polynomial. You should see the polynomial pass through the data points. Use the MATLAB function `polyval` to approximate the force when the projectile is traveling 750 ft/sec by interpolating the value.

3. Try to approximate the function sin x with a cubic polynomial. First plot sin x:

```
xs = -pi:pi/10:pi;
ys = sin(xs);
plot(xs,ys,'*'), hold
```

Choose the points:

```
x = x(1:5:20)
y = sin(x)
```

Now find the Lagrange interpolating polynomial for the data points given by the vectors `x` and `y`. Plot the polynomial and the data to visually check the quality of the approximation.

The MATLAB function `c = polyfit(x,y,deg)` finds a polynomial of degree `deg` that approximates the data given by the vectors `x` and `y`. The vector `c` gives the coefficients of the polynomial in the order that MATLAB uses (which is the reverse of the usual order). Thus, to plot one of these polynomials on the interval $[a, b]$, you need only use this sequence of commands

```
c = polyfit(x,y);
xs = a:(b-a)/100:b;
ys = polyval(c,xs);
plot(xs,ys);
```

4. Using the data from Exercise 33 on page 27 of the text, first plot the data and then plot approximating polynomials of degrees 1 and 2. Find the Lagrange interpolating polynomial. How does it compare with the two approximating polynomials?

5. Using the data from Exercise 34 on page 27 of the text, first plot the data and then plot each of the approximating polynomials of degrees 1, 2, 3, 4 and 5. Describe what happens. Find the Lagrange interpolating polynomial. How does the interpolating polynomial compare with the approximating polynomials? Plot the approximating polynomials of degrees 1, 2, and 3 on the interval [0, 20] on a single graph. Now plot the approximating polynomials of degrees 4 and 5 on that graph. If you were going to *extrapolate* a value for the wind tunnel at 1500 ft/sec, would you use the Lagrange interpolating polynomial or one of the approximating polynomials? Explain your answer.

6. Althought an interpolating polynomial does pass through the given data points, it may not be a very good approximating tool. Plot the function

$$f(x) = \frac{1}{x^2 + 1}$$

in MATLAB:

```
xs = -10:.1:10;
ys = 1./(xs.^2 + 1);
plot(xs,ys), hold
```

Now take a few data points from the function and plot them:

```
x = (-10:2:10)';
```

```
y = 1./(x.^2 + 1);
plot(x,y,'o')
```

Find the Lagrange interpolating polynomial for these data and plot it over the function. Describe on how well the Lagrange polynomial approximates the function $f(x)$. You can get a numerical gauge of the approximation by looking at the difference between the Lagrange polynomial and $f(x)$ on the plotting vector `xs`. Suppose that `c` is the vector of coefficients (as supplied by `polyfit`) for the interpolating polynomial. In MATLAB, let

```
yc = polyval(c,xs);
```

Now look at the difference

```
max(abs(ys - yc))
```

Use MATLAB to locate the coordinates of the plotting vector `xs` where this maximum occurs. Your answers should agree with what you see on the graph.