Module 4: Coping with Multiple Predictors

Multidimensional Splines (Continued…)

STAT/BIOSTAT 527, University of Washington

Emily Fox

May 8th, 2014

Curse of Dimensionality

- To maintain a fixed level of accuracy for a given nonparametric estimator, the sample size must increase exponentially in *d*
- Set MSE = δ

$n \propto \left(\frac{c}{\delta}\right)^{d/4}, c>0$

- Why? Using data in local nbhd
  - □ In high dim, few points in any nbhd
- Consider example with *n* uniformly distributed points in $[-1,1]^d$
  - □ d=1: in $[-0.1, 0.1] \sim n \times \left(\frac{1}{10}\right)$
  - □ d=10 in $[-0.1, 0.1]^{10}$ $= \frac{n}{10}$ obs. in interval

$\sim n \times \left(\frac{1}{10}\right)^{10}$

$= \frac{n}{10,000,000,000}$

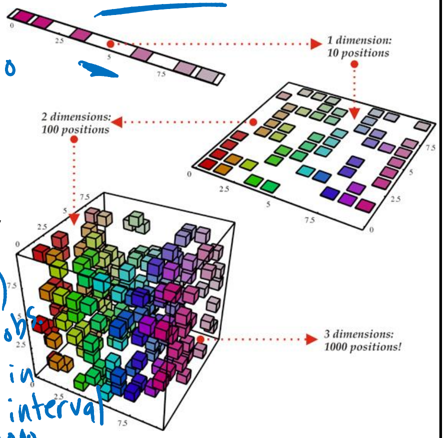


Figure from Yoshua Bengio's website

Natural Thin Plate Splines

$$\min_f \sum_{i=1}^{n} \{y_i - f(\underset{\hookleftarrow\, x_i \in \mathbb{R}^2}{x_i})\}^2 + \lambda J(f)$$

"bending energy"

$$J(f) = \int\int_{\mathbb{R}^2} \left[\left(\frac{\partial^2 f(x)}{\partial x_1^2} \right)^2 + 2 \left(\frac{\partial^2 f(x)}{\partial x_1 x_2} \right)^2 + \left(\frac{\partial^2 f(x)}{\partial x_2^2} \right)^2 \right] dx_1 dx_2$$

- Solution: Unique minimizer is the ***natural thin plate spline*** with knots at the x_{ij}
- Proof: See Green and Silverman (1994) and Duchon (1977)
- Similar properties and intuition as in 1d:
  - □ As λ→0, Sol'n approaches an interpolator
  - □ As λ→∞, LS plane (no 2nd derivative)

Tensor Product Splines

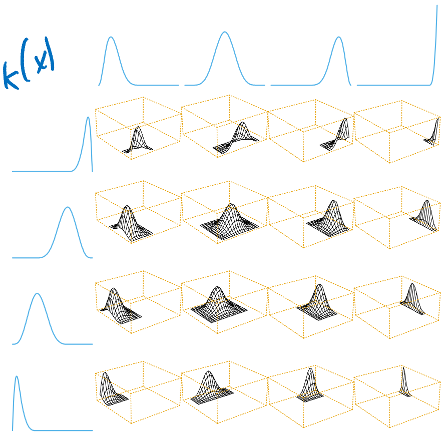
- We use this tensor product basis

$$g_{jk}(x) = h_{1j}(x_1)h_{2k}(x_2)$$

to model $f(x)$

$$\overset{x_1}{M_1} \quad \overset{x_2}{M_2}$$

$$f(x) = \sum_{j=1}^{M_1} \sum_{k=1}^{M_2} \theta_{jk}\, g_{jk}(x)$$



- This formulation extends (in theory) to any dimension d
- Note that as the dimension of the basis grows exponentially with the input dimension d

From Hastie, Tibshirani, Friedman book

Generalized Additive Models

- Both for computational reasons and added interpretability, models that assume an additive structure are very popular
- Assuming a GLM framework:

$$g(\mu(x)) = \alpha + f_1(x_1) + \ldots + f_d(x_d)$$

LM: $y = \alpha + f_1(x_1) + f_2(x_2) + \ldots + f_d(x_d)$ "j" term

$\alpha \in \mathbb{R}$

- Is this model identifiable? No, can shift α and shift f_j's to compensate → exactly same $g(\mu)$ to match change

Fix: Constrain $\sum_{i=1}^{n} f_j(x_{ij}) = 0$ → $\hat{\alpha} = \bar{y}$

- Can model $f_j(x_j)$ using any smoother

many choices! (spline, kernel method, etc.) (module 2)

©Emily Fox 2014

GAM Example

GLM: $g(\mu) = X^T\beta$

- Consider using a penalized regression spline of order p_j with L_j knots for each covariate x_j

$$\text{or } y \quad g(\mu) = \beta_0 + \sum_{j=1}^{d} \Big[\sum_{k=1}^{p_j} \beta_{jk} x_j^k + \sum_{l=1}^{L_j} b_{jl} (x_j - \xi_{jl})_+^{p_j} \Big] = f_j(x_j)$$

- Penalization is applied to the spline coefficients b_j

$$\sum_{j=1}^{d} \lambda_j \sum_{\ell=1}^{L_j} b_{j\ell}^2$$

Comments:

- The GAM is very interpretable
  - $f_i(x_i)$ is not influenced by the other $f_j(x_j)$
  - Can plot f_j to straightforwardly see the relationship between x_i and y
- Will see that this also leads to computational efficiencies

(x_j vs. y)

f_j vs. y

©Emily Fox 2014

Backfitting Algo.

- To begin, assume a standard (non-GLM) regression setting

  $y = f(x) + \varepsilon$

- For concreteness, consider

  $$\min_{f_1,\dots,f_d} \sum_{i=1}^{n} \Big(y_i - \alpha - \sum_{j=1}^{d} f_j(x_{ij})\Big)^2 + \sum_{j=1}^{d} \lambda_j \int f_j''(t_j)^2 \, dt_j$$

- Result is an ***additive cubic spline model*** with knots at the unique values of x_{ij}
  - □ For *X* full column rank, can show that solution is unique. Otherwise, linear part of $f_j(x_j)$ is not uniquely determined

- Here, clearly $\hat{\alpha} = \bar{y}$ $\quad \big(\sum_i f_j(x_{ij}) = 0\big)$

- How do we think about fitting the other parameters??

Backfitting

$$y = \alpha + \underbrace{f_1(x_1) + \dots + f_d(x_d)}_{f(x)} + \varepsilon$$

- ***Backfitting*** is an iterative fitting procedure

- Since *f*(*x*) is additive, if we condition on the fit of all other components $f_j(x_j)$, $j \neq i$, then we know how to fit $f_i(x_i)$

  i^{th} iteration: $\underbrace{y - \alpha - \sum_{j \neq i} f_j(x_j)}_{r = \text{partial residual} \dots} = f_i(x_i) + \varepsilon$

- Iterate the estimation procedure until convergence

is a fixed # if we fix $f_j(x_j)$ $(j \neq i)$

just like lasso, coordinate descent

Backfitting Algorithm

Algorithm 9.1 *The Backfitting Algorithm for Additive Models.*

1. Initialize: $\hat{\alpha} = \frac{1}{N}\sum_1^N y_i$, $\hat{f}_j \equiv 0, \forall i, j$. ← init f_j's
2. Cycle: $j = 1, 2, \dots, p, \dots, 1, 2, \dots, p, \dots,$

$$\text{Smoother step } \star \quad \hat{f}_j \leftarrow \mathcal{S}_j\left[\underbrace{\{y_i - \hat{\alpha} - \overbrace{\sum_{k \neq j} \hat{f}_k(x_{ik})\}_1^N}^{r = \text{partial resid.}}}\right], \quad \therefore \text{(spline, kernel) smoother for } x_j$$

$$\hat{f}_j \leftarrow \hat{f}_j - \frac{1}{N}\sum_{i=1}^N \hat{f}_j(x_{ij}). \quad \leftarrow \text{numerical accuracy}$$

until the functions $\hat{f}_j$ change less than a prespecified threshold.

