CSSS 569 · Visualizing Data

EXPLORATORY DATA ANALYSIS: BETWEEN DATA & MODEL

Christopher Adolph

Department of Political Science

and

Center for Statistics and the Social Sciences

University of Washington, Seattle

Exploratory Data Analysis

Exploratory Data Analysis refers to a approach pioneered by statistician John Tukey

Emphasis on

- letting the data speak for itself
- non-parametric models
- tools for exploring high-dimensional datasets

Relatively little used in social science, where we prefer parametric models

Dangers: sometimes parametric models are fragile, and EDA could help show this

Without preliminary EDA, finding the right parametric specification may be harder

EDA: Some example techniques

The `lattice` package implements a set of EDA techniques pioneered by Bell Labs/Bill Cleveland.

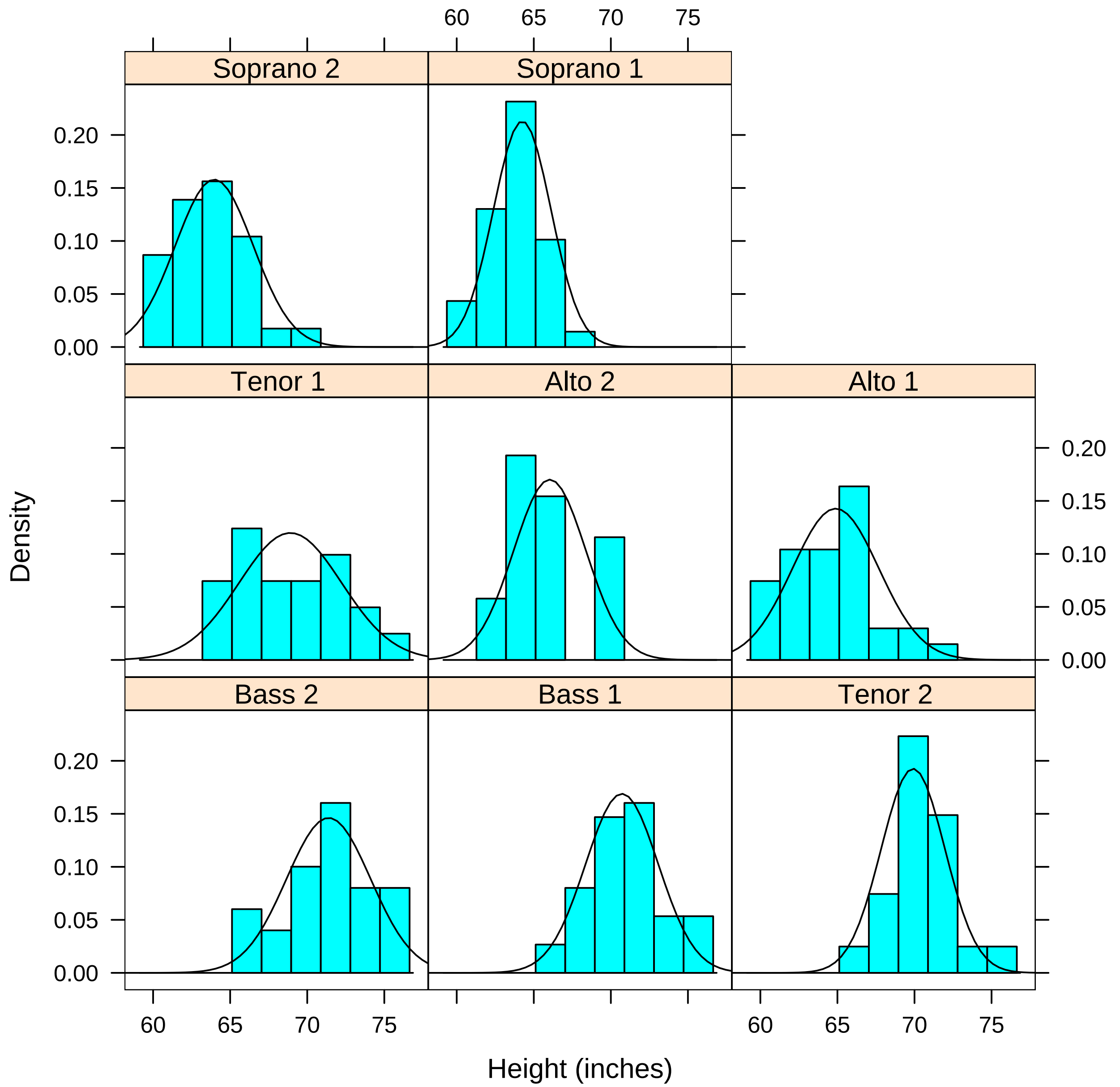
Basic idea: small multiples that show relations between x and y conditioning on z, and perhaps w, etc.

