STK-IN4300
Statistical Learning Methods in Data Science

Riccardo De Bin

debin@math.uio.no

Outline of the lecture

- Kernel Smoothing Methods
  - One dimensional kernel smoothers
  - Selecting the width of a kernel
  - Local linear regression
  - Local polynomial regression
  - Local regression in $\mathbb{R}^p$
  - Structured local regression models in $\mathbb{R}^p$
  - Kernel density estimation
  - Mixture models for density estimation
- Nonparametric Density Estimation with a Parametric Start

One dimensional kernel smoothers: from kNN to kernel smoothers

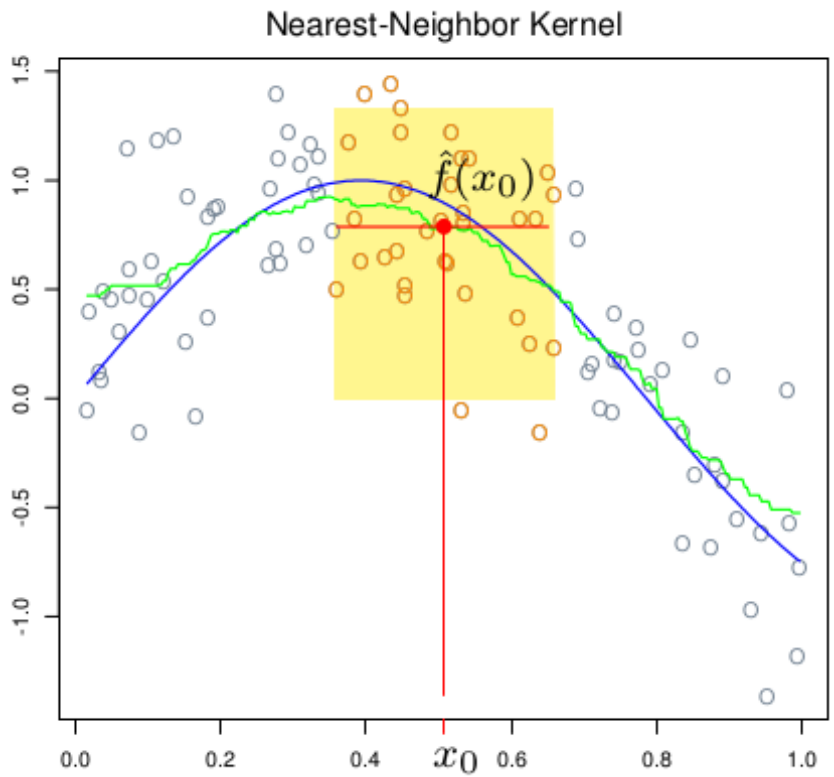
When we introduced the kNN algorithm,

$$\hat{f}(x) = \text{Ave}(y_i | x_i \in N_k(x))$$

- justified as an estimate of $E[Y|X = x]$.

Drawbacks:

- ugly discontinuities;
- same weight to all points despite their distance to x.



One dimensional kernel smoothers: definition

Alternative: weight the effect of each point based on its distance.

$$\hat{f}(x_0) = \frac{\sum_{i=1}^{N} K(x_0, x_i) y_i}{\sum_{i=1}^{N} K(x_0, x_i)},$$

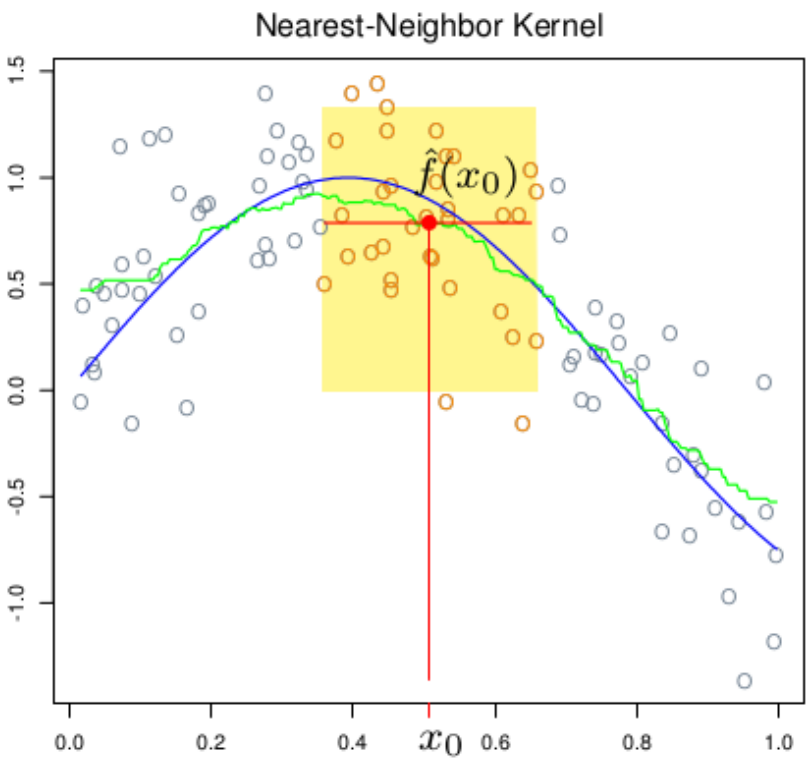
where

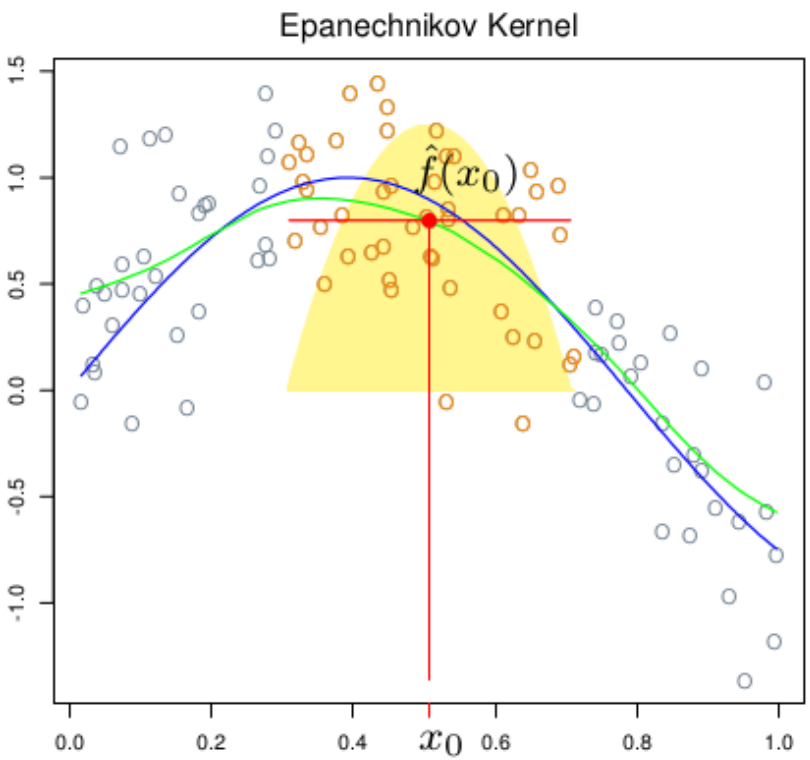
$$K_\lambda(x_0, x) = D\left(\frac{|x - x_0|}{\lambda}\right). \tag{1}$$

Here:

- $D(\cdot)$ is called kernel;
- λ is the bandwidth or smoothing parameter.

