CS 370 - Numerical Computation **Fall 2018**

Interpolation

Lecturer: Christopher Batty *Notes By: Harsh Mistry*

The basic problem of Interpolation is, Given a set of data points from an (unknown) function $y = p(x)$, can we approximate p's value at other points

2.1 Uses for Interpolation

- Fitting curves to data. (Related to Regression)
- Estimating an unknown function's properties: values, derivatives, etc
- Interpolation plays a role in many numerical methods such as differentiation, integration, differential equations, optimization, etc

2.2 Linear Interpolation

- The simplest form of interpolation, given two points, find a line that best fits the points.
- Calculate the slope between two points and produce a line equation $y = ax + b$
- Linear interpolation breaks down when attempting to generalize solutions with more than 2 points

2.3 Polynomial Interpolation

Theorem 2.1 ***Unisolvence Theorem*** - *Given n data pairs (x_i, y_i), $i = 1, \ldots, n$ with distinct x_i, there is a unique polynomial $p(x)$ of degree $\leq n - 1$ that interpolates the data.*

- For n points, we must find all coefficients of the polynomial

$$p(x) = c_1 + c_2 x + \ldots + c_n x^{n-1}$$

- As before, each (x_i, y_i) point gives one linear equation

$$y_i = c_1 + c_2 x_i + \ldots + c_n x_i^{n-1}$$

- Then solve the n x n linear system which should yield $V\vec{c} = \vec{y}$
- V is called a Vandermonde Matrix

Note : $detV = \prod_{i<j}(x_i - x_j)$

2.4 The Monomial Basis

$p(x) = c_1 + c_2x + \ldots + c_nx^{n-1}$ is called the monomial form and can be rewritten as

$$p(x) = \sum_{i=1}^{n} c_i x^{i-1}$$

The sequence $1, x, x^2, x^3$ ... is called the monomial basis. Monomial form is a sum of coefficients c_i times these basis functions.

2.5 The Lagrange Basis

- The Lagrange basis is a different basis for interpolating polynomials.
- We define the Lagrange basis functions $L_k(x)$, to construct a polynomial as

  $$p(x) = y_1L_1(x) + y_2L_2(x) + \ldots + y_nL_n(x) = \sum_{k=1}^{n} y_kL_k(x)$$

  where y_i are coefficients
- Given n data points (x_i, y_i), we define

  $$L_k(x) = \frac{(x-x_1)(\ldots)(x-x_{k-1})(x-x_{k+1})(\ldots)(x-x_n)}{(x_k-x_1)(\ldots)(x_k-x_{k-1})(x_k-x_{k+1})(\ldots)(x_k-x_n)}$$

2.5.1 Why?

We may perfer the Lagrange basis as we can directly write down the polynomial from the Lagrange basis functions, L_k, and the data points, x_i, y_i. There is no need to solve a linear system.

2.6 Runge's Phenomenon

When involving a polynomial with a high degree, we often are left with excessive oscillation and wiggling. This is called Runge's Phenomenon.

2.6.1 Avoiding the Phenomenon

- Select data/interpolation points in a *smarter* way
- Fit even higher degree polynomials, but also constrain derivatives to somehow reduce *wiggliness*
- Fit lower degree polynomials that don't exactly interpolate, but do minimize some error measure
- Or use piecewise polynomials

2.7 Piecewise Functions and Interpolation

- As we know, piecewise functions are functions with different definitions for distinct intervals of the domain
- One option of Piecewise Interpolation, is to continually apply Liner Interpolation for each set of points, but this can result in an some what unsatisfactory interpolation which may have *kinks*
- The goal is to achieve smoothness because its beneficial for aesthetic purposes and for mathematical applications needing derivatives

2.8 Hermite Interpolation

- Greater smoothness requires controlling derivatives of the polynomial.
- **Hermite Interpolation** is the problem of fitting a polynomial given function values and derivatives.

2.8.1 Closed-form solution

If we define the Polynomial on the i^{th} interval, $p_i(x)$ as

$$p_i(x) = a_i + b_i(x - x_i) + c_i(x - x_i)^2 + d_i(x - x_i)^3$$

there exist direct formulas for polynomial coefficients

- $a_i = y_i$
- $b_i = S_i$
- $c_i = \frac{3y_i' - 2S_i - S_{i+1}}{\Delta x_i}$
- $d_i = \frac{S_{i+1} + S_i - 2y_i'}{\Delta x_i^2}$

where we define

- $\Delta x_i = x_{i+1} - x_i$
- $y_i' = \frac{y_{i+1} - y_i}{\Delta x_i}$

Definition 2.2 ***Knots*** *: Points where the interpolate transitions from one polynomial/to another*

Definition 2.3 ***Nodes*** *: Points where some control points/data is specified*

For hermite interpolation, these two points are the same, but they can differ for other curves

2.9 Cubic Splines

Fit a cubic, $S_i(x)$ on each interval, but now require matching first and second derivatives between intervals

Require "interpolating conditions" on each interval

$$S_i(x_i) = y_i, \qquad S_i(x_{i+1}) = y_{i+1}$$

and "derivative conditions" at each interior point x_{i+1}

$$S_i'(x_{i+1}) = S_{i+1}'(x_{i+1})$$

$$s_i''(x_{i+1}) = s_{i+1}''(x_{i+1})$$

2.9.1 Boundary Conditions

- Clamped/complete- endpoints have slope set to set value
- Free/Natural/Variational - "curvature" goes to zero at the end points, so the curve "straightens out"
- Periodic Boundary Conditions - start and end derivatives match each other, which gives a "wrap-around" type behaviour
- "Not-a-Knot" boundary conditions - Last two segments on an end become the same polynomial

2.9.2 Hermite v.s. Cubic Splines

- Hermite Interpolation - each interpolant can be found independently
  - Solve $n-1$ separate systems of 4 equations
- Cubic Spline - must solve for all polynomials together at once!
  - Solve one system with 4(n-1) equations

2.10 Spline Energy

There is an mathematical relationship between physical splines and cubic splines.

Bending a spline by placing the "drafting ducks" introduces some stored potential energy (the bending energy)

2.10.1 Real spline energy

The shape of a spline will find the minimum snergy configuration. (The least smooth shape)

Mathematical splines also minimize an "energy". i.e a measure of the total amount of curvature

2.10.2 Strategies for smooth curves

- Hermite Interpolation
  - Given values and slopes
  - Separate solve per interval
  - Continuous 1st derivatives
- Cubic spline interpolation
  - Given values only
  - One big solve for all coeffs.
  - Continuous 1st an 2nd derivatives

2.10.3 Size and Cost

- Naive cubic spline system has matrix size $(4n-4)^2$ for n points
- Basic algorithms for linear systems take $O(N^3)$ time for N unknowns

2.10.4 Cublic splines via Hermite Interpolation

Strategy : Use hermite interpolation as a stepping stone to build a cubic spline

1. Express unknown polys with closed form hermite equations
2. Treat s_i (slopes at nodes) as unknowns
3. Solve s_i that give continuous 2nd derivatives
4. Give s_i plug into closed form Hermite equations to recover polynomial coefficients : a_i, b_i, c_i, d_i