# MATH 3341: Introduction to Scientific Computing Lab

Melissa Butler

University of Wyoming

October 11, 2021

# Lab 08: MATLAB Interpolation Routines and Their Derivatives

# Polynomial Interpolation Routines

# polyfit and polyval

- `p = polyfit(xdata, ydata, n)`: finds the coefficients of a polynomial $p(x)$ of degree $n$, i.e., $p(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_{n+1}$, that fits the data `xdata`, `ydata` best in a least-squares sense. `p` is a row vector of length `n + 1` containing the polynomial coefficients in descending powers, `p` stores $[p_1, p_2, \ldots, p_n, p_{n+1}]$.

# polyfit and polyval

- `p = polyfit(xdata, ydata, n)`: finds the coefficients of a polynomial $p(x)$ of degree $n$, i.e., $p(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_{n+1}$, that fits the data `xdata`, `ydata` best in a least-squares sense. `p` is a row vector of length `n + 1` containing the polynomial coefficients in descending powers, `p` stores $[p_1, p_2, \ldots, p_n, p_{n+1}]$.
- `y = polyval(p, x)`: returns the value of a polynomial `p` evaluated at `x`: $y = p(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_{n+1}$.

# polyfit and polyval

- `p = polyfit(xdata, ydata, n)`: finds the coefficients of a polynomial $p(x)$ of degree $n$, i.e., $p(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_{n+1}$, that fits the data `xdata`, `ydata` best in a least-squares sense. `p` is a row vector of length `n + 1` containing the polynomial coefficients in descending powers, `p` stores $[p_1, p_2, \ldots, p_n, p_{n+1}]$.
- `y = polyval(p, x)`: returns the value of a polynomial `p` evaluated at `x`: $y = p(x) = p_1 x^n + p_2 x^{n-1} + \cdots + p_n x + p_{n+1}$.
- Example:

```
xdata = [-2, 0, 1]
ydata = [9, 1, 3]
p = polyfit(xdata, ydata, 2)  % p = [2, 0, 1]
y = polyval(p, 2)             % y = 9
```

  In other words, the fitted polynomial is $p(x) = 2x^2 + 0x + 1 = 2x^2 + 1$, and evaluate $p(x)$ at $x = 2$, we have $y = p(2) = 2 \times 2^2 + 1 = 9$.

# Piecewise Polynomial: `spline`, `pchip`, and `ppval`

- `pp = spline(xdata, ydata)`: Use cubic spline (piecewise cubic polynomial) to fit the data `xdata` and `ydata`. `pp` is a `struct` (structure) contains number of pieces of cubic polynomials (`pp.pieces`), coefficients matrix (`pp.coefs`) of which the `ith` row are the coeffcients for the `ith` piece cubic polynomial, break points (`pp.breaks`) which is a row vector contains the endpoints of the interval for each pieces.

# Piecewise Polynomial: `spline`, `pchip`, and `ppval`

- `pp = spline(xdata, ydata)`: Use cubic spline (piecewise cubic polynomial) to fit the data `xdata` and `ydata`. `pp` is a `struct` (structure) contains number of pieces of cubic polynomials (`pp.pieces`), coefficients matrix (`pp.coefs`) of which the `i`th row are the coeffcients for the `i`th piece cubic polynomial, break points (`pp.breaks`) which is a row vector contains the endpoints of the interval for each pieces.
- `pp = pchip(xdata, ydata)`: Use Piecewise Cubic Hermite Interpolating Polynomial to fit the data `xdata` and `ydata`. `pp` is same as above.

# Piecewise Polynomial: `spline`, `pchip`, and `ppval`

- `pp = spline(xdata, ydata)`: Use cubic spline (piecewise cubic polynomial) to fit the data `xdata` and `ydata`. `pp` is a `struct` (structure) contains number of pieces of cubic polynomials (`pp.pieces`), coefficients matrix (`pp.coefs`) of which the `i`th row are the coeffcients for the `i`th piece cubic polynomial, break points (`pp.breaks`) which is a row vector contains the endpoints of the interval for each pieces.
- `pp = pchip(xdata, ydata)`: Use Piecewise Cubic Hermite Interpolating Polynomial to fit the data `xdata` and `ydata`. `pp` is same as above.
- `y = ppval(pp, x)`: determines which intervals `x` lies on and then evaluate the corresponding cubic polynomial at `x`.

# Piecewise Polynomial: `spline`, `pchip`, and `ppval`

- `pp = spline(xdata, ydata)`: Use cubic spline (piecewise cubic polynomial) to fit the data `xdata` and `ydata`. `pp` is a `struct` (structure) contains number of pieces of cubic polynomials (`pp.pieces`), coefficients matrix (`pp.coefs`) of which the `i`th row are the coeffcients for the `i`th piece cubic polynomial, break points (`pp.breaks`) which is a row vector contains the endpoints of the interval for each pieces.
- `pp = pchip(xdata, ydata)`: Use Piecewise Cubic Hermite Interpolating Polynomial to fit the data `xdata` and `ydata`. `pp` is same as above.
- `y = ppval(pp, x)`: determines which intervals `x` lies on and then evaluate the corresponding cubic polynomial at `x`.
- `y = spline(xdata, ydata, x)`: is the same as `y = ppval(spline(xdata, ydata), x)`, thus providing, in `y`, the values of the interpolant at `x`.

# Piecewise Polynomial: spline, pchip, and ppval

Example:

```
xdata = [0 1 2 3]
ydata = [10 8 6 4]
pp = spline(xdata, ydata)
y = ppval(pp, 1.5)              % y = 7
y = spline(xdata, ydata, 1.5) % same as y = ppval(pp, 1.5)
```

# Derivatives of Interpolation Polynomials

# `polyder`: Differentiate polynomial

- `dp = polyder(p)`: returns the derivative of the polynomial whose coefficients are the elements of vector `p`.

# `polyder`: Differentiate polynomial

- `dp = polyder(p)`: returns the derivative of the polynomial whose coefficients are the elements of vector `p`.
- Example:

```
p = [4 3 2 1]
dp = polyder(p) % dp = [12 6 2]
```

That is, given a polynomial $p(x) = 5x^3 + 3x^2 + 2x + 1$, the derivative with respect to $x$ is $p'(x) = dp(x) = 12x^2 + 6x + 2$.