From Hastie, Tibshirani, Friedman book

©Emily Fox 2014

Review of GLMs

$LM: E(Y) = \alpha + \beta^T x$
$Y \in \mathbb{R}$

- Mean parameters are a linear combination of inputs, passed through a possibly nonlinear function $Y \in [0,1]$

Focus on canonical form

- Assume a distribution in the exponential family

$$p(y \mid x) = \exp\left[\frac{y\overset{\text{natural param}}{\theta(x)} - \overset{\text{log-partition fcn}}{b}(\theta(x))}{\underset{\text{dispersion}}{\sigma^2}} + \underbrace{c(y, \sigma^2)}_{\text{const. wrt } \theta}\right]$$

  - □ Using theory of exponential families,

$$\mu(x) = E[Y \mid x] = b'(\theta(x))$$
$$\text{var}(Y \mid x) = \sigma^2 b''(\theta(x)) \overset{\Delta}{=} \sigma^2 V_x$$

$$\overset{\in\mathbb{R}}{\underset{\text{link}}{\text{logit}}(\underset{\in[0,1]}{\hat{p}})} = \alpha + \underset{(\in\mathbb{R})}{\beta^T x}$$

Review of GLMs

$$p(y \mid x) = \exp\left[\frac{y\theta(x) - b(\theta(x))}{\sigma^2} + c(y, \sigma^2)\right]$$

- Mean parameters are a linear combination of inputs, passed through a possibly nonlinear function

- A parametric GLM assumes

$$\underbrace{g}_{\text{"link fcn"}}(\overset{E(y)}{\mu(x)}) = \beta^T x$$

  - □ With a canonical link function,

$$\theta(x) = g(\mu(x))$$

  - □ The link function is assumed to be invertible

$$\mu(x) = g^{-1}(\theta(x))$$

Examples

$$p(y \mid x) = \exp\left[\frac{y\theta(x) - b(\theta(x))}{\sigma^2} + c(y, \sigma^2)\right]$$

- Linear regression

$\theta_i = \theta(x_i)$ $\quad b(\theta)$

$$\log p(y_i \mid x_i, \beta, \sigma^2) = \frac{y_i\tilde{\mu}_i - \frac{\tilde{\mu}_i^2}{2}}{\sigma^2} \underbrace{- \frac{1}{2}\left(\frac{y_i^2}{\sigma^2} + \log(2\pi\sigma^2)\right)}_{c(y_i, \sigma^2)}$$

$$\theta_i = \tilde{\mu}_i = \beta^T x_i$$

$$b(\theta(x)) = \frac{\theta(x)^2}{2}$$

$$\mu(x) = b'(\theta(x)) = \theta(x) = \tilde{\mu}(x)$$

$$b''(\theta) = 1 \Rightarrow var(y_i) = \sigma^2 b''(\theta) = \sigma^2$$

$$\theta(x) = g(\mu(x))$$

$$= \tilde{\mu}(x) = \mu(x)$$

$$\Rightarrow g(\cdot) = I(\cdot)$$

$$g(t) = t \quad \text{Identity link fcn}$$

Examples

$p(y_i|x_i) = \pi_i^{y_i} (1-\pi_i)^{1-y_i}$

$$p(y \mid x) = \exp\left[\frac{y\theta(x) - b(\theta(x))}{\sigma^2} + c(y,\sigma^2)\right]$$

$\pi_i = E(y_i)$ $\quad g(\downarrow) = \beta^T x$ $\quad \rightarrow \mu = g^{-1}(x^T\beta)$

$\| \quad \pi$

- Binomial regression

$$\log p(y_i \mid x_i, \beta, \sigma^2) = y_i \overbrace{\log\left(\frac{\pi_i}{1-\pi_i}\right)}^{\theta_i} + \overbrace{m\log(1-\pi_i)}^{-b(\theta)} + \overbrace{\log\binom{m}{y_i}}^{c}$$

$\wedge \; \sigma^2 = 1$

$\theta(x) = \log \frac{\pi(x)}{1-\pi(x)}$

$b(\theta(x)) = m \log(1+e^{\theta(x)})$

$\underline{\underline{\mu(x)}} = b'(\theta(x)) = \frac{m}{1+e^{\theta(x)}} e^{\theta(x)} = \underline{m\,\pi(x)}$ ✓

$\mathrm{var}(y) = b''(\theta(x)) = m\,\pi(x)(1-\pi(x))$

$\theta(x) = g(\mu(x))$

$= \log \frac{\frac{\mu(x)}{m}}{1-\frac{\mu(x)}{m}}$

$= \log \frac{\mu(x)}{m-\mu(x)}$

$g(t) = \log \frac{t}{m-t}$

ML Estimation

$$p(y \mid x) = \exp\left[\frac{y\theta(x) - b(\theta(x))}{\sigma^2} + c(y,\sigma^2)\right]$$

$\theta_i = \beta^T x_i$

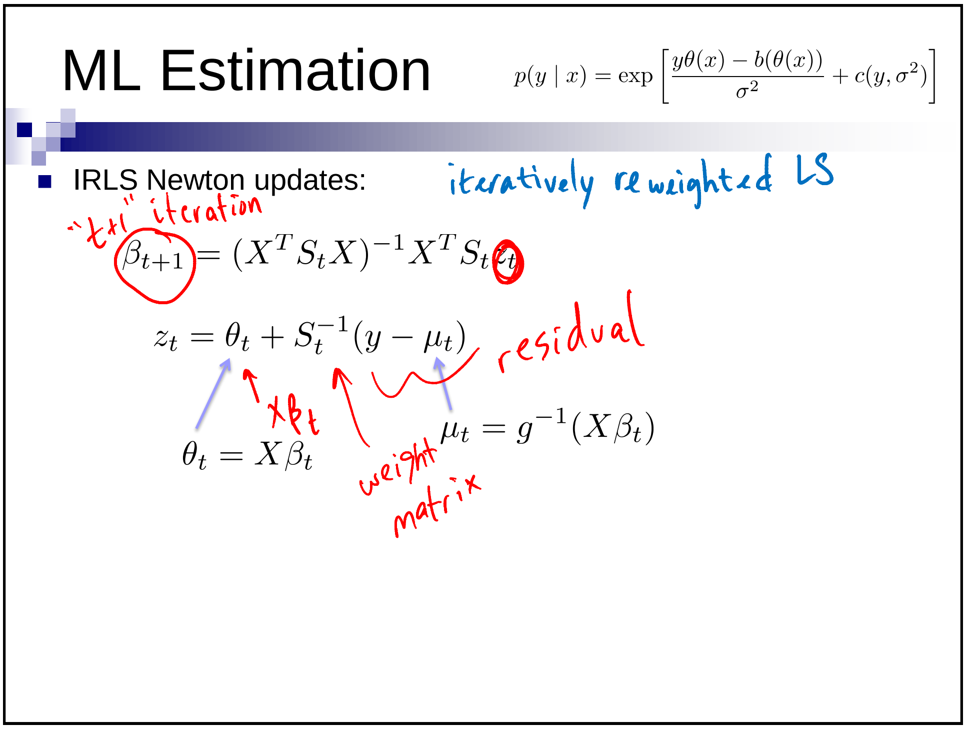
- Maximize the log-likelihood

$$\log p(y_1, \ldots, y_n \mid \beta) = \sum_{i=1}^{n} \frac{y_i\theta_i - b(\theta_i)}{\sigma^2} + \text{const}$$

$$\frac{d\ell_i}{d\beta_j} = \frac{d\ell_i}{d\theta_i}\frac{d\theta_i}{d\beta_j} = \sum_{i=1}^{n} \frac{y_i - b'(\theta_i)}{\sigma^2} \cancel{\frac{d\theta_i}{d\beta_j}} \; x_{ij} = 0$$