Lattice plots consist of multiple panels of plotted data

The panels are linked to strips which identify a conditioning variable

Let's see how this works with *histograms* and *scatterplots*

Lattice in action



Key lattice options

```
histogram( ~ height | voice.part, data = singer,
          xlab = "Height (inches)", type = "density",
          panel = function(x, ...) {
             panel.histogram(x, ...)
             panel.mathdensity(dmath = dnorm, col = "black",
                     args = list(mean=mean(x),sd=sd(x))
                              )
             }
          )

dev.off()
```

Notice two trademark elements of lattice:

- the use of a formula to input the data
- the presence of a customizable panel function

Lattice

Key parameters for lattice plots often hide in `panel.XXX()` where `XXX()` is the function of interest

Example: the key parameter for 3D plots (how to spin them) is `screen`, which is documented in `panel.cloud()` only

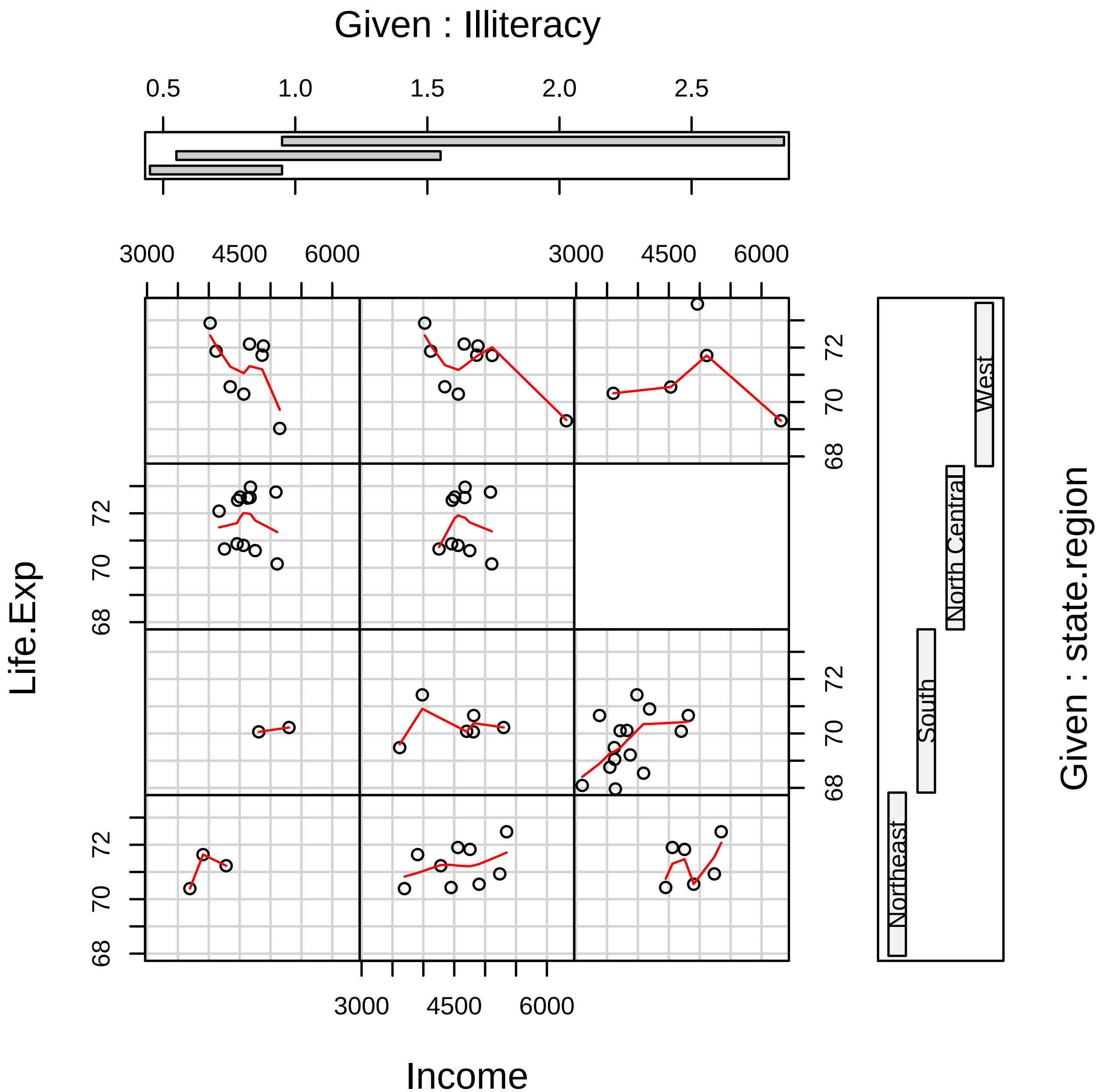
`par()` doesn't work for lattice.

