Lecture 13 - curve fitting & interpolation

Class Business

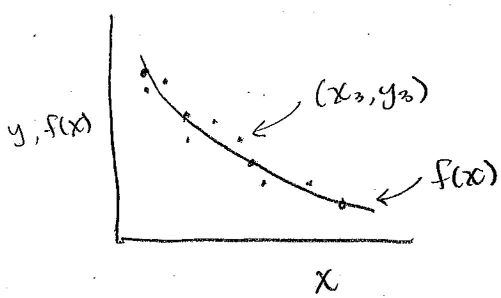
* prayer / spiritual thought
* announcements
* unit of the day

$$1\,hp = 745.7\,W.$$

I. Curve Fitting vs. Interpolation

A. Curve Fitting

* Recall that in curve fitting, we have data $\{x_i, y_i\}$, and we are trying to find a model $f(x)$ that best fits the data.



usually we know a form of $f(x)$ and we vary some parameters.

e.g. $f(x) = ax + b$, vary a, b for best fit.

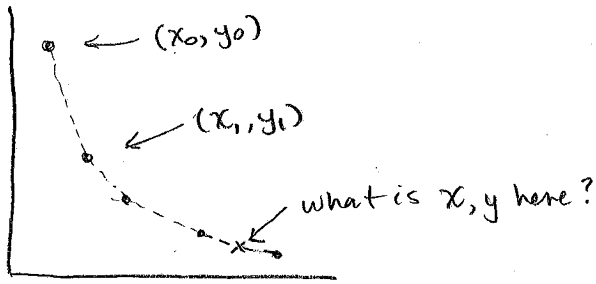
* We fit the model assuming the spread is from noise:

$$\underbrace{y_i}_{\text{data}} = \underbrace{f(x_i)}_{\text{model}} + \underbrace{\varepsilon_i}_{\text{noise}}$$

* In curve fitting we are trying to find one function that approximately fits all of the data.

B. Interpolation

* In interpolation, we have some data $\{x_i, y_i\}$ at discrete points, and we want to know the value of y for some x not in our data set.



* In interpolation, we don't have an underlying model, and we assume that the noise is negligible

* In interpolation we fit many piecewise functions that exactly fits all of the data.

Examples in python file

* Polynomial uniqueness theorem - can exactly fit an n^{th} order polynomial with $n+1$ points

  e.g. line with 2 points
  quadratic with 3 points
  cubic with 4 points
  etc.

* Runge's phenomenon - why can't we use just one polynomial? It oscillates wildly. Will need piecewise polynomials

• R-oo-ng-a rhymes w/ roomba

* Spline – a piecewise, polynomial interpolation scheme that exactly fits every data point.

  e.g. linear spline (linear interpolation)

  quadratic spline

  cubic spline ← has continuous 1st & 2nd derivatives.

Practice: curve fit or Interpolate

II. Curve Fitting & Interpolation in Python

See Examples in python file

A. Curve fitting

* numpy.polyfit() & numpy.polyval()
  - linear least squares of a polynomial
  - equivalent to LINEST in Excel

* scipy.optimize.curve_fit()
  - general, non-linear least squares
  - equivalent to minimizing the sum of squared error using solver.

* R^2: get by hand

$$R^2 = 1 - \frac{SSE}{SST} \qquad SSE = \sum_i \left[y_i - f(x_i) \right]^2$$

$$SST = \sum_i \left[y_i - \bar{y} \right]^2$$

B. Interpolation

* scipy.interpolate.interp1d()
  - 'linear', 'quadratic', 'cubic'

Practice

- Polynomial, linear least-squares
- Non-linear least-squares
- R^2
- Spline interpolation