One dimensional kernel smoothers: comparison





One dimensional kernel smoothers: typical kernels

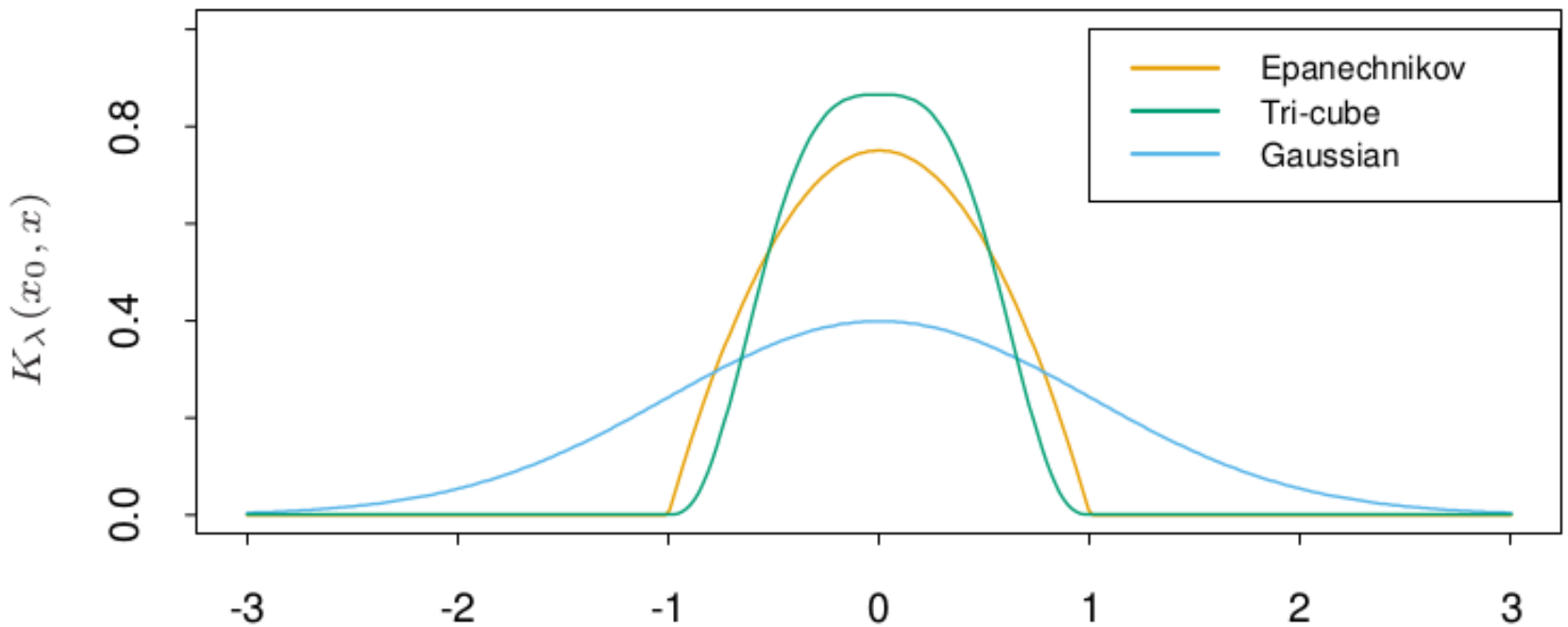
We need to choose $D(\cdot)$:

- symmetric around x_0;
- goes off smoothly with the distance.

Typical choices:

Nucleus	D(t)	Support
Normal	$\frac{1}{\sqrt{2\pi}} \exp\{-\frac{1}{2}t^2\}$	$\mathbb{R}$
Rectangular	$\frac{1}{2}$	$(-1, 1)$
Epanechnikov	$\frac{3}{4}(1-t^2)$	$(-1, 1)$
Biquadratic	$\frac{15}{16}(1-t^2)^2$	$(-1, 1)$
Tricubic	$\frac{70}{81}(1-\lvert t\rvert^3)^3$	$(-1, 1)$

One dimensional kernel smoothers: comparison

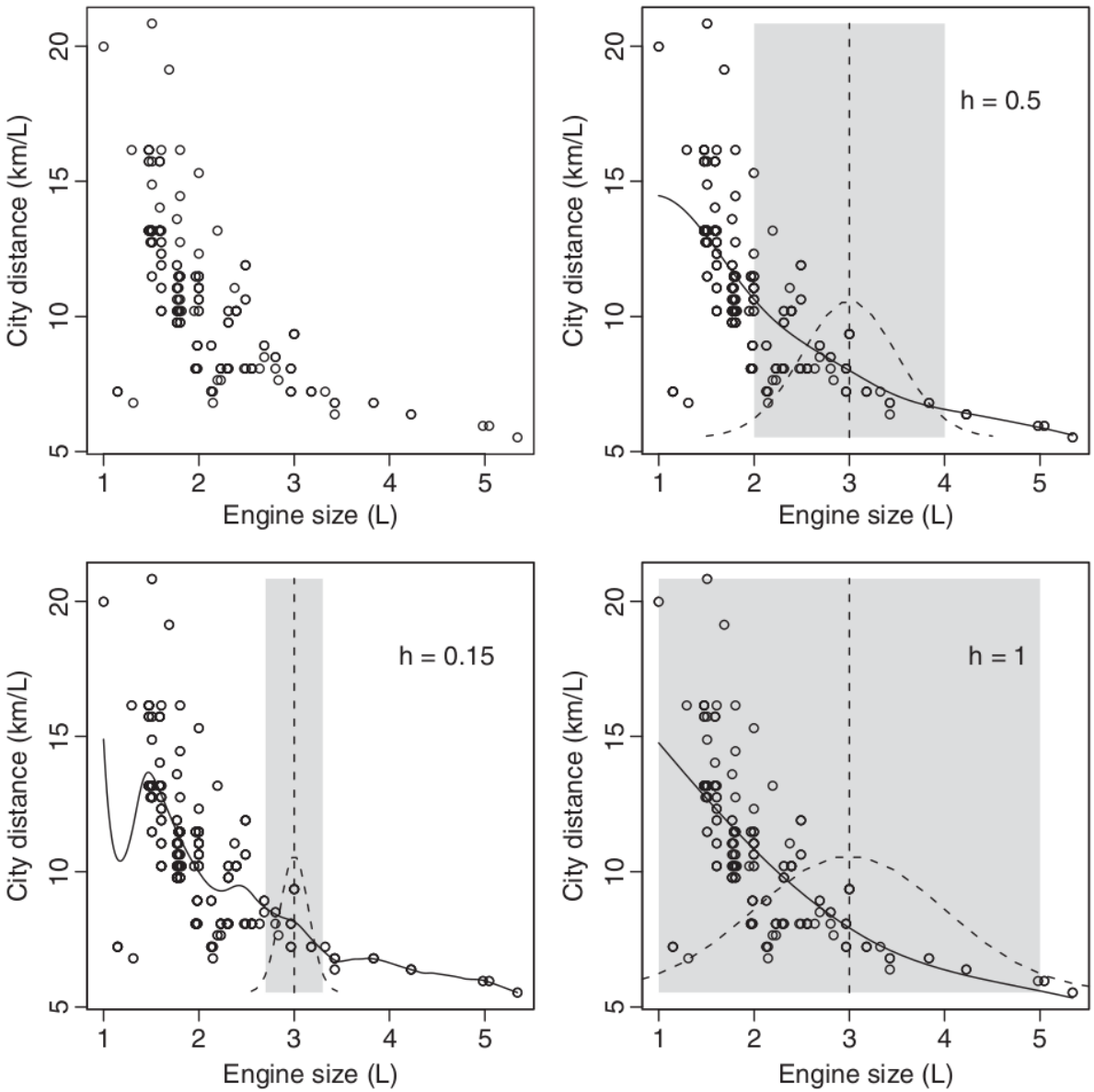


One dimensional kernel smoothers: choice of the smoothing parameter

Choice of the bandwidth λ:

- controls how large is the interval around x_0 to consider,
  - for Epanechnikov, biquadratic or tricubic kernels $\rightarrow$ radius of the support;
  - for Gaussian kernel, standard deviation;
- large values implies lower variance but higher bias,
  - λ small $\rightarrow$ $\hat{f}(x_0)$ based on few points $\rightarrow$ y_i's closer to y_0;
  - λ large $\rightarrow$ more points $\rightarrow$ stronger effect of averaging;
- alternatively,
  - adapt to the local density (fix k as in kNN);
  - expressed by substituting λ with $h_\lambda(x_0)$ in (1);
  - keep bias constant, variance is inversely proportional to the local density.

One dimensional kernel smoothers: effect of the smoothing parameter



Selecting the width of a kernel: bias and variance

Assume $y_i = f(x_i) + \epsilon_i$, ϵ_i i.i.d. s.t. $E[\epsilon_i] = 0$ and Var $= \sigma^2$, then

$$E[\hat{f}(x)] \approx f(x) + \frac{\lambda^2}{2}\sigma_D^2 f''(x)$$

and

$$\text{Var}[\hat{f}(x)] \approx \frac{\sigma^2}{N\lambda}\frac{R_D}{g(x)}$$

for N large and λ sufficiently close to 0 (**?**).
Here:

- $\sigma_D^2 = \int t^2 D(t)dt$;
- $R_D = \int D(t)^2 dt$;
- $g(x)$ is the density from which the x_i were sampled.

