Module 4: Coping with Multiple Predictors

Multidimensional Splines Recap

STAT/BIOSTAT 527, University of Washington
Emily Fox
May 14th, 2013

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
  - Tensor product splines
  - Generalized additive models
  - Combinations (to model some interaction terms)

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

  $n \propto \left(\frac{c}{\delta}\right)^{\frac{d}{4}}$
- Why? Using data in local nbhd
  - In high dim, few points in any nbhd

  everything is far away in high dim
- Consider example with *n* uniformly distributed points in $[-1,1]^d$
  - d=1: In [-0.1, 0.1], $\approx \frac{n}{10}$ obs. in interval
  - d=10 In $[-0.1, 0.1]^d$,

  roughly $n\left(\frac{0.2}{2}\right)^{10} = \frac{n}{10{,}000{,}000{,}000}$

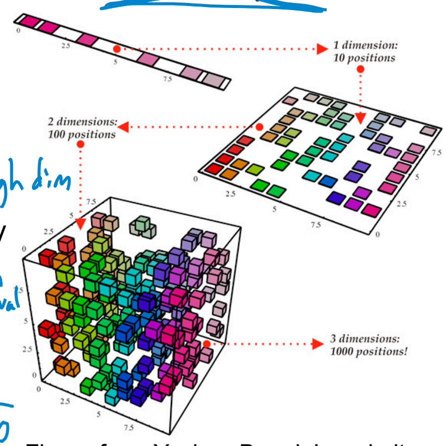


Figure from Yoshua Bengio's website

Natural Thin Plate Splines

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

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

- Solution: ***natural thin plate spline*** with knots at the x_{ij}
- For general λ, solution is a linear basis expansion of the form

$$f(x) = \beta_0 + \beta^T x + \sum_{j=1}^{n} b_j h_j(x)$$

with

$$h_j(x) = ||x - x_j||^2 \log ||x - x_j||$$

RBF

- Interpretation: We take an elastic flat plate that interpolates points (x_i, y_i) and penalize its "bending energy"

Tensor Product Splines

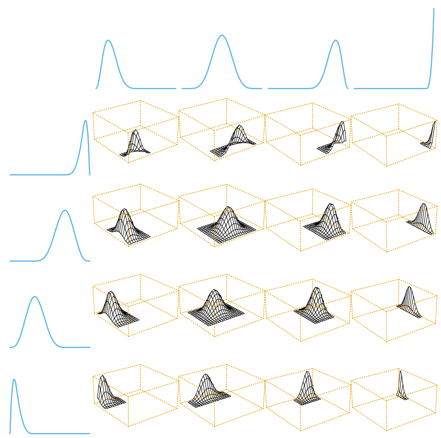
- We use this tensor product basis

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

to model $f(x)$

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

- This formulation extends (in theory) to any dimension d
- Note that 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:

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

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

  Fix: Constrain $\sum_{i=1}^{n} f_j(x_{ij}) = 0$

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

  many, many choices here (see all of module 2)

  or GPs...

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; take avg., then fix

2. Cycle: $j = 1, 2, \ldots, p, \ldots, 1, 2, \ldots, p, \ldots,$

$$\hat{f}_j \leftarrow \mathcal{S}_j\left[\{y_i - \hat{\alpha} - \sum_{k \neq j} \hat{f}_k(x_{ik})\}_1^N\right],$$

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

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

partial res. — smoother chosen for x_j — fit using partial res. — numerical reasons

From Hastie, Tibshirani, Friedman book

Other GAM formulations

- Semiparametric models:

  $g(\mu) = X^T\beta + \alpha + f(z)$ model linearly; model nonparam.

- ANOVA decompositions:

  $f(x) = \alpha + \sum_j f_j(x_j) + \sum_{j<k} f_{jk}(x_j, x_k) + \ldots$ main effects; capture interactions

  Choice of:
  - Maximum order of interaction
  - Which terms to include – maybe not all main effects + interactions
  - What representation

  – reg. splines + tensor product for interaction or thin plate ...

- 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

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

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

Module 4: Coping with Multiple Predictors

Multidimensional Kernel Methods

STAT/BIOSTAT 527, University of Washington

Emily Fox

May 14th, 2013

Nadaraya-Watson Estimator

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

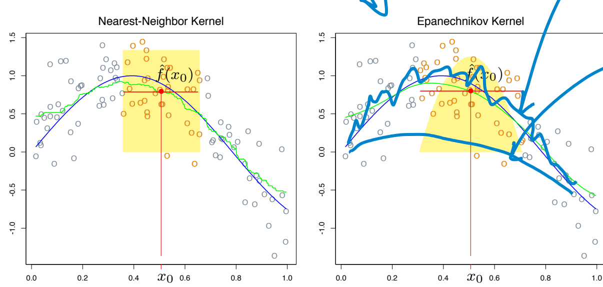
- Example:
  - Boxcar kernel → local avgs
  - Epanechnikov
  - Gaussian

  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:

