Interpolation

Polynomial Interpolation - Global

Naive Approach → Vandermonde Matrix

Multiple methods to get the polynomial

Lagrange Interpolation

Goal: Get $f(x) = y_1 L_1(x) + y_2 L_2(x) + \dots + y_n L_n(x)$

That passes through $(x_1, y_1) \to (x_n, y_n)$

$$f(x_1) = y_1 \quad , \; f(x_2) = y_2 \; , \; \text{etc.}$$

The functions $L_i(x)$ obey

$$L_i(x_j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$

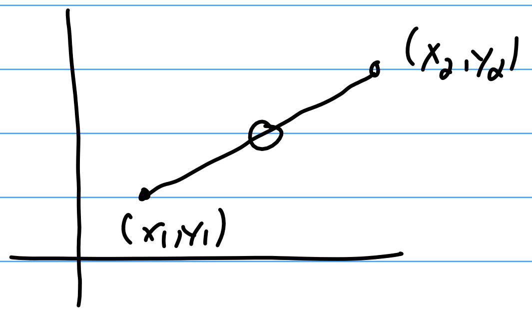
$L_1(x_1) = 1 \qquad L_1(x_2) = 0 \quad \dots \quad L_1(x_n) = 0$

Additional constraint : $\sum_{i=1}^{n} L_i(x) = 1$

If $L_i(x)$ obey these conditions then they are Lagrange Polynomials

To develop $L_i(x)$ look at (x_1, y_1) (x_2, y_2)

$x_2 > x_1$



$$f(x) = y_1 + \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x - x_1)$$

$$= \frac{y_1(x_2 - x) + y_2(x - x_1)}{(x_2 - x_1)}$$

$$= y_1 \frac{(x_2 - x)}{(x_2 - x_1)} + y_2 \frac{(x - x_1)}{(x_2 - x_1)}$$

$$= y_1 \frac{(x - x_2)}{(x_1 - x_2)} + y_2 \frac{(x - x_1)}{(x_2 - x_1)} \stackrel{?}{=} y_1 L_1(x) + y_2 L_2(x)$$

Do $L_1(x) = \dfrac{x - x_2}{x_1 - x_2}$ $L_2(x) = \dfrac{x - x_1}{x_2 - x_1}$ obey the 2 properties?

1) $L_1(x_1) = \dfrac{x_1 - x_2}{x_1 - x_2} = 1$ $\qquad L_1(x_2) = \dfrac{x_2 - x_2}{x_1 - x_2} = 0$

$L_2(x_1) = \dfrac{x_1 - x_1}{x_2 - x_1} = 0$ $\qquad L_2(x_2) = \dfrac{x_2 - x_1}{x_2 - x_1} = 1$

2) $L_1(x) + L_2(x) = \dfrac{x - x_2}{x_1 - x_2} + \dfrac{x - x_1}{x_2 - x_1} = \dfrac{x - x_2}{x_1 - x_2} + \dfrac{x_1 - x}{x_1 - x_2}$

$= \dfrac{x_1 - x_2}{x_1 - x_2} = 1$

$\Rightarrow L_1(x)$ & $L_2(x)$ are Lagrange Polynomials

What if $x = \dfrac{x_1 + x_2}{2}$ $\quad L_1(x) = 1/2$

$L_2(x) = 1/2$

For 3-points : (x_1, y_1) (x_2, y_2) (x_3, y_3)

$L_1(x) = \dfrac{(x - x_2)(x - x_3)}{(x_1 - x_2)(x_1 - x_3)}$ ← Does not include x_1
← numerator @ x_1