Selecting the width of a kernel: bias and variance

Note:

- the bias is a multiple of λ^2;
  - $\lambda \to 0$ reduce the bias;
- the variance is a multiple of $\frac{1}{N\lambda}$;
  - $\lambda \to \infty$ reduce the variance.

The quantities $g(x)$ and $f''(x)$ are unknown, otherwise

$$\lambda_{\text{opt}} = \left(\frac{\sigma^2 R_D}{\sigma_D^4 f''(x)^2 g(x) N} \right)^{1/5};$$

note that λ must tend to 0 with rate $N^{-1/5}$ (i.e., very slowly).

Selecting the width of a kernel: AIC

Anyway, local smoothers are linear estimators,

$$\hat{f}(x) = S_\lambda y$$

as S_λ, the smoothing matrix, does not depend on y.

Therefore, an Akaike Information Criterion can be implemented,

$$AIC = \log \hat{\sigma} + 2 \text{ trace}\{S_\lambda\}$$

where trace$\{S_\lambda\}$ are the effective degrees of freedom.

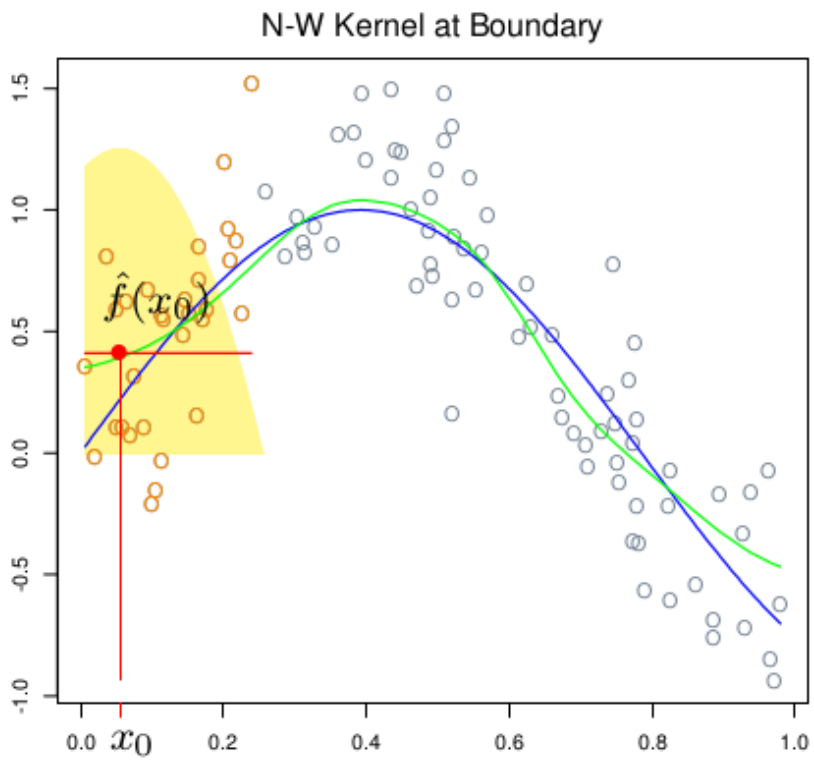
Otherwise it is always possible to implement a cross-validation procedure.

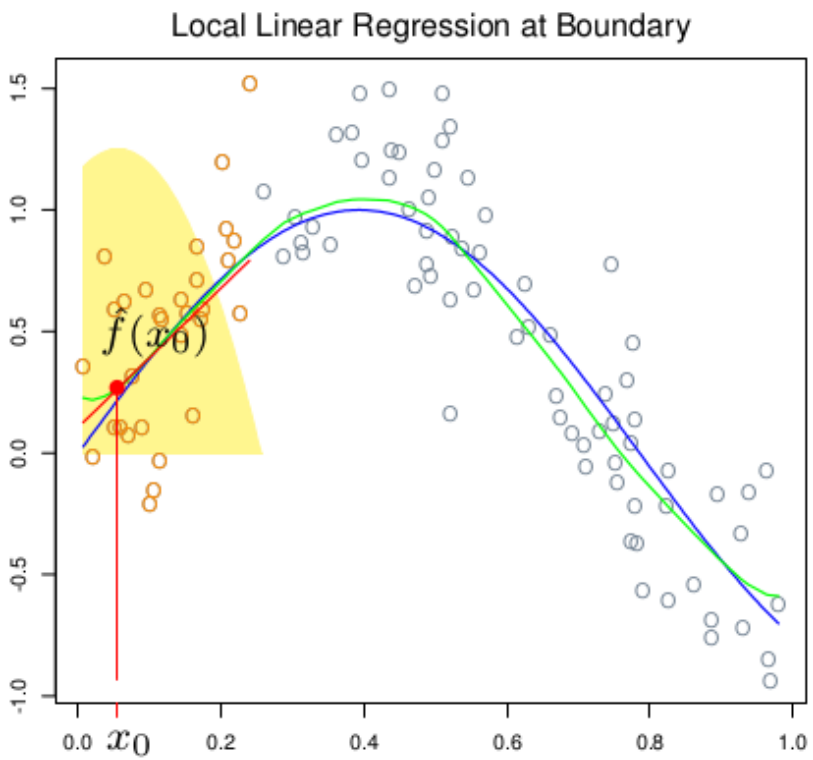
One dimensional Kernel Smoothers: other issues

Other points to consider:

- boundary issues:
  - estimates are less accurate close to the boundaries;
  - less observations;
  - asymmetry in the kernel;
- ties in the x_i's:
  - possibly more weight on a single x_i;
  - there can be different y_i for the same x_i.

Local linear regression: problems at the boundaries





Local linear regression: problems at the boundaries

By fitting a straight line, we solve the problem to the first order.

$\downarrow$

Local linear regression

Locally weighted linear regression solves, at each target point x_0,

$$\min_{\alpha(x_0),\beta(x_0)} \sum_{i=1}^{N} K_\lambda(x_0, x_i)[y_i - \alpha(x_0) - \beta(x_0)x_i]^2.$$

The estimate is $\hat{f}(x_0) = \hat{\alpha}(x_0) + \hat{\beta}(x_0)x_0$:

- the model is fit on all data belonging to the support of K_λ;
- it is only evaluated in x_0.