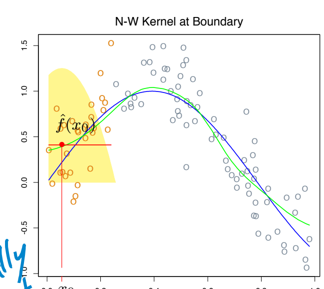
$\hat{f} = \arg\min_a \sum (y_i - a)^2$

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

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

$\hat{f}(x) = \arg\min_a \sum w_i(x)(y_i - a)^2$ ← 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 target x_0 :

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

- Minimize:

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

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

  Note: not equivalent to fitting a 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

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

- 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:

  $$\min_{\beta_{x_0}} (y - X_{x_0}\beta_{x_0})^T W_{x_0} (y - X_{x_0}\beta)$$

  $$X_{x_0} = \begin{bmatrix} 1 & x_1 - x_0 & \dots & \frac{(x_1-x_0)^d}{d!} \\ \vdots & & & \\ 1 & x_n - x_0 & \dots & \frac{(x_n-x_0)^d}{d!} \end{bmatrix}$$

- 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

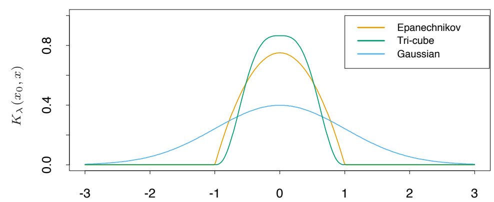
- Kernel smoothing and local regression extend straightforwardly to the multivariate *x* scenario

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

  - Need *d*-dimensional kernel
  - Nadaraya-Watson kernel smoother fits locally constant model
  - Local linear regression fits local hyperplane via weighted LS
  - …
- Challenges:
  - Defining kernel
  - Curse of dimensionality

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

$$K_\lambda(x_0, x) =$$

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

Multivariate Kernels

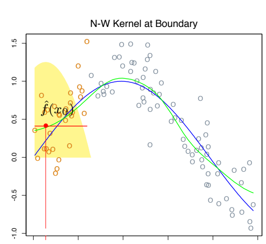
- Many choices, even more than in 1d
- Examples:
  - Product kernels

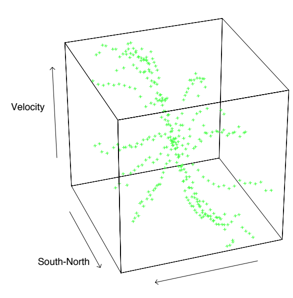
$$K_{\lambda_1,\lambda_2}(x_0, x) =$$

- 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

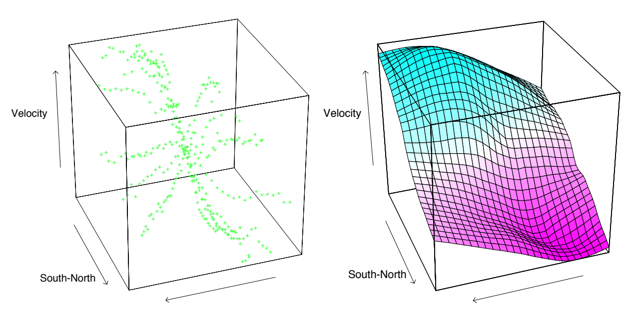
- Assume a RBF kernel
- For each target location x_0, goal is to minimize

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

- Equivalently,
- 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:

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

$$f(x) \approx f(x_0) + (x_1 - x_{01})\frac{\partial f}{\partial x_{01}} + (x_2 - x_{02})\frac{\partial f}{\partial x_{02}}$$
$$+ (x_1 - x_{01})^2 \frac{1}{2}\frac{\partial^2 f}{\partial x_{01}^2} + (x_1 - x_{01})(x_2 - x_{02})\frac{1}{2}\frac{\partial^2 f}{\partial x_{01}\partial x_{02}} + (x_2 - x_{02})^2 \frac{1}{2}\frac{\partial^2 f}{\partial x_{02}^2}$$

- Suggests the use of a local polynomial:
- 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
- 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
- The volume fraction is therefore:

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

- 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

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

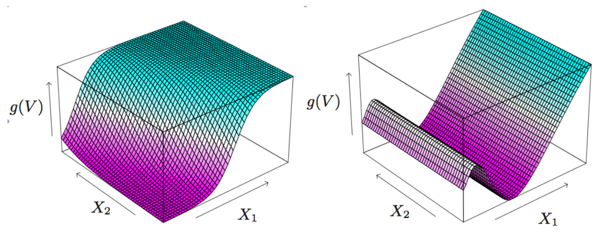
- Choices for A
  - Diagonal →
  - Low rank →
  - General

Projection Pursuit Regression

- To help deal with high-dimensional regression, consider

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

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



PPR Comments

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

- 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 }

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

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 $\hat{\alpha} = \text{avg}(y_i)$
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

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

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

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

- 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

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

  or, more simply, standard GAMs

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

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

Varying Coefficient Models

- Special case of a structured model
- Divide the set of *d* covariates into two sets
- Consider a ***conditionally linear*** model

