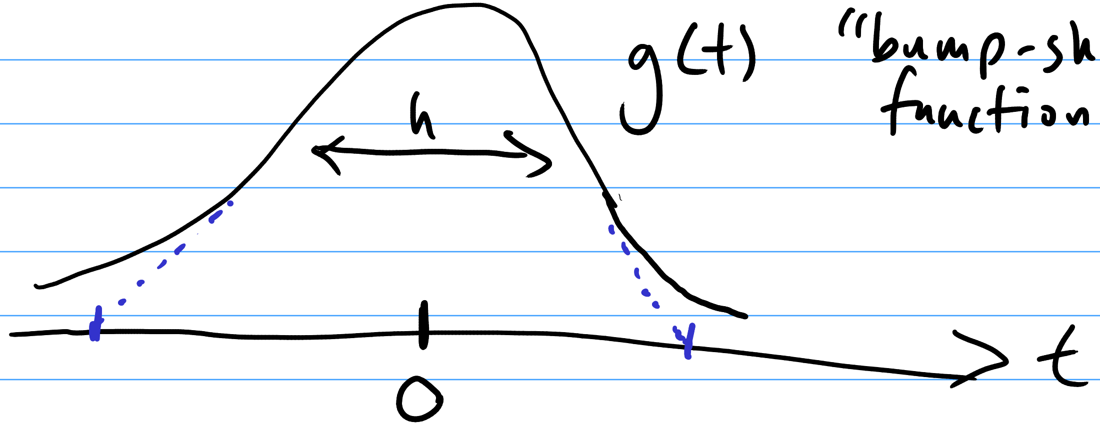
Detailed project instructions coming <u>soon</u>!

## Radial basis function networks

Goals: introduce RBFs
fitting w/ fixed centers
differences from kernel smoothers

# Kernel smoother

$h$ = bandwidth




$g(t)$ "bump-shaped function"

$$g(t) = \exp\left(-\frac{t^2}{2h^2}\right)$$

Gaussian function

$$\underbrace{K(\vec{x}, \vec{x}\,')}_{\text{kernel}} = g(\underbrace{\|\vec{x} - \vec{x}\,'\|)}_{\text{RBF}}$$

$$f(\vec{x}) = \sum_{i=1}^{n} w_i \, y_i \, K(\vec{x}, \vec{x}_i)$$

- gives smooth $f$ because $g$ smooth

Nadaraya-Watson

$$= \frac{\sum_{i=1}^{n} y_i \, K(\vec{x}, \vec{x}_i)}{\sum_{i=1}^{n} K(\vec{x}, \vec{x}_i)}$$

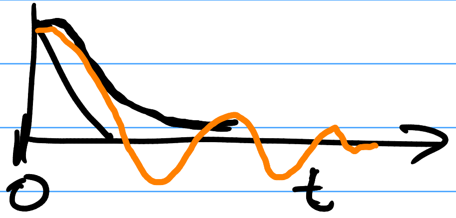
} no fitting except $h$

- biased depending on density of training samples

# Radial basis function (RBF) kernel function

w/ $$\underbrace{k(\vec{x},\vec{x}\,')}_{\text{similarity of } \vec{x},\vec{x}\,'} = g(\underbrace{\|\vec{x}-\vec{x}\,'\|}_{\text{depends only on distance}})$$

kernel function




RBF


**RBF network** has the form

$$f(\vec{x}) = \sum_{i=1}^{m} w_i \, g(\|\vec{x}-\vec{c}_i\|)$$

m ← number of centers

$w_i$: weights

$\vec{c}_i$: centers

Differences:

- no $y_i$, learn from data
- centers are not on data pts (in general)
- # centers could be different than # data pts