Local linear regression: estimation

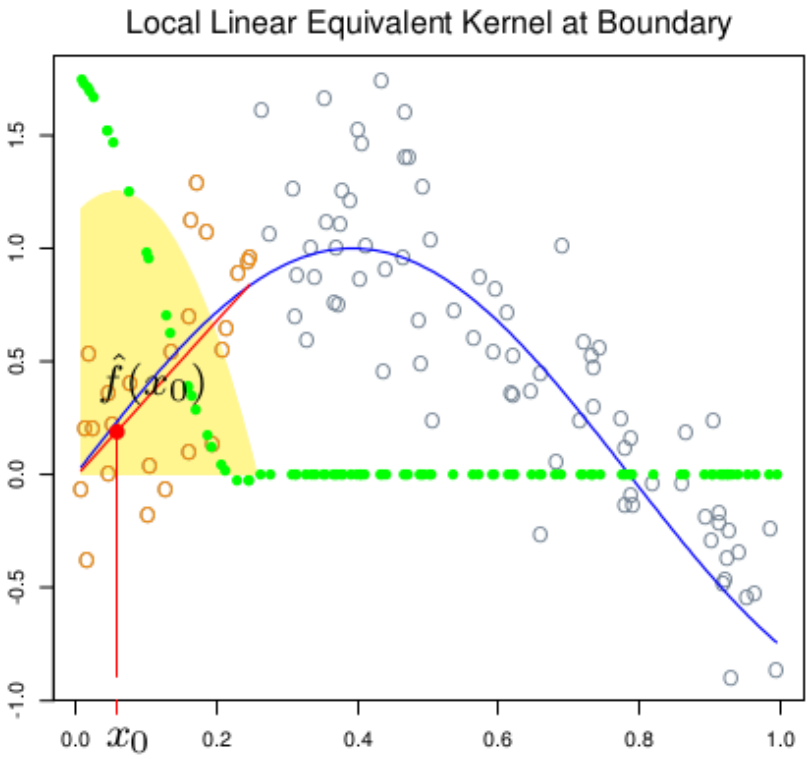
Estimation

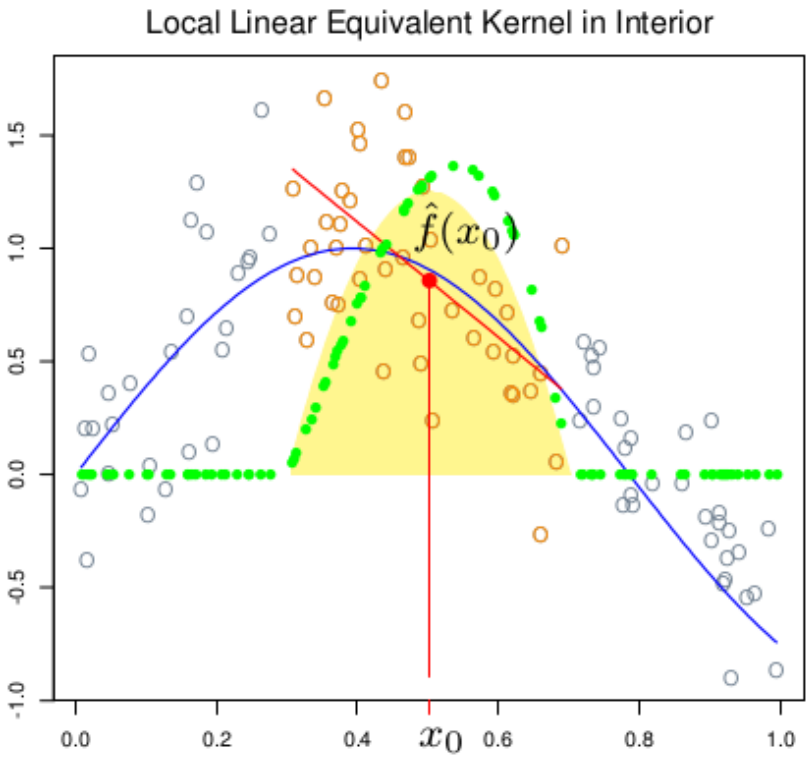
$$\begin{aligned}\hat{f}(x_0) &= b(x_0)^T (B^T W(x_0) B)^{-1} B^T W(x_0) y \\ &= \sum_{i=1}^{N} l_i(x_0) y_i,\end{aligned}$$

where:

- $b(x_0)^T = (1, x_0)$
- $B = (\vec{1}, X)$;
- $W(x_0)$ is a $N \times N$ diagonal matrix with i-th term $K_\lambda(x_0, x_i)$;
- $\hat{f}(x_0)$ is linear in y ($l_i(x_0)$ does not depend on y_i);
- the weights $l_i(x_0)$ are sometimes called equivalent kernels,
  - combine the weighting kernel $K_\lambda(x_0, \cdot)$ and the LS operator.

Local linear regression: bias correction due asymmetry





Local linear regression: bias

Using a Taylor expansion of $f(x_i)$ around x_0,

$$
\begin{aligned}
E[\hat{f}(x_0)] &= \sum_{i=1}^{N} l_i(x_0) f(x_i) \\
&= f(x_0) \sum_{i=1}^{N} l_i(x_0) + f'(x_0) \sum_{i=1}^{N} (x_i - x_0) l_i(x_0) + \\
&\quad + \frac{f''(x_0)}{2} \sum_{i=1}^{N} (x_i - x_0)^2 l_i(x_0) + \dots \qquad (2)
\end{aligned}
$$

For local linear regression,

- $\sum_{i=1} l_i(x_0) = 1$;
- $\sum_{i=1}^{N} (x_i - x_0) l_i(x_0) = 0$.

Therefore,

- $E[\hat{f}(x_0)] - f(x_0) = \frac{f''(x_0)}{2} \sum_{i=1}^{N} (x_i - x_0)^2 l_i(x_0) + \dots$.

Local polynomial regression: bias

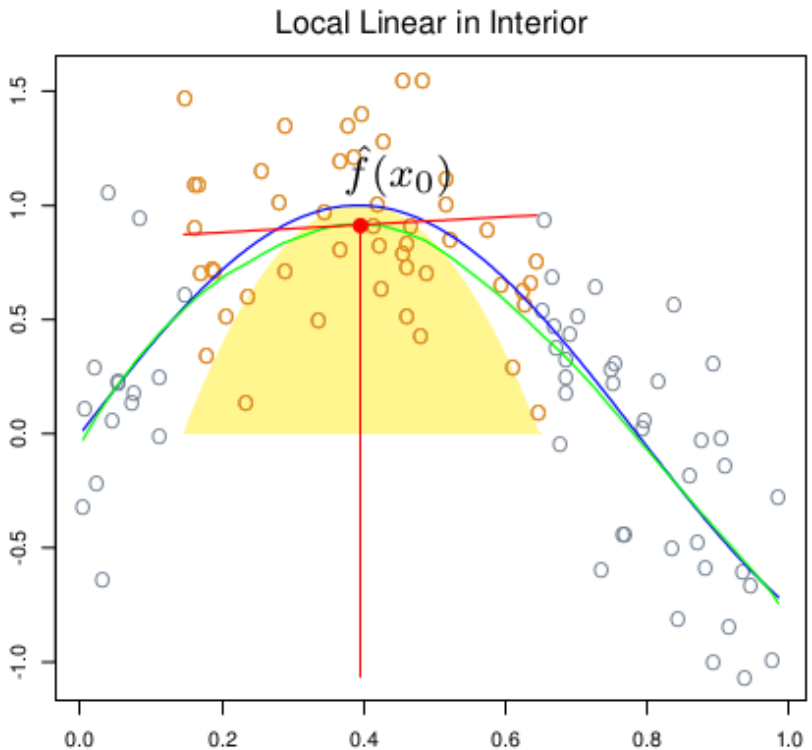
Why limiting to a linear fit?

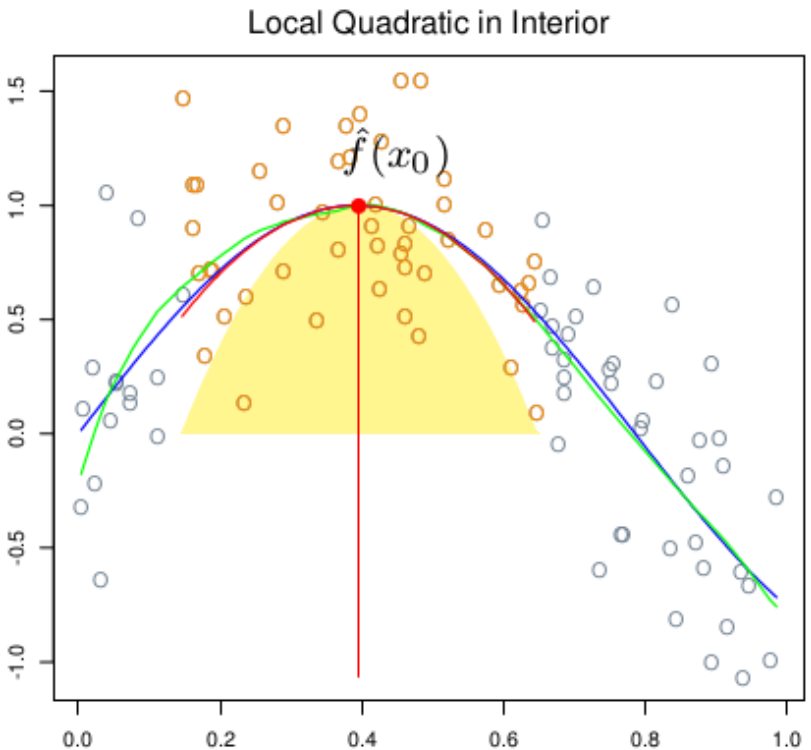
$$\min_{\alpha(x_0),\beta_1(x_0),\ldots,\beta_d(x_0)} \sum_{i=1}^{N} K_\lambda(x_0, x_i) \left[y_i - \alpha(x_0) - \sum_{j=1}^{d} \beta_j(x_0) x_i^j \right]^2,$$

with solution $f(\hat{x_0}) = \hat{\alpha}(x_0) + \sum_{j=1}^{d} \hat{\beta}(x_0) x_0^j$.

- it can be shown that the bias, using (2), only involves components of degree $d + 1$;
- in contrast to local linear regression, it tends to be closer to the true function in regions with high curvature,
  - no *trimming the hills and filling the gaps* effect.