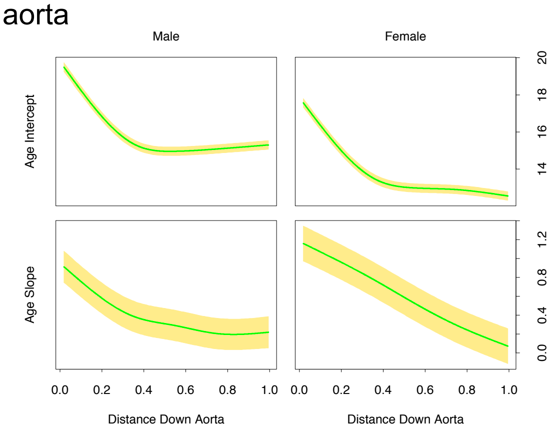
$$f(x) =$$

- Due to its local nature, it's natural to fit such a model using locally weighted LS

$$\min_{\alpha(z_0), \beta(z_0)} \sum_{i=1}^{n} K_\lambda(z_0, z_i)(y_i - \alpha(z_0) - x_{1i}\beta_1(z_0) - \cdots - x_{qi}\beta_q(z_0))^2$$

Varying Coefficient Models

- Example = Human aorta data
- Response = diameter of aorta
- Covariates
  - Linear in "age"
  - Coefficients vary in "gender" and "depth"
- Separate model for M/F
- Results:
  - Aorta thickens with age
  - Relationship is less clear for larger depth



From Hastie, Tibshirani, Friedman book

Varying Coefficient Models

- Alternatively, one can use splines instead of local regression as a smoother for the varying coefficient functions $\beta_j(z)$
- Consider penalized linear splines with *L* knots
  - For univariate *x* and *z*, for simplicity, we have

$$E[y \mid x, z] = \alpha_0^{(0)} + \alpha_1^{(0)} z + \sum_{\ell=1}^{L} b_\ell^{(0)} (z - \xi_\ell)_+$$
$$+ \left(\alpha_0^{(1)} + \alpha_1^{(1)} z + \sum_{\ell=1}^{L} b_\ell^{(1)} (z - \xi_\ell)_+ \right) x$$

Example: Time-Varying Coeff

- Let *z* correspond to time *t*, a simple case being:

$$y_t =$$

- This model directly relates to (Bayesian) dynamic linear models

$$y_t = \alpha + z_t\beta_t + \epsilon_t \quad \epsilon_t \sim N(0, \sigma_\epsilon^2)$$
$$\beta_t = \beta_{t-1} + \nu_t \quad v_t \sim N(0, \sigma_\nu^2)$$

See West and Harrison 1997

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? $\hat{p} = \frac{1}{n}\sum \delta_{x_i}$

$\hat{p}$

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

$$\hat{p}(x_0) = \frac{\# x_i \in Nbhd(x_0)}{n\lambda}$$

λ ← 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

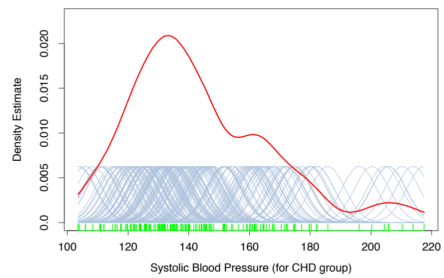
- Popular choice = Gaussian kernel → ***Gaussian KDE***

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

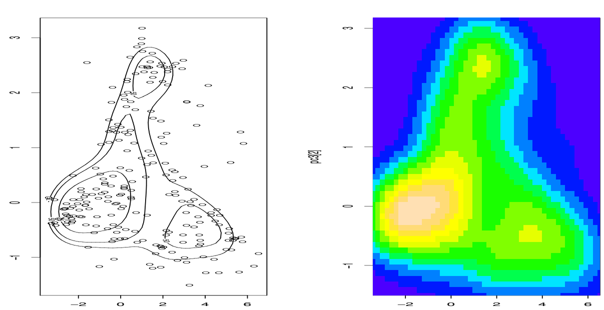
Multivariate KDE

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

- 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*

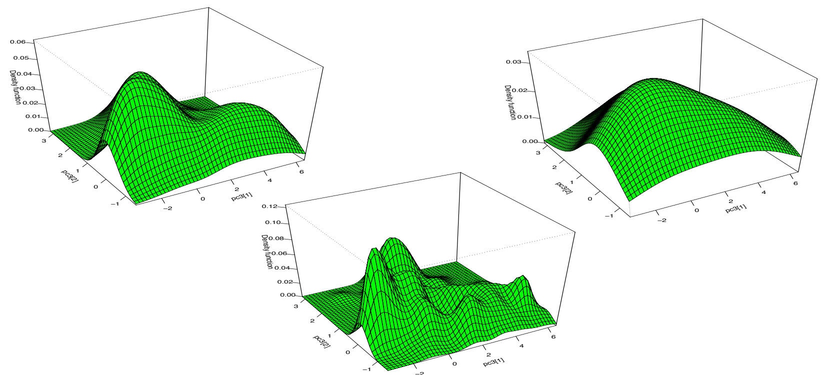
Multivariate KDE Example

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



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



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