- No closed-form solution, so use iterative methods
  - □ 2nd order methods like IRLS require Hessian

$$H = -\frac{1}{\sigma^2} X^T S X \qquad S = \mathrm{diag}\left(\frac{d\mu_1}{d\theta_1}, \ldots, \frac{d\mu_n}{d\theta_n}\right)$$



ML Estimation

$p(y \mid x) = \exp\left[\frac{y\theta(x) - b(\theta(x))}{\sigma^2} + c(y, \sigma^2)\right]$

- IRLS Newton updates: iteratively reweighted LS

"t+1" iteration

$$\beta_{t+1} = (X^T S_t X)^{-1} X^T S_t z_t$$

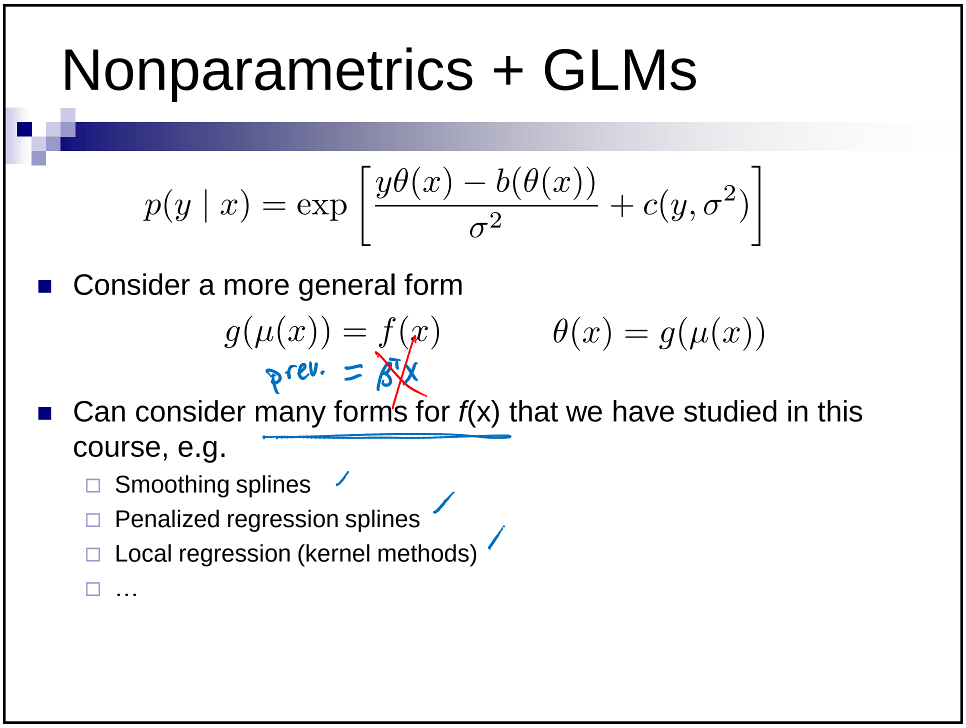
$$z_t = \theta_t + S_t^{-1}(y - \mu_t)$$

residual

$X\beta_t$

weight matrix

$$\theta_t = X\beta_t \qquad \mu_t = g^{-1}(X\beta_t)$$



Nonparametrics + GLMs

$$p(y \mid x) = \exp\left[\frac{y\theta(x) - b(\theta(x))}{\sigma^2} + c(y, \sigma^2)\right]$$

- Consider a more general form

$$g(\mu(x)) = f(x) \qquad \theta(x) = g(\mu(x))$$

prev. = $\cancel{\beta^T X}$

- Can consider many forms for *f*(x) that we have studied in this course, e.g.
  - □ Smoothing splines ✓
  - □ Penalized regression splines ✓
  - □ Local regression (kernel methods) ✓
  - □ …

GAMs and Logistic Regression

- A generalized additive logistic regression model has the form

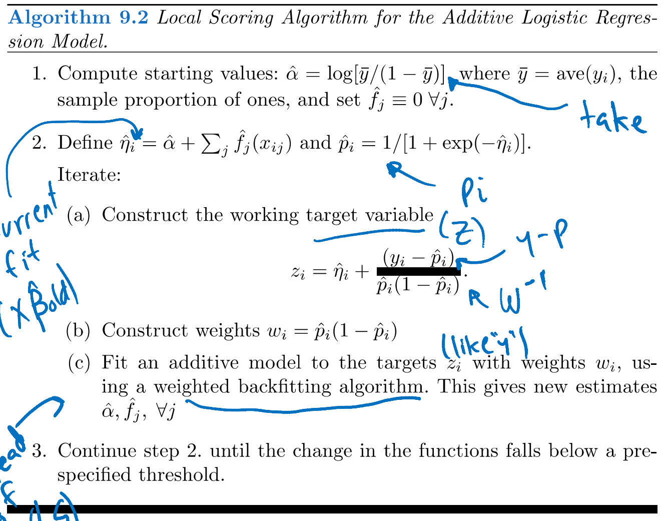
$$g(\mu) \equiv \text{logit}(P(Y=1|X)) = \alpha + f_1(x_1) + \ldots + f_d(x_d)$$

- The functions $f_1, \ldots, f_d$ can be estimated using a backfitting algorithm, too
- First, recall IRLS algorithm for *parametric* logistic regression

$$\underset{\text{"new" }y}{z} = \underset{\text{current fit}}{X\beta^{\text{old}}} + \underset{\text{weights}}{W^{-1}}(y - p) \quad p_i = p(x_i|\beta^{old})$$

$$\beta^{\text{new}} \leftarrow \arg\min_{\beta} (\underset{\text{like "y"}}{z} - X\beta)^T W (z - X\beta)$$

GAMs and Logistic Regression



Algorithm 9.2 *Local Scoring Algorithm for the Additive Logistic Regression Model.*

1. Compute starting values: $\hat\alpha = \log[\bar y/(1-\bar y)]$, where $\bar y = \text{ave}(y_i)$, the sample proportion of ones, and set $\hat f_j \equiv 0\ \forall j$. take $\bar y$, fix,
2. Define $\hat\eta_i = \hat\alpha + \sum_j \hat f_j(x_{ij})$ and $\hat p_i = 1/[1+\exp(-\hat\eta_i)]$. p_i

   current fit ($X\hat\beta^{old}$)

   Iterate:

   (a) Construct the working target variable (z)

   $$z_i = \hat\eta_i + \frac{\overset{y-p}{(y_i - \hat p_i)}}{\underset{W^{-1}}{\hat p_i(1-\hat p_i)}}.$$

   (b) Construct weights $w_i = \hat p_i(1-\hat p_i)$

   (c) Fit an additive model to the targets z_i (like "y") with weights w_i, using a weighted backfitting algorithm. This gives new estimates $\hat\alpha, \hat f_j, \forall j$

   weighted (bf) instead of (LS)