Simple way to pick centers

\# training pts

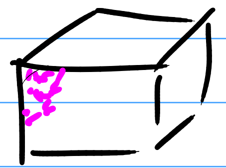
1) Set $m < n$

Sample $I_j \sim \text{Uniform}(\{1, \dots, n\})$

Set $\vec{c}_j = \vec{x}_{I_j}$

Roughly centers distributed like the data

2) Form a grid over $\vec{x}$ domain of $m$ points



Having $m < n$ could increase efficiency a lot

# Fitting an RBF network

$X$ = training set $(\vec{x}_i, y_i)_{i=1}^{n}$ regression or classification

Fixed centers: $\min_{\vec{w}} \sum_{i=1}^{n} \ell(f(\vec{x}_i), y_i) + R(\vec{w})$

$$f(\vec{x}_i) = \sum_{j=1}^{m} w_j \underbrace{g(\|\vec{x}_i - \vec{c}_j\|)}_{G_{ij}}$$

$$= \sum_{j=1}^{m} G_{ij} w_j$$

$$= (G\vec{w})_i$$

$$G = \overbrace{\begin{bmatrix} g(\|\vec{x}_1 - \vec{c}_1\|) & g(\|\vec{x}_1 - \vec{c}_2\|) \cdots \\ g(\|\vec{x}_2 - \vec{c}_1\|) & \ddots \\ \vdots & \\ g(\|\vec{x}_n - \vec{c}_1\|) & \end{bmatrix}}^{m \text{ centers}}$$

matrix of similarities

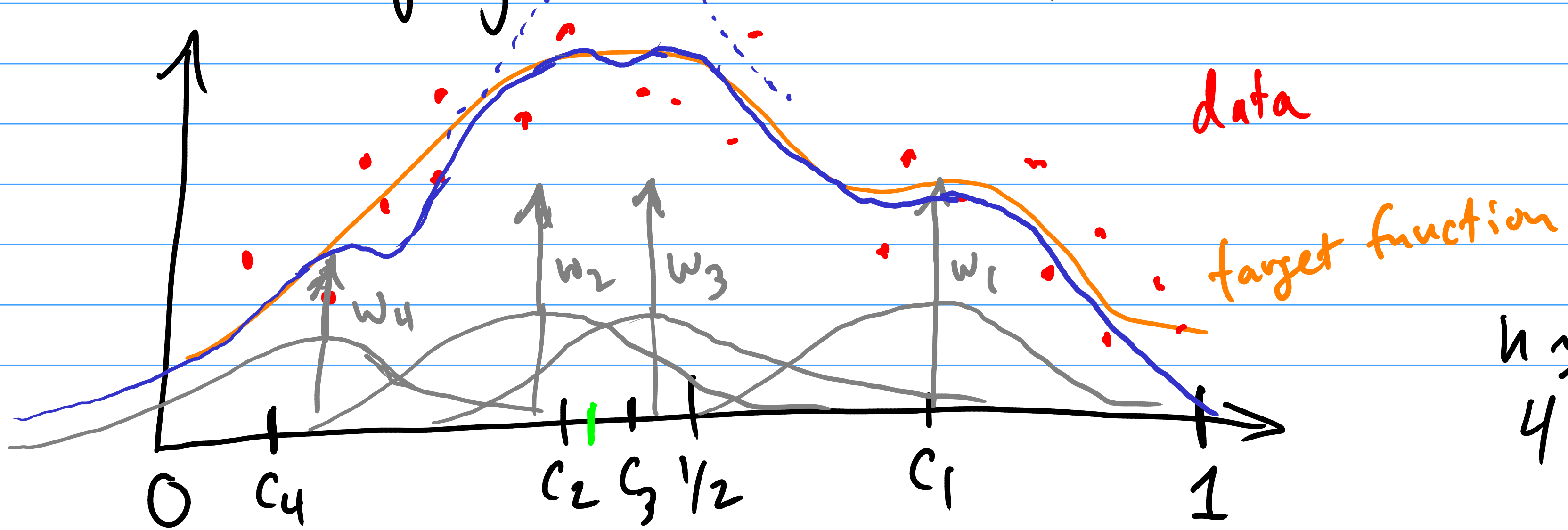
$n \times m$

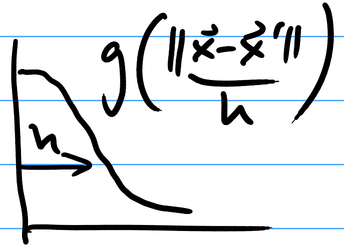
$$f(\vec{x}_i) = (G\vec{w})_i \implies \hat{y} = G\vec{w}$$