$L_2(x) = \dfrac{(x - x_1)(x - x_3)}{(x_2 - x_1)(x_2 - x_3)}$

$L_3(x) = \dfrac{(x - x_1)(x - x_2)}{(x_3 - x_1)(x_3 - x_2)}$

In general $\quad f(x) = \sum_{i=1}^{n} y_i \cdot L_i(x)$

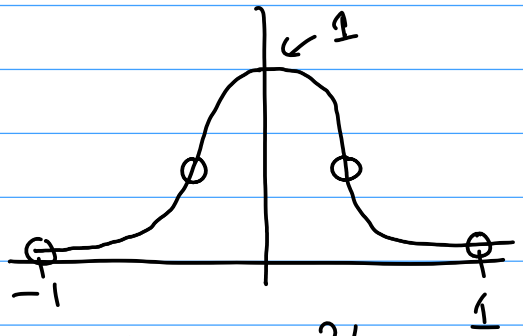
with $\quad L_i(x) = \prod_{\substack{j=1 \\ j \neq i}}^{n} \dfrac{(x - x_j)}{(x_i - x_j)}$

Runge's Phenomena

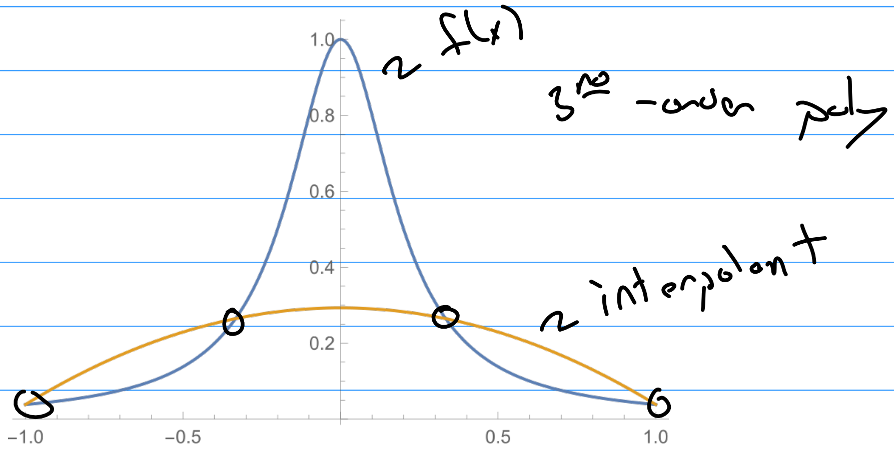
You would expect that as the # of data points increases, for a fixed region the accuracy should increase.

For uniform spacing it does <u>not</u>.

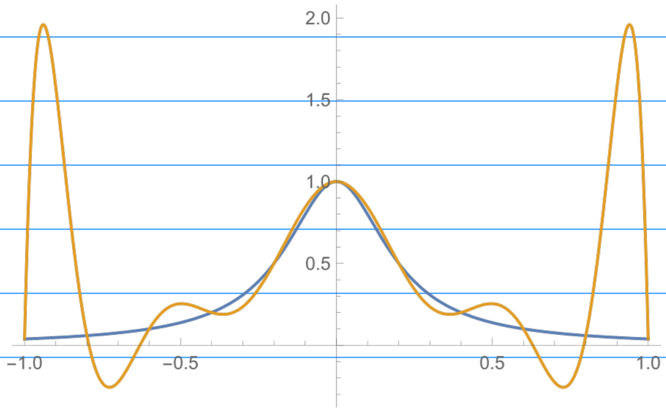
Consider $\dfrac{1}{1+25x^2}$ over $x \in [-1, 1]$



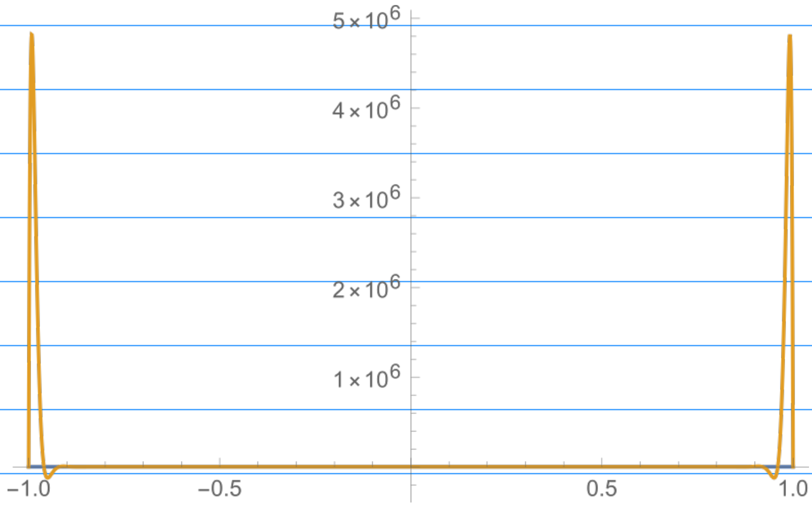
4 points, $h = 2/3$, 3^{rd}-order polynomial