3. Continue step 2. until the change in the functions falls below a pre-specified threshold.

From Hastie, Tibshirani, Friedman book

GAM Logistic Example

- Example: *predicting spam*
- Data from UCI repository

0 1

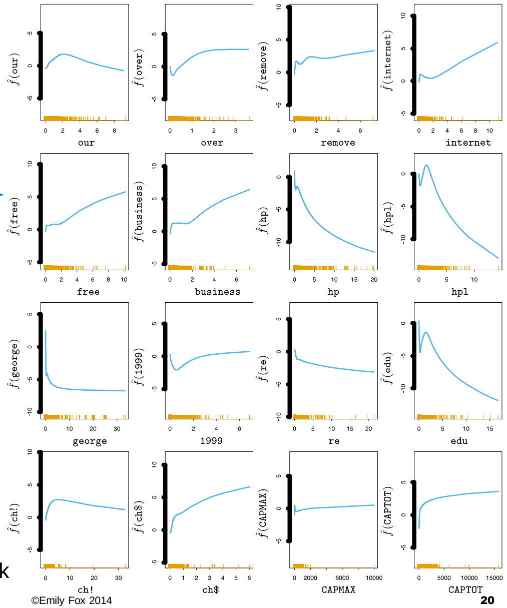
- Response variable: *email* or *spam*
- 57 predictors:
  - □ 48 quantitative – percentage of words in email that match a give word such as “business”, “address”, “internet”,…
  - □ 6 quantitative – percentage of characters in the email that match a given character (; , [! $ #)
  - □ The average length of uninterrupted capital letters: CAPAVE
  - □ The length of the longest uninterrupted sequence of capital letters: CAPMAX
  - □ The sum of the length of uninterrupted sequences of capital letters: CAPTOT

GAM Logistic Example

- Test set of 1536 emails
- Training set: n=3065
- Use a GAM with a cubic smoothing spline smoother
  - □ Each with 4 dof

$tr(L_\lambda)=4$

- Estimated functions for significant predictors
  - □ Note large discontinuity near 0 for many
- Test error of 6.6%



From Hastie, Tibshirani, Friedman book

Other GAM formulations

- Semiparametric models:

$$g(\mu) = \underset{\uparrow \text{linear model}}{X^T\beta} + \alpha + \overset{\text{(spline, etc.)}}{f(z)}$$

non-parametric smoother

- ANOVA decompositions:

$$f(x) = \alpha + \underbrace{\sum_j f_j(x_j)}_{\uparrow \text{main effects}} + \underbrace{\sum_{j<k} f_{jk}(x_j, x_k) + \dots}_{\uparrow \text{interactions}}$$

Choice of:

- □ Maximum order of interaction
- □ Which terms to include - maybe don't want some main effects / inter.
- □ What representation - splines, kernels, etc.

- Tradeoff between full model and decomposed model

Connection with Thin Plate Splines

- Recall formulation that lead to natural thin plate splines:

$$\min_f \sum_{i=1}^n \{y_i - f(x_i)\}^2 + \lambda J(f)$$

$$J(f) = \int\int_{\mathbb{R}^2} \left[\left(\frac{\partial^2 f(x)}{\partial x_1^2} \right)^2 + 2 \left(\frac{\partial^2 f(x)}{\partial x_1 x_2} \right)^2 + \left(\frac{\partial^2 f(x)}{\partial x_2^2} \right)^2 \right] dx_1 dx_2$$

- There exists a *J*(*f*) such that the solution has the form

$$f(x) = f_1(x_1) + \dots + f_d(x_d)$$

- However, it is more natural to just assume this form and apply

$$J(f) = J(f_1 + f_2 + \dots + f_d) = \sum_{j=1}^d \int f_j''(t_j)^2 dt_j$$

What you need to know

- Nothing is conceptually hard about multivariate *x*
- In practice, nonparametric methods struggle from curse of dimensionality
- Options considered:
  - □ Thin plate splines → 2d or more
  - □ Tensor product splines
  - □ Generalized additive models (use the above)
  - □ Combinations (to model some interaction terms)

Readings

- Wakefield – 12.1-12.3
- Hastie, Tibshirani, Friedman – 5.7, 9.1
- Wasserman – 4.5, 5.12

Module 4: Coping with Multiple Predictors

Multidimensional Kernel Methods

STAT/BIOSTAT 527, University of Washington

Emily Fox

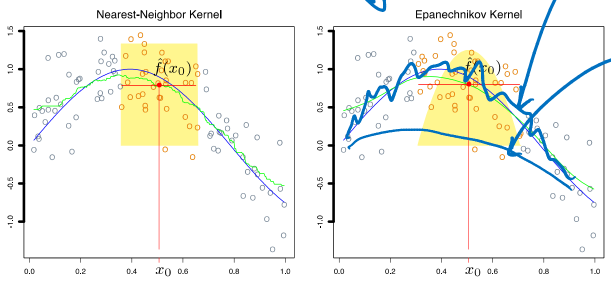
May 8th, 2014

Nadaraya-Watson Estimator

$$\hat{f}(x_0) = \frac{\sum_{i=1}^n K_\lambda(x_0, x_i) y_i}{\sum_{i=1}^n K_\lambda(x_0, x_i)}$$

- Example:
  - □ Boxcar kernel → local avgs
  - □ Epanechnikov
  - □ Gaussian typical
- Often, choice of kernel matters much less than choice of λ

small λ, low bias, high var



large λ high bias, low var

From Hastie, Tibshirani, Friedman book

Local Linear Regression

- Locally weighted averages can be badly biased at the boundaries because of asymmetries in the kernel
- Reinterpretation:

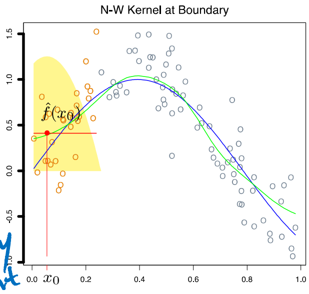
$\hat{f} = \arg\min_a \sum (y_i - a)^2$

$\rightarrow \hat{f} = \bar{Y}$

$K\left(\frac{|x_i - x|}{\lambda}\right)$

$\hat{f}(x) = \arg\min_a \sum w_i(x)(y_i - a)^2$ ← restrict to locally constant

$\rightarrow \hat{f}(x) = \frac{\sum w_i(x) y_i}{\sum w_i(x)}$



From Hastie, Tibshirani, Friedman book

- Equivalent to the Nadaraya-Watson estimator
- Locally constant estimator obtained from weighted least squares

Local Linear Regression

- Consider locally weighted linear regression instead
- Local linear model around fixed $\underline{\text{target}}$ x_0 :

$$\beta_{0x_0} + \beta_{1x_0}(x - x_0)$$

- Minimize:

$$\min_{\underline{\beta}_{x_0}} \sum_i K_\lambda(x_0, x_i)\left(y_i - \beta_{0x_0} - \beta_{1x_0}(x_i - x_0)\right)^2$$

- Return: $\hat{f}(x_0) = \hat{\beta}_{0x_0}$ ← fit at x_0

Note: not equivalent to fitting a $\underline{\text{local constant}}$!

- Fit a new local polynomial for *every* target x_0

Local Polynomial Regression

- Consider local polynomial of degree *d* centered about x_0

  $$P_{x_0}(x;\beta_{x_0}) = \beta_{0x_0} + \beta_{1x_0}(x-x_0) + \frac{\beta_{2x_0}}{2!}(x-x_0)^2 + \cdots + \frac{\beta_{dx_0}}{d!}(x-x_0)^d$$

- Minimize: $\min_{\beta_{x_0}} \sum_{i=1}^{n} K_\lambda(x_0,x_i)(y_i - P_{x_0}(x;\beta_{x_0}))^2$
- Equivalently:

  $$\min_{\beta_{x_0}} (y - \underbrace{X_{x_0}}_{\begin{bmatrix} 1 & x_1-x_0 & \cdots & \frac{(x_1-x_0)^d}{d!} \\ \vdots & & & \\ 1 & x_n-x_0 & \cdots & \frac{(x_n-x_0)^d}{d!} \end{bmatrix}} \beta_{x_0})^T W_{x_0} (y - X_{x_0}\beta)$$

- Return: $\hat{f}(x_0) = \hat{\beta}_{0x_0}$
- Bias only has components of degree *d+1* and higher