Use `trellis.par.get()` and `trellis.par.set()` to modify lattice parameters

What are the lattice parameters? Mostly undocumented!

`print(trellis.par.get())` gives a list of them, for what it's worth

Another example, this time from base



Lattice-like graphics in base

```
attach(data.frame(state.x77))
coplot(Life.Exp ~ Income | Illiteracy * state.region,
       number = 3,  # of conditioning intervals
       panel = function(x, y, ...)
                 panel.smooth(x, y, span = 0.8, ...)
       )
```

Notice the use of two conditioning variables

Notice the smoother added by panel

Basic Exploratory Data Analysis

We've now covered some of the basic tools of EDA:

- scatterplot matrices
- conditional plots (lattice/trellis)
- histogram-like plots

Let's take a close look at two EDA topics:

the border between EDA & modeling

EDA for high dimensional data

Why is this a topic for visualization?

Simple answer: only way to understand some fits is visually

Deeper answer: visual EDA complements and supports better statistical modeling

Henceforth, our goal will be to use VDQIs to improve our statistical modeling and inference

Complement to your other coursework

The border between EDA and modeling

Models make simplifying assumptions

The precision of model estimates comes from these assumptions

Wanted: Assumptions “pretty close” to the behavior of the data

How do we check? non-parametric & semi-parametric EDA

Approach: partially relax modeling assumptions, and see if data support simplification

E.g., let the line wiggle if it wants; then check for approximate linearity

A framework for probability models of data

Introducing graphical techniques for a wide variety of statistical methods

$\rightarrow$ We need a language to refer to diverse probability models

Most models have a stochatic component:

$$\mathbf{y} \sim f_{\mathcal{D}}(\boldsymbol{\mu}, \boldsymbol{\alpha})$$

and a systematic component

$$\boldsymbol{\mu} = g(\mathbf{X}, \boldsymbol{\beta})$$

$\mathbf{y}$ is the data vector of interest

$\mathbf{X}$ is a matrix of covariates

$f_{\mathcal{D}}$ is a probability density function for distribution $\mathcal{D}$

$\boldsymbol{\mu}$ is (usually) the expected value

$\boldsymbol{\alpha}$ is a "nuisance" parameter vector

$\boldsymbol{\beta}$ is a parameter vector associated with the covariates

A framework for probability models of data

$$\mathbf{y} \sim f_{\mathcal{D}}(\mu, \alpha)$$
$$\mu = g(\mathbf{X}, \beta)$$

nests most (if not all) models you know.

Linear regression:	$\mathbf{y}$	$\sim$	$f_{\text{Normal}}(\boldsymbol{\mu}, \sigma^2)$
(continuous data)	$\boldsymbol{\mu}$	$=$	$\mathbf{X}\boldsymbol{\beta}$
Logit:	$\mathbf{y}$	$\sim$	$f_{\text{Bernoulli}}(\boldsymbol{\mu})$
(binary data)	$\boldsymbol{\mu}$	$=$	$\text{logit}^{-1}(\mathbf{X}\boldsymbol{\beta})$
Poisson:	$\mathbf{y}$	$\sim$	$f_{\text{Poisson}}(\boldsymbol{\lambda})$
(count data)	$\boldsymbol{\lambda}$	$=$	$\exp(\mathbf{X}\boldsymbol{\beta})$

and so on

A framework for probability models of data

$$\mathbf{y} \sim f_{\mathcal{D}}(\mu, \alpha)$$
$$\mu = g(\mathbf{X}, \beta)$$

nests most (if not all) models you know.

Linear regression:	$\mathbf{y}$	$\sim$	$f_{\text{Normal}}(\boldsymbol{\mu}, \sigma^2)$
(continuous data)	$\boldsymbol{\mu}$	$=$	$\mathbf{X}\boldsymbol{\beta}$
Logit:	$\mathbf{y}$	$\sim$	$f_{\text{Bernoulli}}(\boldsymbol{\mu})$
(binary data)	$\boldsymbol{\mu}$	$=$	$[1 - \exp(-\mathbf{X}\boldsymbol{\beta})]^{-1}$
Poisson:	$\mathbf{y}$	$\sim$	$f_{\text{Poisson}}(\boldsymbol{\lambda})$
(count data)	$\boldsymbol{\lambda}$	$=$	$\exp(\mathbf{X}\boldsymbol{\beta})$

and so on

A framework for probability models of data

Note the parallel to the notation of Generalized Linear Models (GLMs)