Local polynomial regression: regions with high curvature





Local polynomial regression: bias-variance trade-off

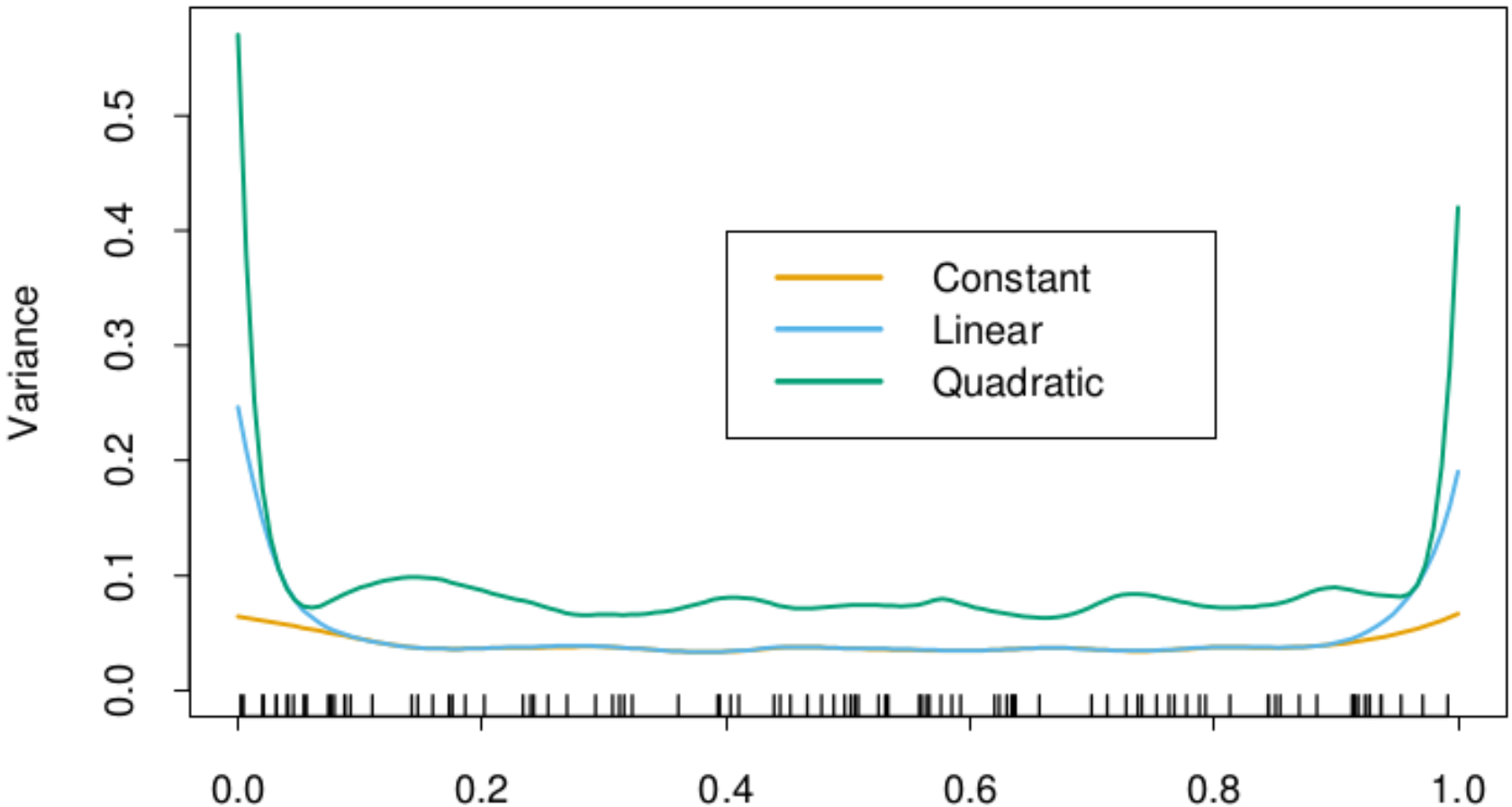
Not surprisingly, there is a price for having less bias.

Assuming a model $y_i = f(x_i) + \epsilon_i$, where ϵ_i are i.i.d. with mean 0 and variance σ^2,

$$\text{Var}(\hat{f}(x_i)) = \sigma^2 ||l(x_0)||$$

It can be shown that $||l(x_0)||$ increase with $d \Rightarrow$ bias-variance trade-off in the choice of d.

Local polynomial regression: variance



Local polynomial regression: final remarks

Some final remarks:

- local linear fits help dramatically in alleviating boundary issues;
- quadratic fits do a little better, but increase variance;
- quadratic fits solve issues in high curvature regions;
- asymptotic analyses suggest that polynomials of odd degrees should be preferred to those of even degrees,
  - the MSE is asymptotically dominated by boundary effects;
- anyway, the choice of d is problem specific.

Local regression in $\mathbb{R}^p$: extension

Kernel smoothing and local regression can be easily generalized to more dimensions:

- average weighted by a kernel with support in $\mathbb{R}^p$;
- for local regression, fit locally an hyperplane.

With $d = 1$ and $p = 2$,

- $b(X) = (1, X_1, X_2)$

At each x_0, solve

With $d = 2$ and $p = 2$,

- $b(X) = (1, X_1, X_2, X_1^2, X_2^2, X_1X_2)$

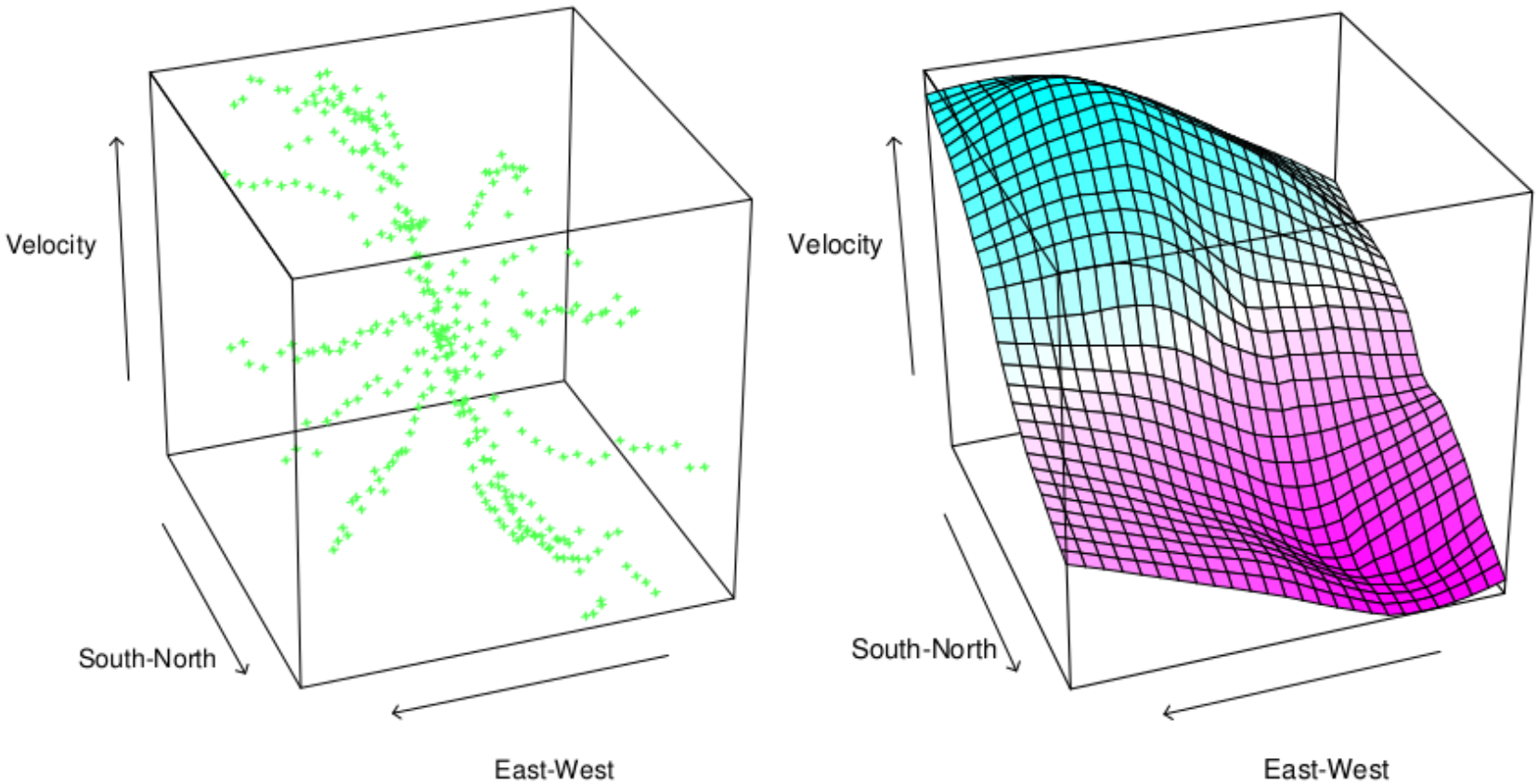
$$\min_{\beta(x_0)} \sum_{i=1}^{N} K_\lambda(x_0, x_i)[y_i - b(x_i)^T \beta(x_0) x_i]^2.$$

where $K_\lambda(x_0, x_i)$ is a radial function,

$$K_\lambda(x_0, x_i) = D\left(\frac{||x - x_0||}{\lambda}\right).$$

Since $||\cdot||$ is the Euclidean norm, standardize each x_j.