Local Polynomial Regression

- Rules of thumb:
  - Local linear fit helps at boundaries with minimum increase in variance
  - Local quadratic fit doesn't help at boundaries and increases variance
  - Local quadratic fit helps most for capturing curvature in the interior
  - Asymptotic analysis →
    local polynomials of odd degree dominate those of even degree
    (MSE dominated by boundary effects)
  - Recommended default choice: **local linear regression**

Local Polynomial Regression

- Kernel smoothing and local regression extend straightforwardly to the multivariate *x* scenario

$$\min_{\beta_{x_0}} \sum_{i=1}^{n} K_\lambda(x_0, \overset{\in \mathbb{R}^d}{x_i})(y_i - P_{x_0}(x;\beta_{x_0}))^2$$

  - □ Need *d*-dimensional kernel

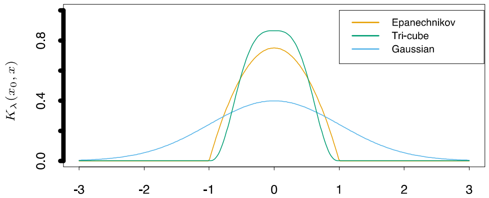
$$K_\lambda(x_0,\cdot) : \mathbb{R}^d \to \mathbb{R} \quad (\text{kernel weights})$$

  - □ Nadaraya-Watson kernel smoother fits locally constant model
  - □ Local linear regression fits local hyperplane via weighted LS
  - □ …

- Challenges:
  - □ Defining kernel ①
  - □ Curse of dimensionality ②

Example Univariate Kernels

- *Gaussian* $K(x) = \frac{1}{2\pi}e^{-\frac{x}{2}}$
- *Epanechnikov* $K(x) = \frac{3}{4}(1-x)^2 I(x)$ ← ind. on -1,1
- *Tricube* $K(x) = \frac{70}{81}(1-|x|^3)^3 I(x)$
- *Boxcar* $K(x) = \frac{1}{2}I(x)$



From Hastie, Tibshirani, Friedman book

Multivariate Kernels

- Many choices, even more than in 1d
- Examples:
  - □ Radial basis kernels

$$K_\lambda(x_0, x) = K\left(\frac{\overset{\mathbb{R}^d}{\|x_0 - x\|}}{\lambda}\right)$$

just compute distance in $\mathbb{R}^d$ and apply kernel as before

E.g., radial Epanechnikov, tricube, squared exponential (Gaussian)

$$\text{SE} \quad K_\lambda(x_0, x) = e^{-\frac{1}{2\lambda}\|x_0 - x\|^2}$$

Multivariate Kernels

- Many choices, even more than in 1d
- Examples:
  - □ Product kernels

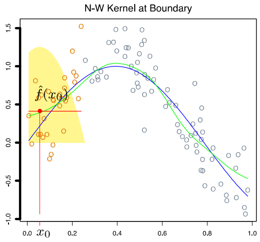
$$K_{\lambda_1,\lambda_2}(\underset{\mathbb{R}^2}{x_0}, x) = K_1\left(\frac{x_{01} - x_1}{\lambda_1}\right) K_2\left(\frac{x_{02} - x_2}{\lambda_2}\right)$$

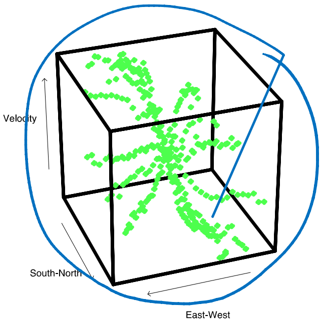
- Choices:
  - □ Form
  - □ Kernel(s)
  - □ Bandwidth(s)

Motivating Local Linear Regression

- Nadaraya-Watson smoothing can be applied to multivariate *x*
- However, boundary issues are even worse in higher dimensions
  - □ Messy to correct for boundary even in 2d (esp. for irregular boundaries)
  - □ Fraction of points close to the boundary increases with dimension

- Local polynomial regression corrects boundary errors up to desired order





From Hastie, Tibshirani, Friedman book

Local Linear Regression

- Assume a RBF kernel $K_\lambda(x_0, x_i) = K\left(\frac{\|x_0 - x_i\|}{\lambda}\right) = w_i(x_0)$

- For each target location x_0, goal is to minimize

$$\min_{\beta_{x_0}} \sum_{i=1}^{n} K_\lambda(x_0, x_i)\Big(y_i - \beta_{0x_0} - \sum_{j=1}^{d} \beta_{jx_0}(x_{ij} - x_{0j})\Big)^2$$

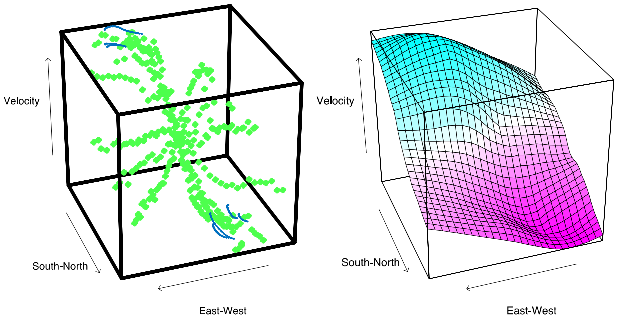
- Equivalently,

$$\min_{\beta_{x_0}} \overbrace{(y - X\beta_{x_0})^T \underset{\uparrow \, \mathrm{diag}(w_i(x_0))}{W_{x_0}} (y - X\beta_{x_0})}^{\text{local LM}}$$

- Solution: $\hat{\beta}_{x_0} = (X_{x_0}^T W_{x_0} X_{x_0})^{-1} X_{x_0}^T W_{x_0} y$
- Return: $\hat{f}(x_0) = \hat{\beta}_{0,x_0}$

Local Linear Example

- Astronomical study
  - □ Response = velocity measurements on a galaxy
  - □ Predictors = two positions
- Note the unusual star-shaped design → very irregular boundary
  - □ Must interpolate over regions with very few observations near boundary



From Hastie, Tibshirani, Friedman book

Motivating Local Polynomial

- One way to think about motivating local polynomials is as follow
- Consider 2d example for simplicity
- For a suitably smooth function *f(x)* = *f*(x_1,x_2), we can approximate it for values *x*=[x_1,x_2] in a nbhd of x_0=[x_{01},x_{02}] as

$$f(x) \approx f(x_0) + (x_1 - x_{01})\frac{\partial f}{\partial x_{01}} + (x_2 - x_{02})\frac{\partial f}{\partial x_{02}}$$

$$+ (x_1 - x_{01})^2 \frac{1}{2}\frac{\partial^2 f}{\partial x_{01}^2} + (x_1 - x_{01})(x_2 - x_{02})\frac{1}{2}\frac{\partial^2 f}{\partial x_{01}\partial x_{02}} + (x_2 - x_{02})^2 \frac{1}{2}\frac{\partial^2 f}{\partial x_{02}^2}$$

interaction

- Suggests the use of a local polynomial:

$$P_{x_0}(x, \beta_{x_0}) = \beta_{0,x_0} + (x_1 - x_{01})\beta_{1,x_0} + \dots + (x_1 - x_{01})^2 \beta_{3,x_0} + \dots$$

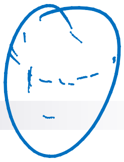
(all as above)

- Then, $\min_{\beta_{x_0}} \sum_{i=1}^{n} K_\lambda(x_0, x_i)(y_i - P_{x_0}(x;\beta_{x_0}))^2$

Scaling to High Dimensions

- Local regression becomes less useful in dimensions greater than 2 or 3
  - □ Impossible to maintain localness (low bias) and large sample size (low variance) without the total sample size increasing exponentially in *d*
- Again, curse of dimensionality
  - □ Sparsity of data
  - □ Points concentrate at boundaries
- Visualization of the fitted function is also hard in high dimensions, and visualization is often a key goal in smoothing