10^{th}-order (11 pts)



50^{th}-order (51 pts)

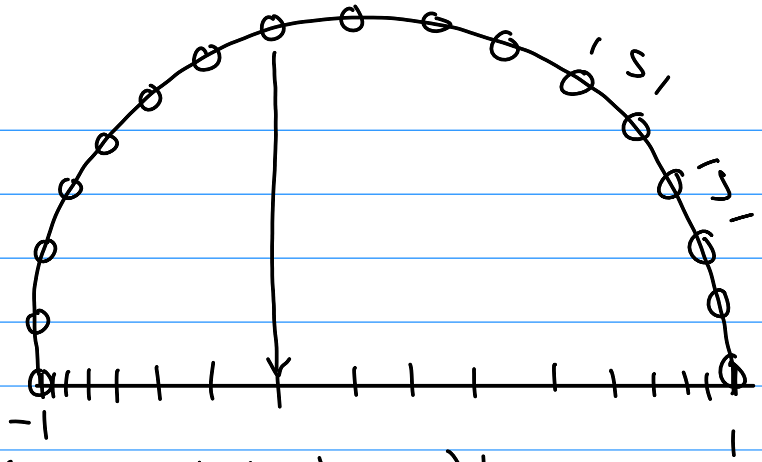


Solution: Don't use uniform meshes!

ex.) Chebyshev Points

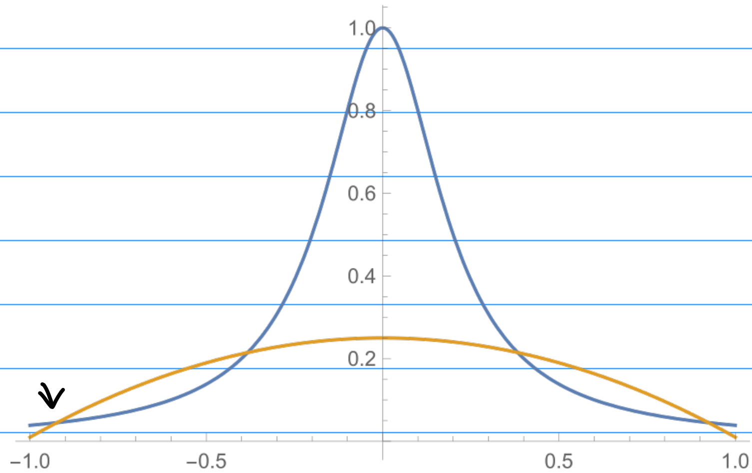
Defined over $[-1,1]$ $\quad x_i = \cos\left(\frac{2i-1}{2n}\pi\right)$

$i \in [1, n]$

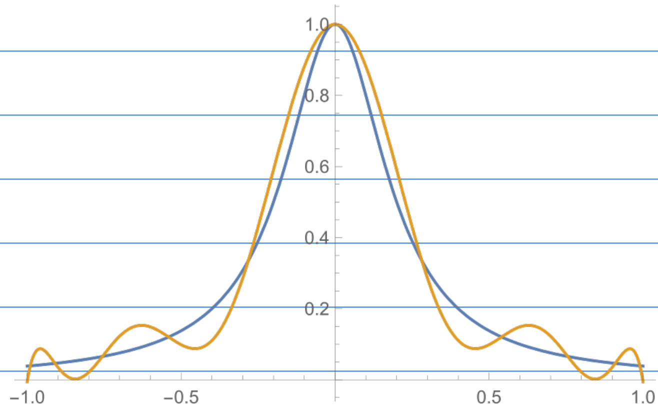


Using Chebyshev Pts:

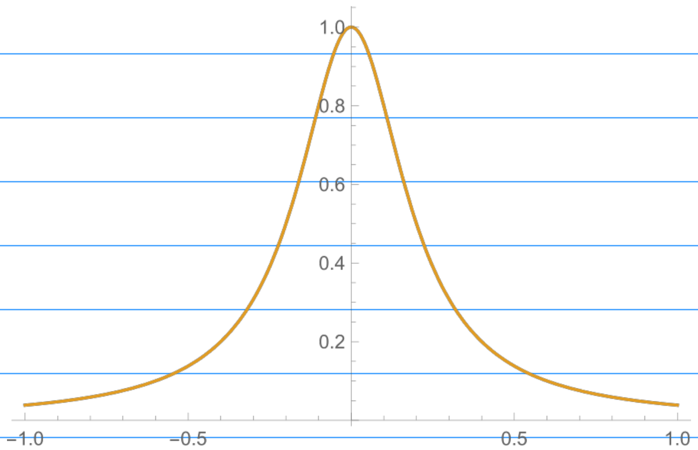
2^{nd} - order



10^{th} - order

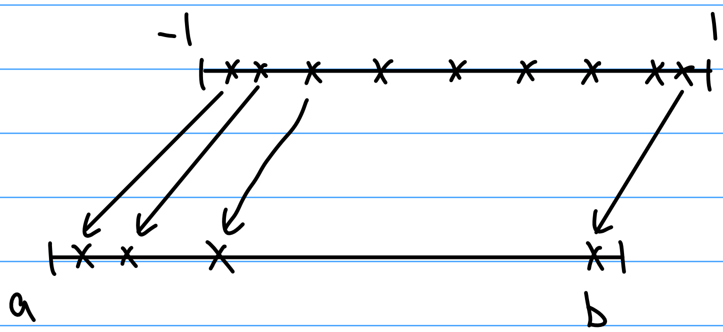


50^{th}



Chebyshev defined over $[-1, 1]$

If $x \notin [-1, 1]$ do a linear mapping



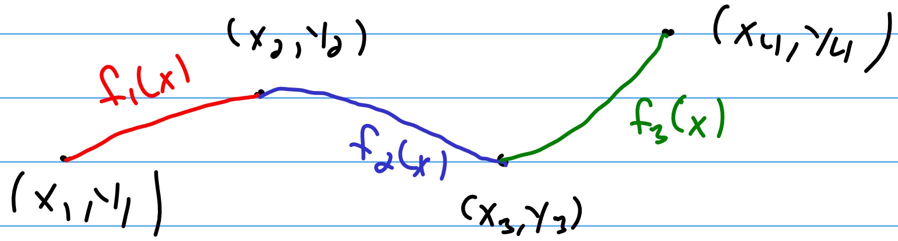
$$x_i = a + \frac{b-a}{2}(\hat{x}_i - 1)$$

point in $[a, b]$ — $x_i \in [a, b]$

Chebyshev point in $[-1, 1]$

Piecewise Interpolation

Given n-data points called knots determine n-1 interpolation functions called splines



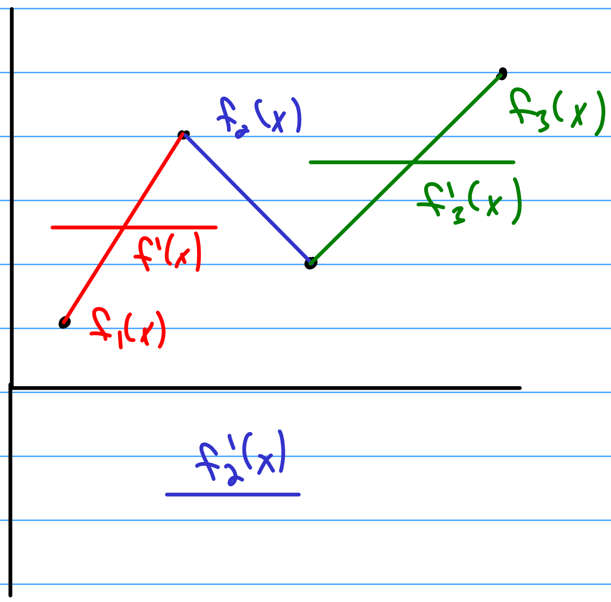
4 knots
3 splines

If $x \in [x_1, x_2)$ use $f_1(x)$
$x \in [x_2, x_3]$ use $f_2(x)$

Linear Splines $\quad f_i(x) = a_i + b_i(x - x_i)$

$$a_i = y_i$$

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



$$f_1(x_2) = f_2(x_2)$$

Linear Spline is only C^0-continuous
→ continuous in value (0th-derivative)
but no derivative