Local regression in $\mathbb{R}^p$: example



Local regression in $\mathbb{R}^p$: remarks

Some remarks:

- boundary issues are even more dramatic than in one dimension;
  - the fraction of points at the boundary increases to 1 by increasing the dimensions;
  - curse of dimensionality;
- local polynomials still perform boundary corrections up to the desired order;
- local regression does not really make sense for $p > 3$,
  - it is impossible to maintain localness (small bias) and sizeable sample in the neighbourhood (small variance);
  - again, curse of dimensionality.

Structured local regression models in $\mathbb{R}^p$: structured kernels

When the ratio dimensions/sample size is too large:

Structured kernels

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

- A is a matrix semidefinite positive;
- we can add structures through A:
  - A diagonal, increase or decrease the importance of the predictor X_j by increasing/decreasing a_{jj};
  - low rank versions of $A \rightarrow$ projection pursuit;

Structured local regression models in $\mathbb{R}^p$: structured regression functions

Structured regression functions

$$f(X_1, \ldots, X_p) = \alpha + \sum_{j=1}^{p} g_j(X_j) + \sum_{k<\ell} g_{k\ell}(X_k, X_\ell) + \ldots$$

- we can simplify the structure;
- examples:
  - remove all interaction terms,
    $f(X_1, \ldots, X_p) = \alpha + \sum_{j=1}^{p} g_j(X_j)$;
  - keep only the first order interactions,
    $f(X_1, \ldots, X_p) = \alpha + \sum_{j=1}^{p} g_j(X_j) + \sum_{k<\ell} g_{k\ell}(X_k, X_\ell)$;
  - ...

Structured local regression models in $\mathbb{R}^p$: varying coefficient models

Varying coefficient models

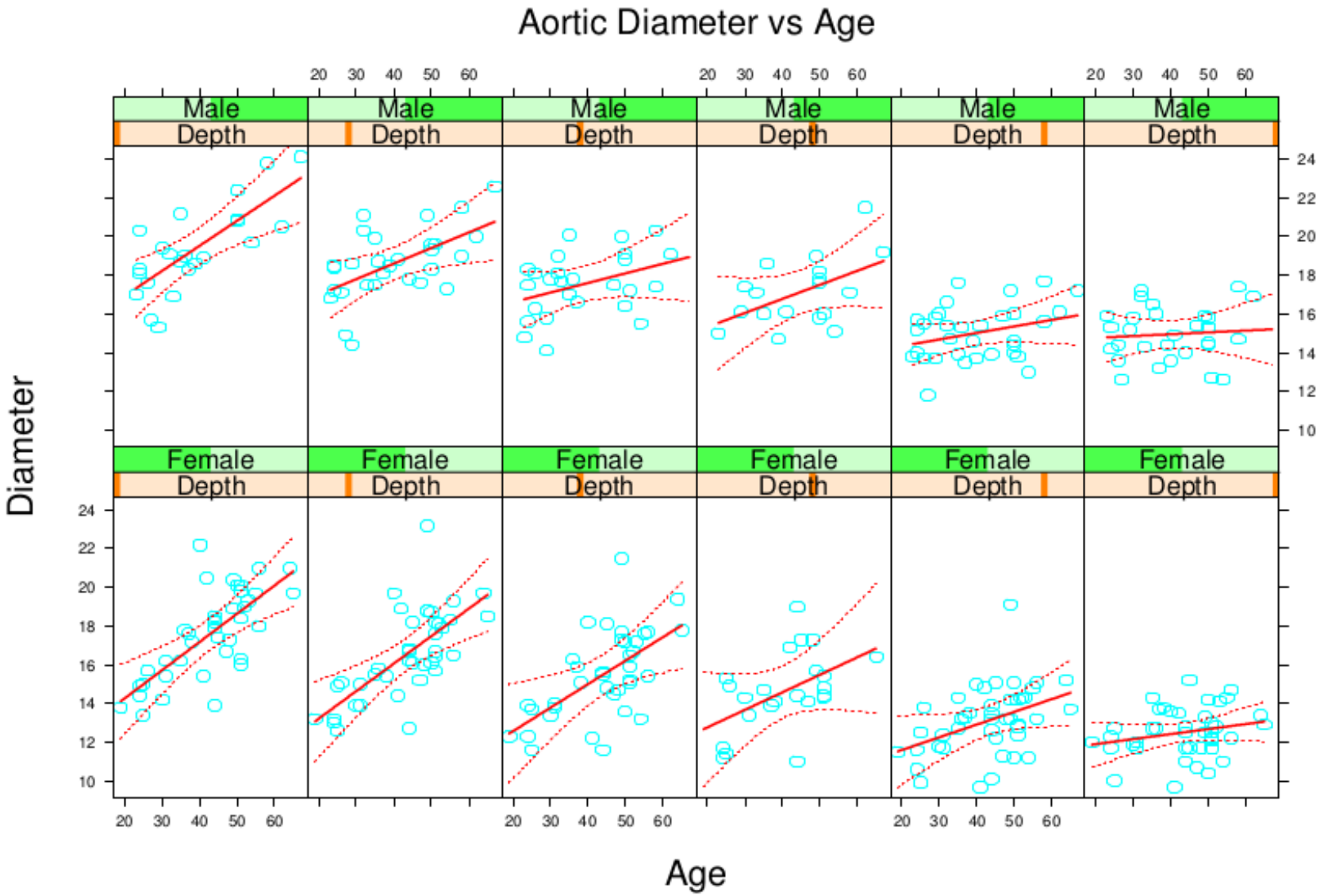
The varying coefficient models:

- are a special case of structured regression functions;
- consider only $q < p$ predictors, all the remaining are in Z;
- assume the conditionally linear model,

$$f(X) = \alpha(Z) + \beta_1(Z)X_1 + \cdots + \beta_q(Z)X_q;$$

- given Z, it is a linear model,
  - solution via least squares estimator;
- the coefficients can vary with Z.

Structured local regression models in $\mathbb{R}^p$: structured regression functions



Kernel density estimation: density estimation

Suppose to have a random sample, $x_i \in \mathbb{R}$, $i = 1, \ldots, N$, and want to estimate its density $f_X(x)$. An estimation at each point x_0 is

$$\hat{f}_X(x_0) = \frac{\#x_i \in \mathcal{N}(x_0)}{N\lambda}.$$

where $\mathcal{N}(x_0)$ is a small metric neighborhood around x_0 of width λ.

Bumpy estimate $\rightarrow$ the **smooth Parzen estimate** is preferred,

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

in which closer observations contributes more.

Kernel density estimation: choice of $K_\lambda(x_0, x)$

For the smooth Parzen estimate, the Gaussian kernel is often used,

$$K_\lambda(x_0, x) = \phi\left(\frac{|x - x_0|}{\lambda}\right)$$

where ϕ is the density of a standard normal.