Boundary Effects



- Everything is far away in high dimensions
- Consider *n* data points uniformly distributed in a *d*-dimensional unit ball
- Example task: Consider nearest neighbor estimate at origin
- Median distance to closest data point is $\left(1 - \frac{1}{2}^{1/n}\right)^{d}$
  - □ For *n*=500 and *d*=10, distance ≈ 0.52
  - □ Closest point is likely more than ½ way to the boundary

most points are closer to boundary of the sample than to any other data point

- Prediction is harder near the edges of the sample boundary

Boundary Effects II

- Another way to think of this effect is in terms of volume
- We want to compute the fraction of volume that lies between radius R = 1 − ε and R = 1
- The volume of a sphere is proportional to $V(R) \propto R^d$
- The volume fraction is therefore:

$$\frac{\overset{1^d}{V_d(1)} - \overset{(1-\epsilon)^d}{V_d(1-\epsilon)}}{V_d(1)\ 1^d} = 1 - (1-\epsilon)^d \xrightarrow{d\uparrow} 1$$

as "d" grows, all pts. are at edges

- Most of the volume of a sphere is concentrated in a thin shell near the surface

Structured Local Regression

- As we have seen before, when faced with data scarcity relative to model complexity, assume structure
- Structured kernels
  - □ Place more or less importance on certain dimensions (or combinations thereof) by modifying the kernel
- Structured regression functions
  - □ Just as with splines, decompose the target regression function
  - □ E.g., ANOVA decompositions and fit low-dim terms with local regression

Structured Kernels

- In many scenarios, RBF or *spherical* kernels are considered
- Places equal weight on all dimensions of *x*
  - □ Typically, standardize data so all dimensions have unit variance
- More generally, can consider structured kernels

modifies distance metric

$$K_{\lambda,A}(x_0,x) = K\left(\frac{(x-x_0)^T A(x-x_0)}{\lambda}\right)$$

- Choices for A
  - □ Diagonal → increase, decrease, or omit influence of any x_j.
  - □ Low rank → useful in presence of correlated X.
  - □ General

$A = U^T U$; $z = Ux \rightarrow x^T A x = z^T z$

©Emily Fox 2014

Projection Pursuit Regression

- To help deal with high-dimensional regression, consider

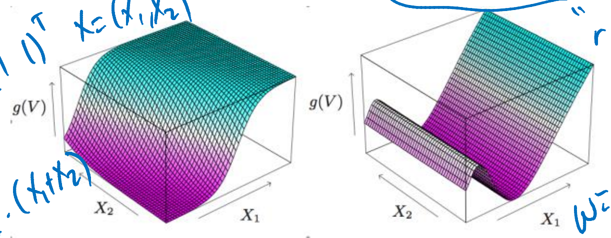
$$f(x_1,\ldots,x_d) = \alpha + \sum_{m=1}^{M} f_m(w_m^T x)$$

additive model, but ~~derived~~ in terms of derived features "$w_m^T x$".

proj. of X onto w_m

$d \times 1$ unit vector

  - □ $||w_m|| = 1$ for $m=1,\ldots,M$
- Seek w_m so the model fits well

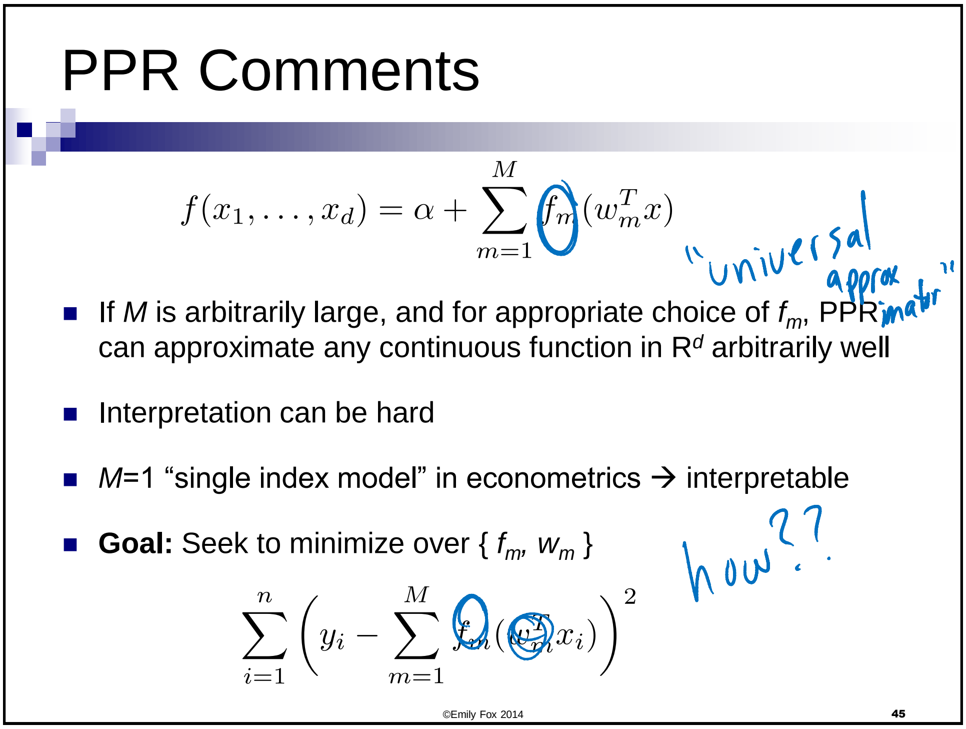


$w_m = \frac{1}{\sqrt{2}}(1\ 1)^T$ $x = (x_1, x_2)$

$w_m^T x = c \cdot (x_1 + x_2)$

"ridge fn's" in R^d (only vary in "w_m" dir'n) $w = (1,0)$

©Emily Fox 2014



PPR Comments

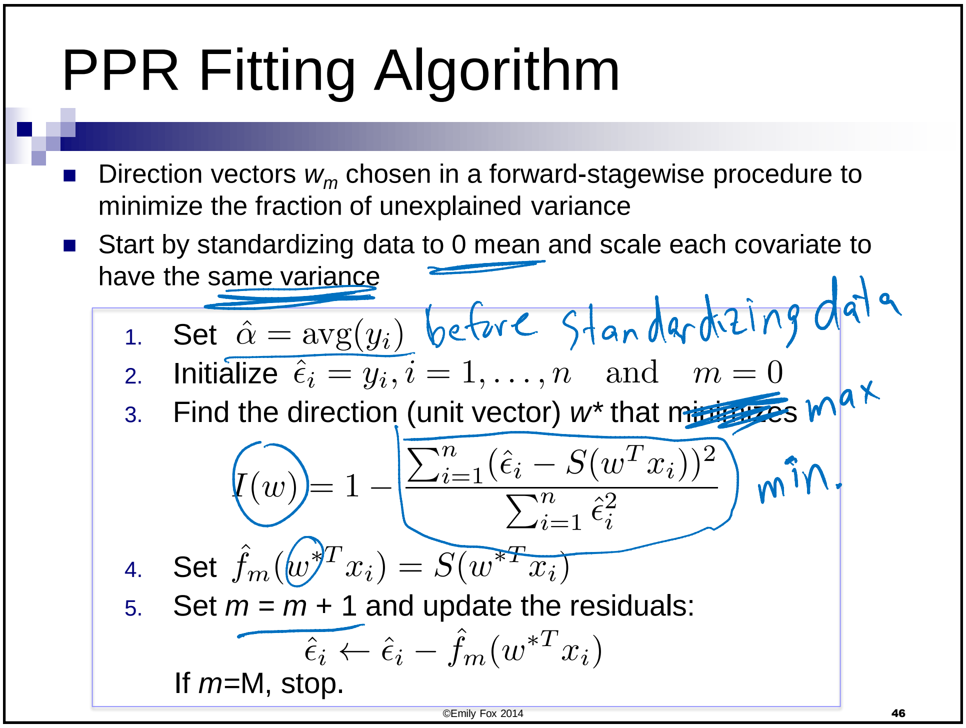
$$f(x_1, \ldots, x_d) = \alpha + \sum_{m=1}^{M} f_m(w_m^T x)$$

"universal approximator"

- If *M* is arbitrarily large, and for appropriate choice of f_m, PPR can approximate any continuous function in R^d arbitrarily well
- Interpretation can be hard
- *M*=1 "single index model" in econometrics → interpretable
- **Goal:** Seek to minimize over { f_m, w_m }

$$\sum_{i=1}^{n} \left(y_i - \sum_{m=1}^{M} f_m(w_m^T x_i) \right)^2$$

how??