A function is C^n-continuous if all values up to the n^{th}-derivatives are continuous.

C^0 – Value
C^1 – 1^{st} derivative
C^2 – 2^{nd} derivative

To get a C^2-continuous interpolant, look at cubic splines

$$f_i(x) = \underline{a_i} + \underline{b_i}(x - x_i) + \underline{c_i}(x - x_i)^2 + \underline{d_i}(x - x_i)^3$$

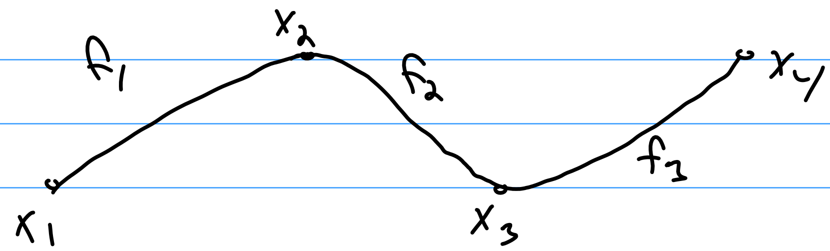
for $x \in [x_i, x_{i+1}]$

4 unknowns for each spline
$n-1$ total splines for n-data points

$\Rightarrow 4(n-1)$ unknowns
$\Rightarrow 4(n-1)$ equations

Define $h_i = x_{i+1} - x_i$
left point of $f_i(x)$ is x_i
right point of $f_i(x)$ is x_{i+1}

$x_1 < x_2 < \ldots < x_n$



$n-1$ left points
$n-1$ right points
$n-2$ common points

Conditions to meet:

#1 Each spline must return the value y @ the left point

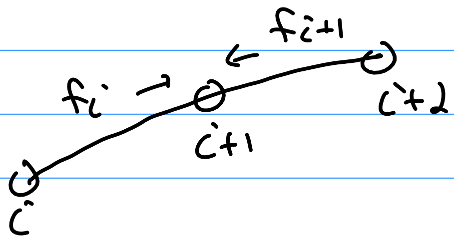
$$f_i(x_i) = y_i \quad \text{for } i = 1, \ldots, n-1$$

$$\Rightarrow a_i = y_i$$

$n-1$ equations

#1a: $f_{n-1}(x_n) = y_n$ +1 equation

#2: Splines must be C^0 at common knots



$$f_i(x_{i+1}) = f_{i+1}(x_{i+1}) \qquad i = 2, \ldots, n-1$$

$n-2$ equations

$$f_i(x_{i+1}) = a_i + b_i(x_{i+1} - x_i) + c_i(x_{i+1} - x_i)^2 + d_i(x_{i+1} - x_i)^3$$

$$= a_i + b_i h_i + c_i h_i^2 + d_i h_i^3 = a_{i+1}$$

$$f_{i+1}(x_{i+1}) = a_{i+1}$$

#3 C^1 at knots $\quad f_i'(x_{i+1}) = f_{i+1}'(x_{i+1})$

$i = 2, \dots, n-1$

$n-2$ equations

$$\underbrace{b_i + 2c_i h_i + 3d_i h_i^2}_{f_i'(x_{i+1})} = \underbrace{b_{i+1}}_{f_{i+1}'(x_{i+1})}$$

#4 C^2 @ knots $\quad f_i''(x_{i+1}) = f_{i+1}''(x_{i+1})$

$n-2$ equations

$$2\underline{c_i} + 6\underline{d_i} h_i = 2c_{i+1}$$

Add up equations

$$\underset{\#1}{(n-1)} + \underset{\#1a}{1} + \underset{C^0}{(n-2)} + \underset{C^1}{(n-2)} + \underset{C^2}{(n-2)} = 4n-6 = 4(n-1)-2$$