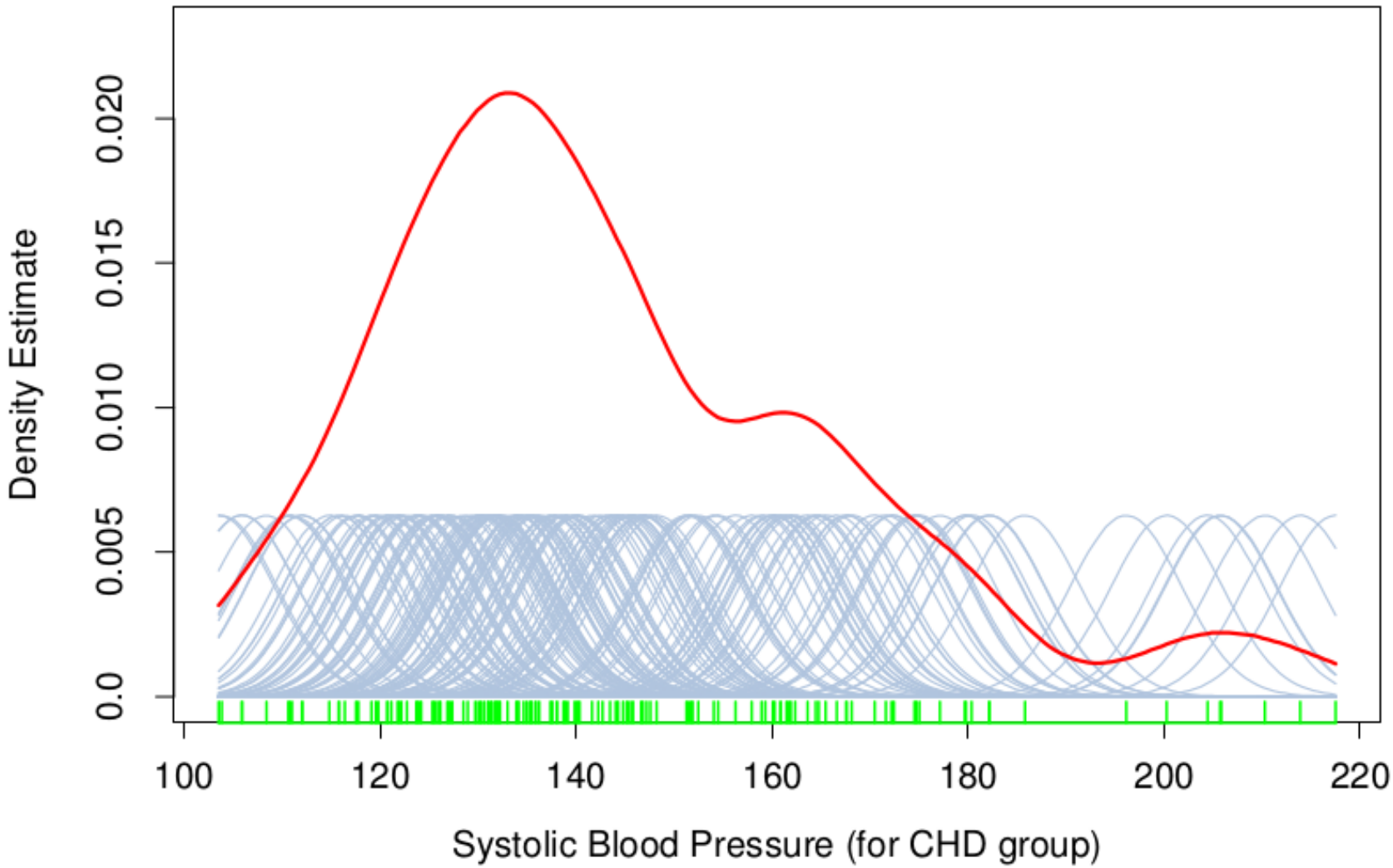
Using the density of a normal with mean 0 and sd λ, denoted ϕ_λ,

$$\begin{aligned} f_X(x) &= \frac{1}{N}\sum_{i=1}^{N} \phi_\lambda(x - x_i) \\ &= (\hat{F} \star \phi_\lambda)(x) \end{aligned}$$

the convolution of the sample empirical distribution $\hat{F}$ with ϕ_λ:

- smooth $\hat{F}$ by adding independent Gaussian noise to each x_i.

Kernel density estimation: example



Mixture models for density estimation: density estimation

The density $f(X)$ can be considered a mixture of distributions,

$$f(X) = \sum_{m=1}^{M} \alpha_m g(x; \mu_m, \Sigma_m)$$

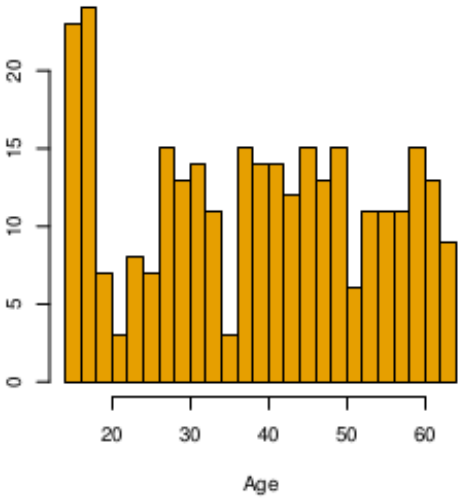
where

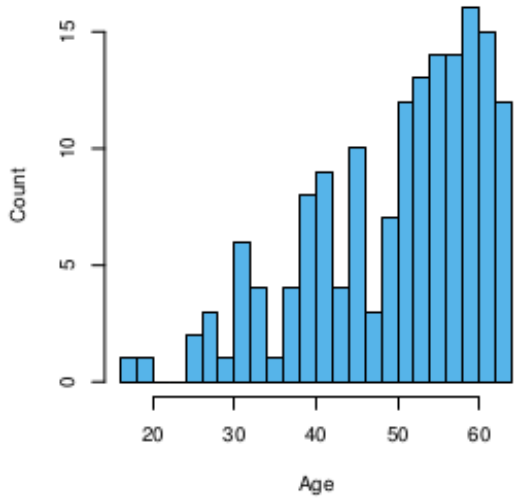
- α_m are the mixing proportions, $\sum_{m=1}^{M} \alpha_m = 1$;
- each density $g(\cdot)$ has mean μ_m ad covariance Σ_m;
- almost always $g(x; \mu_m, \Sigma_m) = \phi(x; \mu_m, \Sigma_m)$;

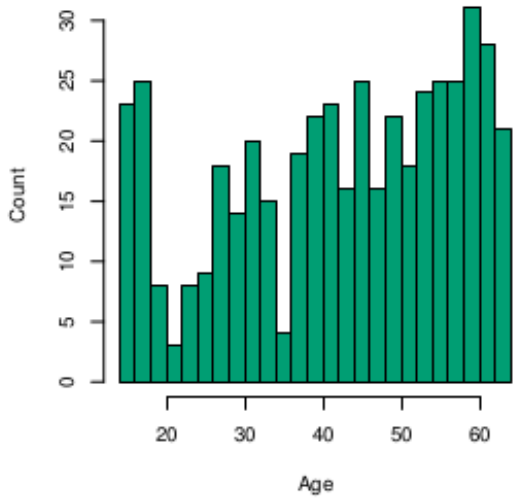
$$\downarrow$$

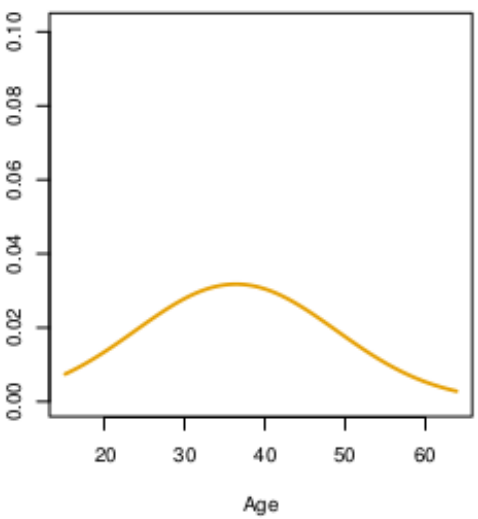
- **Gaussian mixture model**.