PPR Fitting Algorithm

- Direction vectors w_m chosen in a forward-stagewise procedure to minimize the fraction of unexplained variance
- Start by standardizing data to 0 mean and scale each covariate to have the same variance

1. Set $\hat{\alpha} = \text{avg}(y_i)$ before standardizing data
2. Initialize $\hat{\epsilon}_i = y_i, i = 1, \ldots, n \quad \text{and} \quad m = 0$
3. Find the direction (unit vector) *w** that ~~minimizes~~ max

$$I(w) = 1 - \frac{\sum_{i=1}^{n} (\hat{\epsilon}_i - S(w^T x_i))^2}{\sum_{i=1}^{n} \hat{\epsilon}_i^2}$$

min.

4. Set $\hat{f}_m(w^{*T} x_i) = S(w^{*T} x_i)$
5. Set *m* = *m* + 1 and update the residuals:

$$\hat{\epsilon}_i \leftarrow \hat{\epsilon}_i - \hat{f}_m(w^{*T} x_i)$$

If *m*=M, stop.

PPR Fitting Algorithm Comments

$$f(x_1, \dots, x_d) = \alpha + \sum_{m=1}^{M} f_m(\boxed{w_m^T x}) \in R^1$$

- Algorithm considered is a greedy forward-wise procedure
- After each step, the f_m's from the previous steps can be readjusted using backfitting
- Can lead to fewer terms, but unclear if it improves predictions
- Typically the w_m's are not readjusted
- Choice of *M* can be based on a threshold in improvement of fit or using CV

Structured Regression Functions

- Often, instead of structuring the kernel, it makes sense and is simpler to structure the regression function itself
- Just as with splines, we can consider ANOVA decompositions

$$f(x_1, x_2, \dots, x_p) = \alpha + \sum_j f_j(x_j) + \sum_{k<\ell} f_{k\ell}(x_k, x_\ell) + \dots$$

or, more simply, standard GAMs

$$f(x_1, x_2, \dots, x_p) = \alpha + \sum_j f_j(x_j)$$

- Can use ***1d (or low-dim) local regression*** as the smoother for each term and fit using backfitting algorithm

Kernel Density Estimation

- Kernel methods are often used for density estimation (actually, classical origin)
- Assume random sample $X_1, \dots, X_n \overset{iid}{\sim} P$
- Choice #1: empirical estimate? $\hat{p} = \frac{1}{n} \sum \delta_{x_i}$

- Choice #2: as before, maybe we should use an estimator

$$\hat{p}(x_0) = \frac{\# x_i \in \text{Nbhd}(x_0)}{n\lambda \leftarrow} \text{ width of nbhd}$$

- Choice #3: again, consider kernel weightings instead

$$\hat{p}(x_0) = \frac{1}{n\lambda} \sum K_\lambda(x_0, x_i)$$ Parzen est.

Kernel Density Estimation

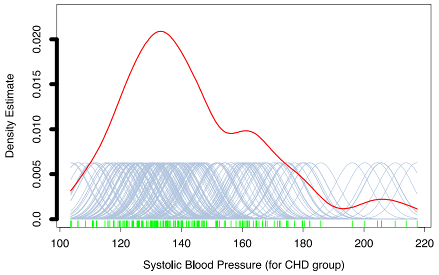
- Popular choice = Gaussian kernel → ***Gaussian KDE***

ϕ_λ

$$\hat{p} = \frac{1}{n} \sum_{i=1}^{n} \phi_\lambda(x - x_i)$$

$$= (\hat{p} * \phi_\lambda)(x)$$

↑ empirical dist.



From Hastie, Tibshirani, Friedman book

Multivariate KDE

- In 1d $\hat{p}(x_0) = \frac{1}{n\lambda}\sum_{i=1}^{n} K_\lambda(x_0, x_i)$
- In R^d, assuming a product kernel,

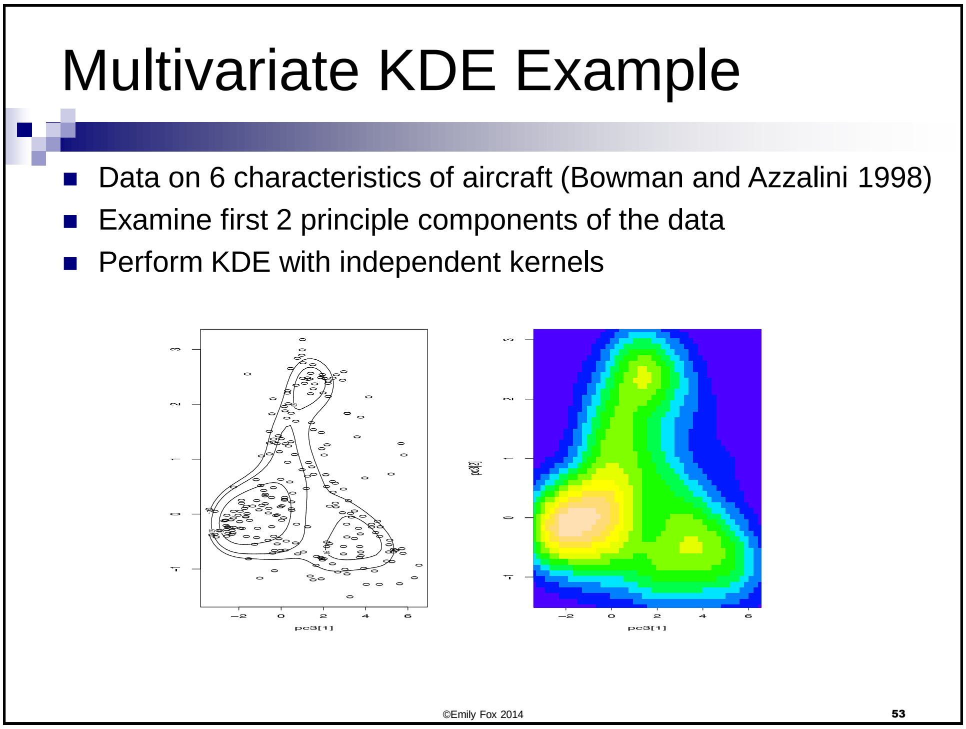
$$\hat{p}(x_0) = \frac{1}{n\lambda_1 \cdots \lambda_d}\sum_{i=1}^{n}\left\{\prod_{j=1}^{d} K_{\lambda_j}(x_{0j}, x_{ij})\right\}$$

- Typical choice = Gaussian RBF

Multivariate KDE

$$\hat{p}(x_0) = \frac{1}{n\lambda_1 \cdots \lambda_d}\sum_{i=1}^{n}\left\{\prod_{j=1}^{d} K_{\lambda_j}(x_{0j}, x_{ij})\right\}$$