Need 2 more

Options:

1) Natural Cubic Spline : $f_1''(x_1) = 0$
$f_{n-1}''(x_n) = 0$

2) Clamped Cubic Spline : $f_1'(x_1) = a \in \mathbb{R}$
$f_{n-1}'(x_n) = b \in \mathbb{R}$

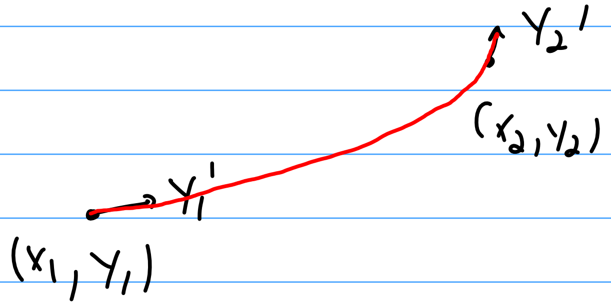
3) "not-a-knot" $\quad f_1'''(x_2) = f_2'''(x_2)$
$f_{n-2}'''(x_{n-1}) = f_{n-1}'''(x_{n-1})$

All result in a $4(n-1) \times 4(n-1)$ Linear system

Hermite Interpolation

Piecewise Intepolation using function values & Derivatives

Given (x_1, y_1, y_1') (x_2, y_2, y_2')



let $t(x) = \dfrac{x - x_i}{x_{i+1} - x_i} = \dfrac{x - x_i}{h_i}$ $\quad t \in [0, 1]$

Spline defined as:

$$p_i(t) = (2t^3 - 2t^2 + 1) y_i + (t^3 - 2t^2 + t) h_i y_i'$$
$$+ (-2t^3 + 3t^2) y_{i+1} + (t^3 - t^2) h_i y_{i+1}'$$

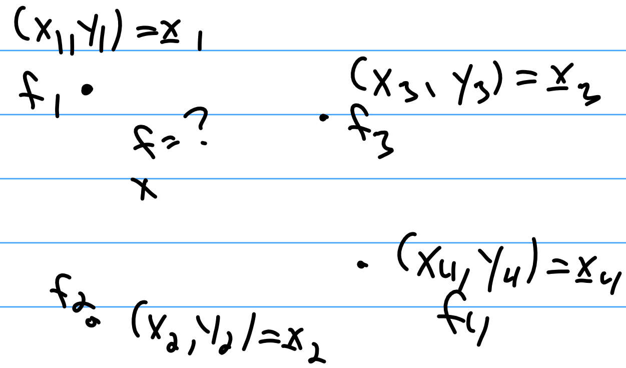
$p_i(1) = p_{i+1}(0)$ $\quad p_i'(1) = p_{i+1}'(0)$

$f_i(x) = p_i\left(\dfrac{x - x_i}{h_i}\right)$ Intepolation function

$$f_i'(x) = \frac{1}{h_i} p_i'\left(\frac{x - x_i}{h_i}\right)$$

Radial Basis Interpolation

Used for scattered (& possibly overlapping) data sets



Interpolation based on Radial Basis Kernels.
functions that depend only on the (radial) Distance between points.

$\phi(r) = \phi(\|\underline{x} - \underline{y}\|)$ = radial basis kernel

Interpolant is $S(\underline{x}) = \sum_{i=1}^{n} w_i \, \phi(\|\underline{x} - \underline{x}_i\|)$

for n-data points $\underline{x}_i$

w_i = weight of data point i

from above: $S(\underline{x}) = w_1 \phi(\|\underline{x} - \underline{x}_1\|) +$
$w_2 \phi(\|\underline{x} - \underline{x}_2\|) +$
$w_3 \phi(\|\underline{x} - \underline{x}_3\|) +$
$w_4 \phi(\|\underline{x} - \underline{x}_4\|)$

w_i obtained by enforcing $S(\underline{x}_i) = f_i$

$\Rightarrow w_1 \phi(r_{11}) + w_2 \phi(r_{12}) + \cdots + w_n \phi(r_{1n}) = f_1$