For example, we can write logit equivalently

$\mathbf{y} \sim f_{\text{Bernoulli}}(\boldsymbol{\mu})$	$\mathbf{y} \sim f_{\text{Bernoulli}}(\boldsymbol{\mu})$
$\boldsymbol{\mu} = g(\mathbf{X}\boldsymbol{\beta})$	$g^{-1}(\mu) = \mathbf{X}\boldsymbol{\beta}$
$\boldsymbol{\mu} = [1 - \exp(-\mathbf{X}\boldsymbol{\beta})]^{-1}$	$\log[\boldsymbol{\mu}/(1 - \boldsymbol{\mu})] = \mathbf{X}\boldsymbol{\beta}$

The framework on the left applies to just about any distribution you will encounter

It is equivalent to the form on the right, which is customary for GLMs

$g(\cdot)$ is called the link function in the GLM context

GLMs are a class of models for which $f_{\mathcal{D}}$ is a member of the exponential family, which includes the Normal, Binomial, Gamma, and Poisson; in R, see `?family()`

A framework for probability models of data

Nice aspects of the framework:

- General: works for most models & most estimation methods (MLE, Bayes, etc.)
- Focuses on the data of interest, $\mathbf{y}$
- Reduces attention to $\boldsymbol{\beta}$, which is just a cog in the machine that turns $\mathbf{X}$ into $\mathbf{y}$
  (For different models, $\boldsymbol{\beta}$ has different, usually non-obvious interpretations)
- For any given counterfactual set of covariate values $\mathbf{x}_c$,
  the conditional expecation $\mathrm{E}(y_c c|\mathbf{x}_c)$ and
  the expected first difference $\mathrm{E}(\mathbf{y}_c - \mathbf{y}_d|\mathbf{x}_c, \mathbf{x}_d)$
  have simple, substantively interesting interpretations

Under this framework,
our problem is simply to explain how y (or an interesting function of y,)
shifts as we vary $\mathbf{x}_c$

That's usually what our model is for, so that's what we want to visualize

Unpacking the framework

Let's focus on

$$\boldsymbol{\mu} = g(\mathbf{X}, \boldsymbol{\beta})$$

Here are some typical specifications of the RHS of this equation in LS models

$$\boldsymbol{\mu} = \beta_0 + \beta_1 \mathbf{x}_1 + \beta_2 \mathbf{x}_2 + \ldots$$

$$\boldsymbol{\mu} = \beta_0 + \beta_1 \mathbf{x}_1^2 + \beta_2 \mathbf{x}_2 + \ldots$$

$$\boldsymbol{\mu} = \beta_0 + \beta_1 \mathbf{x}_1 + \beta_2 \log(\mathbf{x}_2) + \ldots$$

A general form for these transformations:

$$\boldsymbol{\mu} = \boldsymbol{\eta} + h_1(\beta_1, \mathbf{x}_1) + h_2(\beta_2, \mathbf{x}_2) \ldots$$

Unpacking the framework

A general form for these transformations:

$$\boldsymbol{\mu} = \boldsymbol{\eta} + h_1(\beta_1, \mathbf{x}_1) + h_2(\beta_2, \mathbf{x}_2) \ldots$$

which we can write compactly as:

$$\boldsymbol{\mu} = \boldsymbol{\eta} + \sum_{k=1}^{p} h_k(\beta_k, \mathbf{x}_k)$$

$h_k(\cdot)$ could be any arbitrary function.

We could multiply β by the square of $\mathbf{x}_k$

We could multiply β by the log of $\mathbf{x}_k$

We could multiply β by the average of the nearest t neighboring x_{ki}'s

Unpacking the framework

$$\boldsymbol{\mu} = \boldsymbol{\eta} + \sum_{k=1}^{p} h_k(\beta_k, \mathbf{x}_k)$$

What if we don't even need a β_k for some of our $h_k(\cdot)$'s?

E.g., for time series $\mathbf{y}$, we could use a running average of (say) the last 3 values $\mathbf{y}$ itself as the sole predictors:

$$\mathrm{E}(y_i) = \sum_{t=1}^{3} y_{i-t}/3$$

Then we get a nonparametric specification – there are no β's or other unknown parameters!