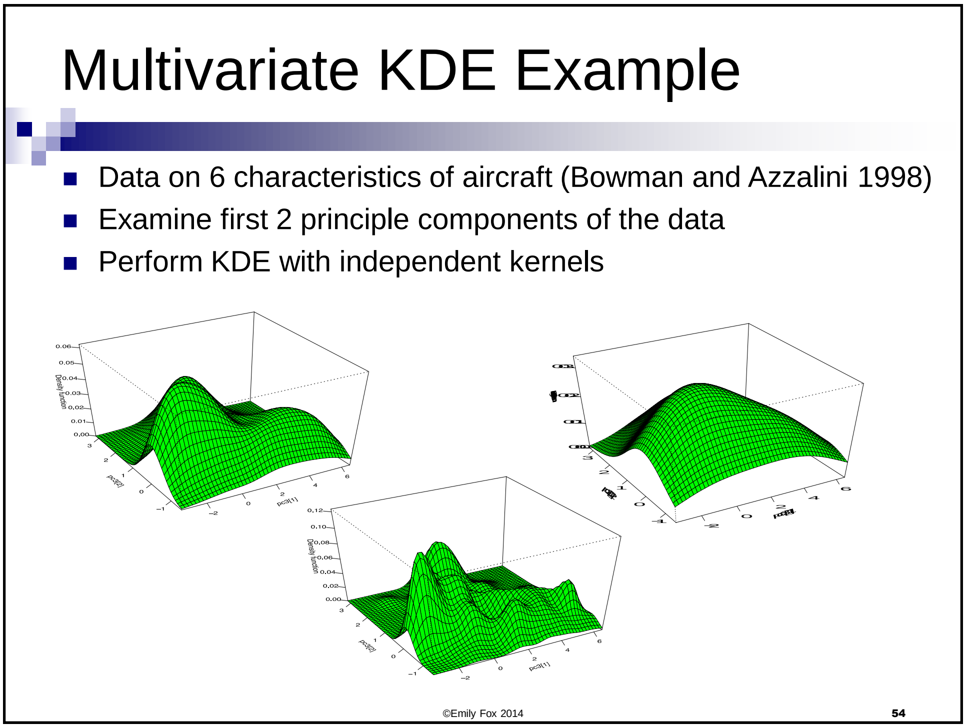
- Risk grows as $O(n^{-4/(4+d)})$
- Example: To ensure relative MSE < 0.1 at 0 when the density is a multivariate norm and optimal bandwidth is chosen
- Always report confidence bands, which get wide with *d*



Multivariate KDE Example

- Data on 6 characteristics of aircraft (Bowman and Azzalini 1998)
- Examine first 2 principle components of the data
- Perform KDE with independent kernels

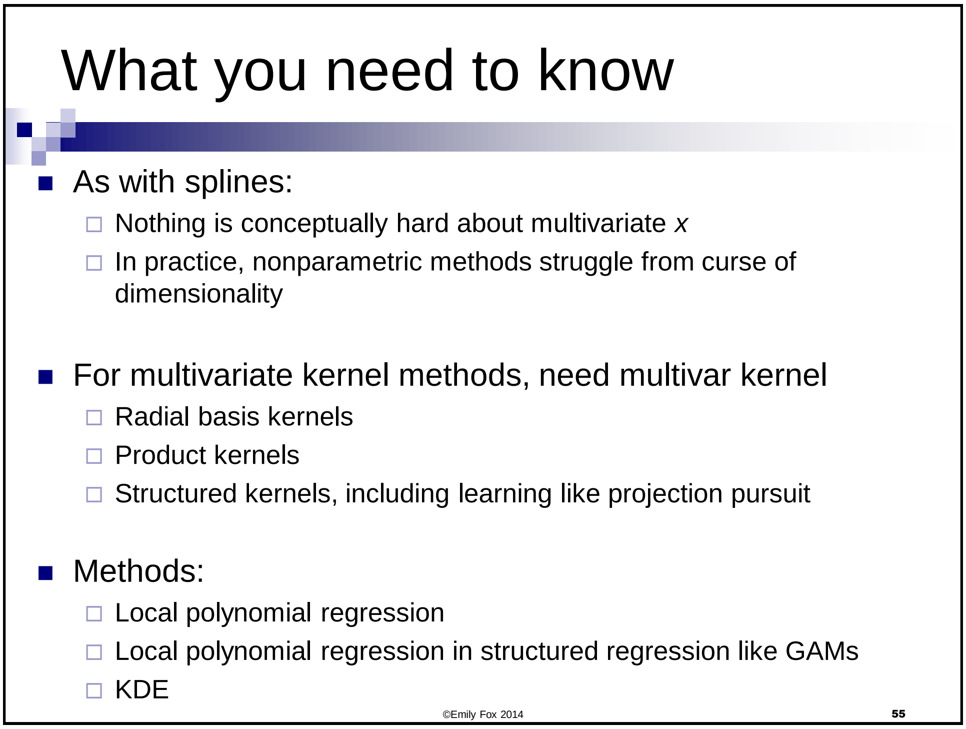
©Emily Fox 2014



Multivariate KDE Example

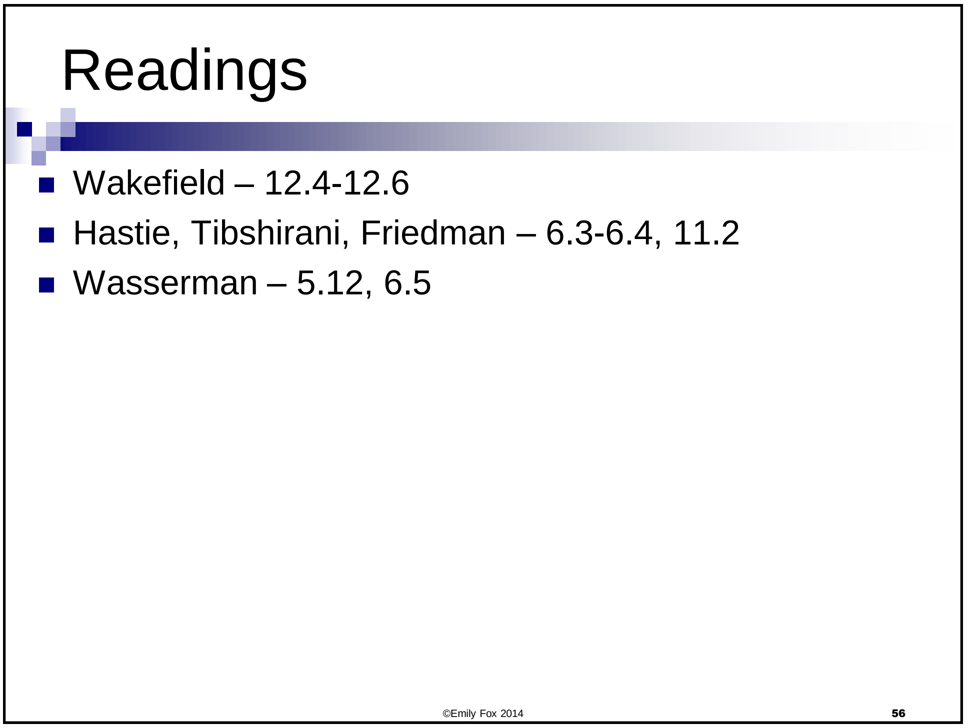
- Data on 6 characteristics of aircraft (Bowman and Azzalini 1998)
- Examine first 2 principle components of the data
- Perform KDE with independent kernels

©Emily Fox 2014



What you need to know

- As with splines:
  - □ Nothing is conceptually hard about multivariate *x*
  - □ In practice, nonparametric methods struggle from curse of dimensionality
- For multivariate kernel methods, need multivar kernel
  - □ Radial basis kernels
  - □ Product kernels
  - □ Structured kernels, including learning like projection pursuit
- Methods:
  - □ Local polynomial regression
  - □ Local polynomial regression in structured regression like GAMs
  - □ KDE



Readings

- Wakefield – 12.4-12.6
- Hastie, Tibshirani, Friedman – 6.3-6.4, 11.2
- Wasserman – 5.12, 6.5