$\implies$ fitting $\vec{w}$ just like fitting a linear model

$\implies$ If $\ell$, $R$ convex, then finding $\vec{w}$ is a convex optimization

If you want to fit $\vec{c}_i$, nonconvex, can still try gradient descent.



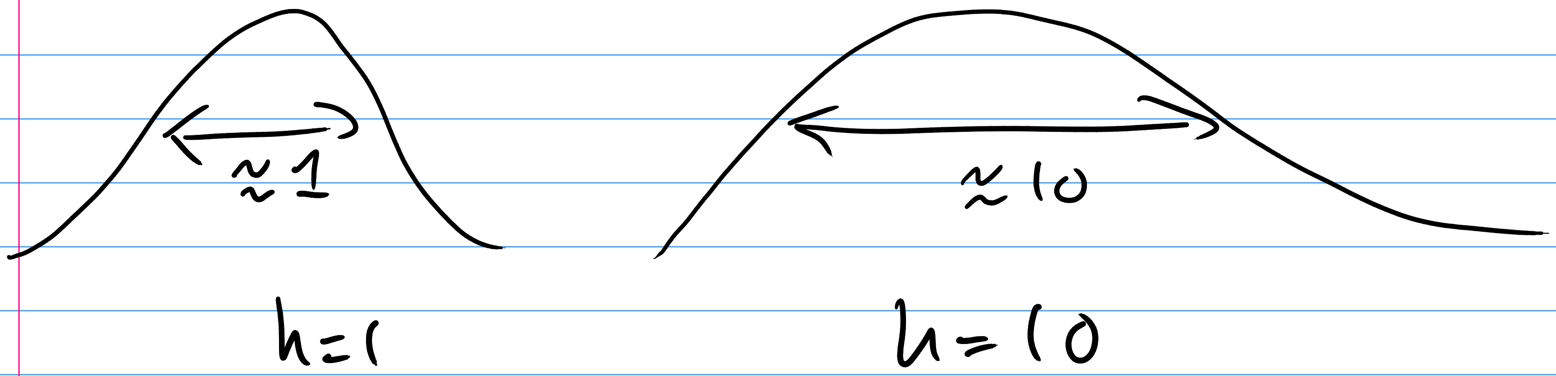

$h \approx 0.5$

4 centers

# Practical considerations:

use C.V. to pick

- bandwidth $h$ very important. Depends on
  - smoothness of target
  - # of data pts $n$, # of centers
  - input dim $d$
- $m$ also important
  - might pick $m$ based on compute



$$\underbrace{\min_{\beta_0, \vec{\beta}} \quad \| X\vec{\beta} + \mathbb{1}\beta_0 - y \|^2 + \lambda \|\vec{\beta}\|_1}_{\text{Cost}}$$

$$C(\vec{\beta}, \beta_0) = \sum_{i=1}^{n} \left( \vec{x}_i^T \vec{\beta} + \beta_0 - y_i \right)^2 + \lambda \sum_{i=1}^{d} |\beta_i|$$

$$\frac{\partial C}{\partial w_i} = 0 \qquad C = \| X\vec{w} - \vec{y} \|^2, \quad \nabla_{\vec{w}} C = 2X^T X \vec{w} - 2X^T \vec{y}$$

$i^{\text{th}}$ element of gradient $(\nabla_{\vec{w}} C)_i$

Pick out entry $i$

write as summation