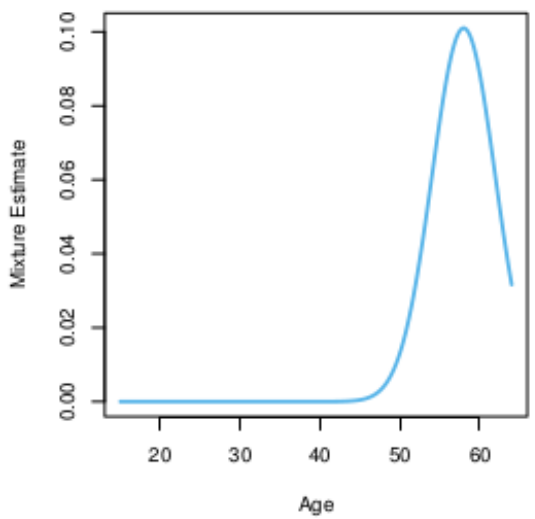
Mixture models for density estimation: example

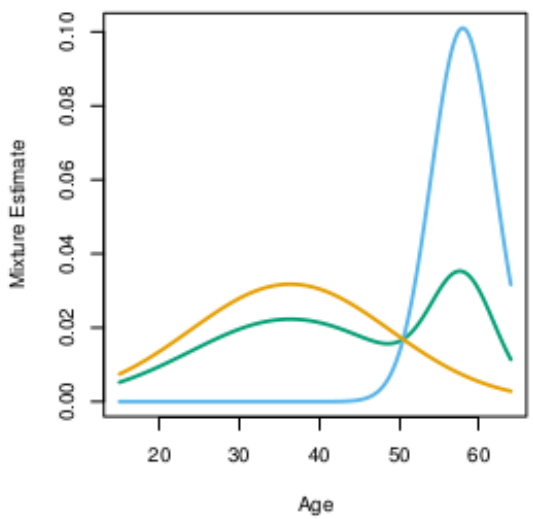












Semiparametric density estimation: ?

? proposed a different option:

- start with a parametric density estimate $f_0(x, \hat{\theta})$;
- multiply it to a correction term $r(x) = f(x)/f_0(x, \hat{\theta})$;
- estimate the correction term with a kernel smoother,

$$\hat{r}(x) = \frac{1}{N} \sum_{i=1}^{N} \frac{K_\lambda(x, x_i)}{f_0(x_i, \hat{\theta})};$$

- the resulting density estimate is

$$\hat{f}_{HG}(x) = f_0(x, \hat{\theta})\hat{r}(x) = \frac{1}{N} \sum_{i=1}^{N} K_\lambda(x, x_i) \frac{f_0(x, \hat{\theta})}{f_0(x_i, \hat{\theta})}.$$

Semiparametric density estimation: ?

Note that:

- the initial parametric estimate is not necessarily a good approximation to the true density:
  - the method often works well with "bad" parametric starts;
  - the better the approximation, the better the result, though;
- $f_0(x, \hat{\theta}) = \text{constant} \rightarrow f_0(x) \sim \text{Unif}$,
  - back to the classic kernel estimator.

Semiparametric density estimation: properties

Consider $f_{HG}(x)$'s variance,

$$\text{Var}(\hat{f}_{HG}(x)) = \text{Var}(\hat{f}_{\text{kernel}}(x)) + O\left(\frac{\lambda}{N} + \frac{1}{N^2}\right),$$

- $\hat{f}_{HG}(x)$ and $\hat{f}_{\text{kernel}}(x)$ have approximatively the same variance;

and bias,

$$E[\hat{f}_{HG}(x)] \approx f(x) + \frac{\lambda^2}{2}\sigma_D^2 f_0(x) r''(x),$$

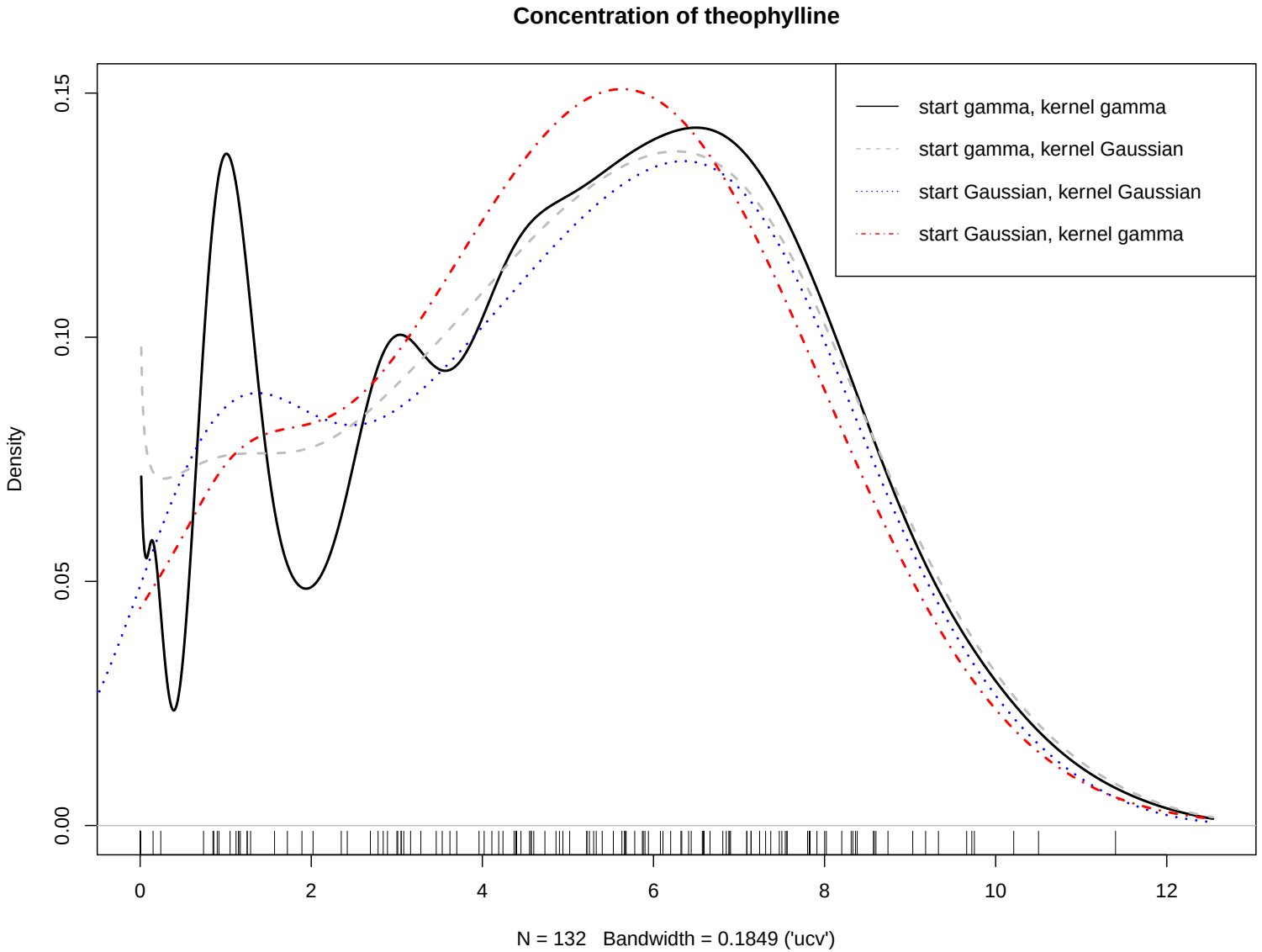
- same order as the bias of $\hat{f}_{\text{kernel}}(x)$, i.e., $O(\lambda^2)$;
- it is proportional to $f_0(x)r''(x)$ rather than $f''(x)$;
- smaller when $f''(x) = f_0''(x)r(x) + 2f_0'(x)r'(x) + f_0(x)r''(x)$,
  - when $f_0(x)$ is a good guess, better performance!

Semiparametric density estimation: derivation example

Example with a Gaussian start,

$$\begin{aligned}\hat{f}_{HG}(x) = f_0(x,\hat{\theta})\hat{r}(x) &= \frac{1}{N}\sum_{i=1}^{N} K_\lambda(x,x_i)\frac{f_0(x,\hat{\theta})}{f_0(x_i,\hat{\theta})} \\ &= \frac{1}{\hat{\sigma}}\phi\left(\frac{x-\hat{\mu}}{\hat{\sigma}}\right)\frac{\frac{1}{N}\sum_{i=1}^{N} K_\lambda(x,x_i)}{\phi\left(\frac{x_i-\hat{\mu}}{\hat{\sigma}}\right)} \\ &= \frac{1}{N}\sum_{i=1}^{N} K_\lambda(x,x_i)\frac{\exp\left\{-\frac{1}{2}\frac{x-\hat{\mu}}{\hat{\sigma}}\right\}}{\exp\left\{-\frac{1}{2}\frac{x_i-\hat{\mu}}{\hat{\sigma}}\right\}}.\end{aligned}$$

Semiparametric density estimation: data example



References I