$\underline{w_1} \phi(\underline{r_{21}}) + \underline{w_2} \phi(\underline{r_{22}}) + \cdots + \underline{w_n} \phi(\underline{r_{2n}}) = \underline{f_2}$

$\vdots \qquad \vdots \qquad \vdots$

unknown: Scalar

known

$r_{ij} = ||\underline{x}_i - \underline{x}_j||$

$r_{ji} = ||\underline{x}_j - \underline{x}_i|| = ||\underline{x}_i - \underline{x}_j|| = r_{ij}$

$\Rightarrow \phi(r_{12}) = \phi(r_{21})$

Result is a linear system $\underline{A}\,\underline{w} = \underline{f}$
where $\underline{A}$ is symmetric

Example kernels:

Gaussian : $\phi(r) = e^{(-\varepsilon r)^2}$ ε = parameter
usually $\varepsilon \sim \frac{1}{h}$

Multiquadric : $\phi(r) = (1 + (\varepsilon r)^2)^{1/2}$

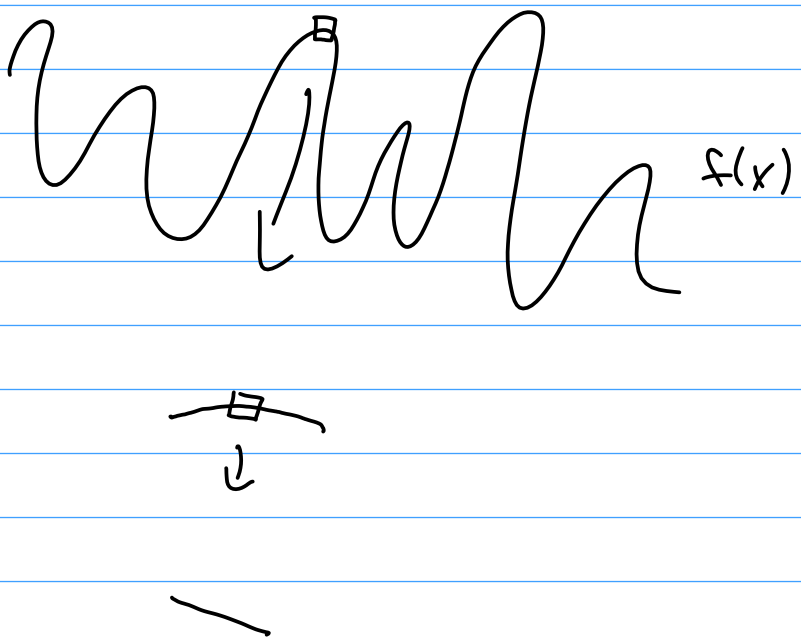
Inverse Multiquadric: $\phi(r) = \dfrac{1}{(1+(\varepsilon r)^2)^{1/2}}$

Polyharmonic Spline : $\phi(r) = r^{2m+1}$ $m \geq 1$ integer
$2m+1$ = odd #

$\phi(r) = (\ln r)(r^{2m})$ $m \geq 1$ integer
$2m$ = even integer

Notes: - Weights are Data-specific
- Matrix is ill-conditioned at small spacings / values of ε

- RBF does not describe flat data well



To fix ill-conditioning & lack of accuracy for lines, add polynomials.

$$s(\underline{x}) = \sum_{i=1}^{n} w_i \, \phi(\|\underline{x} - \underline{x}_i\|) + p(\underline{x})$$

$$p(\underline{x}) = a_{00} + a_{10}x + a_{20}x^2 + a_{01}y + a_{11}xy + a_{02}y^2$$

w/ the constraint that

$P(w_i) = 0$

$$\begin{bmatrix} \underline{A} & \underline{P} \\ \underline{P}^T & \underline{0} \end{bmatrix} \begin{bmatrix} \underline{w} \\ \underline{a} \end{bmatrix} = \begin{bmatrix} \underline{f} \\ \underline{0} \end{bmatrix}$$