Note that nonparametric does not mean choiceless—
we still had to choose 3 rather than 5 values to average

But there is no population parameter to *estimate*,
just a modeling choice to *calibrate*

Non-parametric models: smooths

Non-parametric "smooths", like the moving average, fall on the border between plots of the data and traditional regression models

Scatterplots leave model "fitting" up to the viewer's eyes

Regression models, such as this linear regression on polynomials:

$$y_i = \beta_0 + \beta x_{1i} + \beta_1 x_{1i}^2 + \varepsilon_i$$

or this logit regression on linear terms

$$\Pr(y_i|x_i) = \mathrm{logit}^{-1}(\beta_0 + \beta_1 x_{1i})$$

make assumptions the viewer must take on faith, esp. the *specification*

Choices about whether terms should enter as linear, cubed, logged, etc
are often arbitrary

Smooths reduce the influence of the model on functional form, in favor of the data

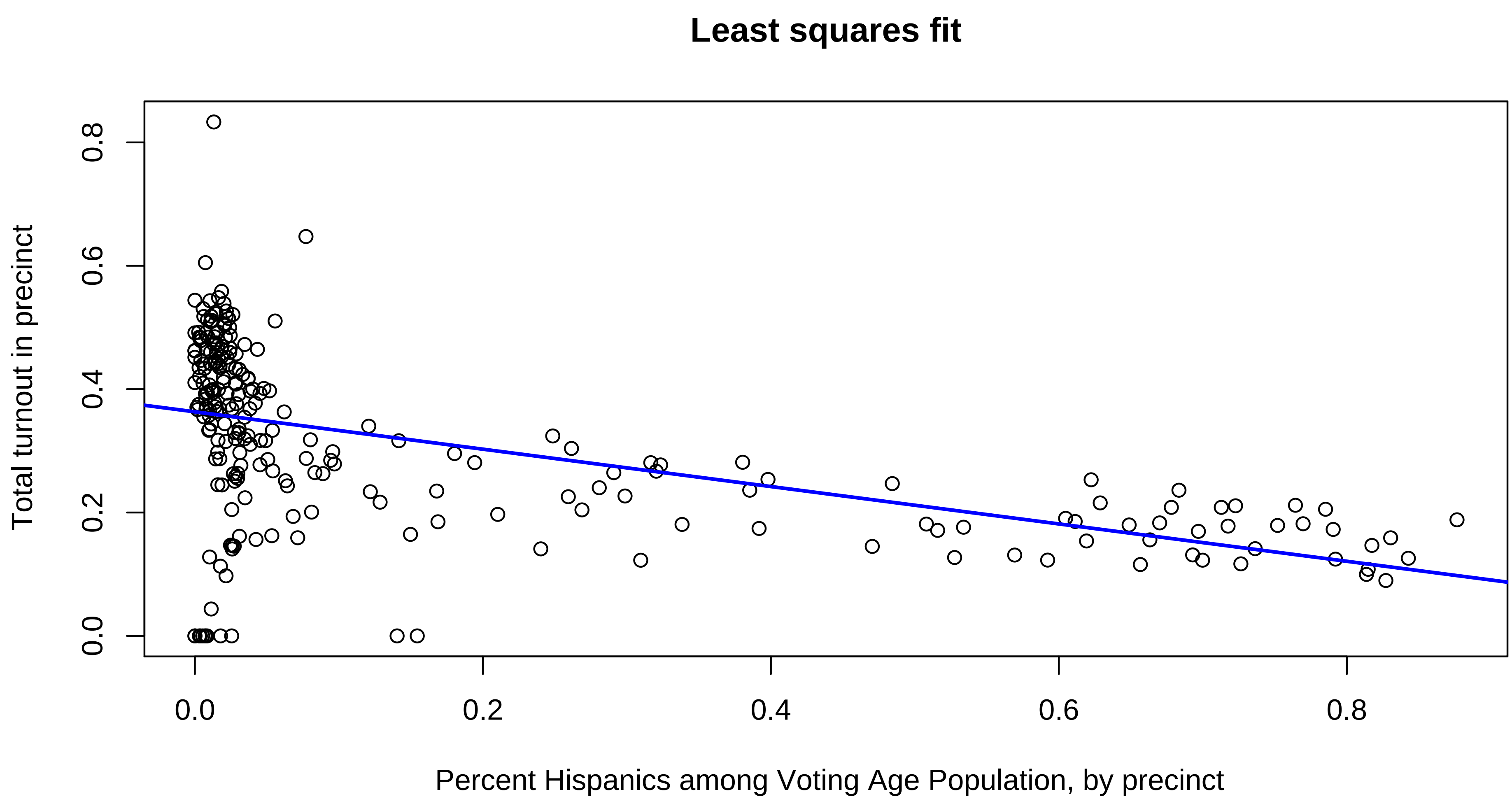
Non-parametric models: smooths

Some commonly used smoothers:

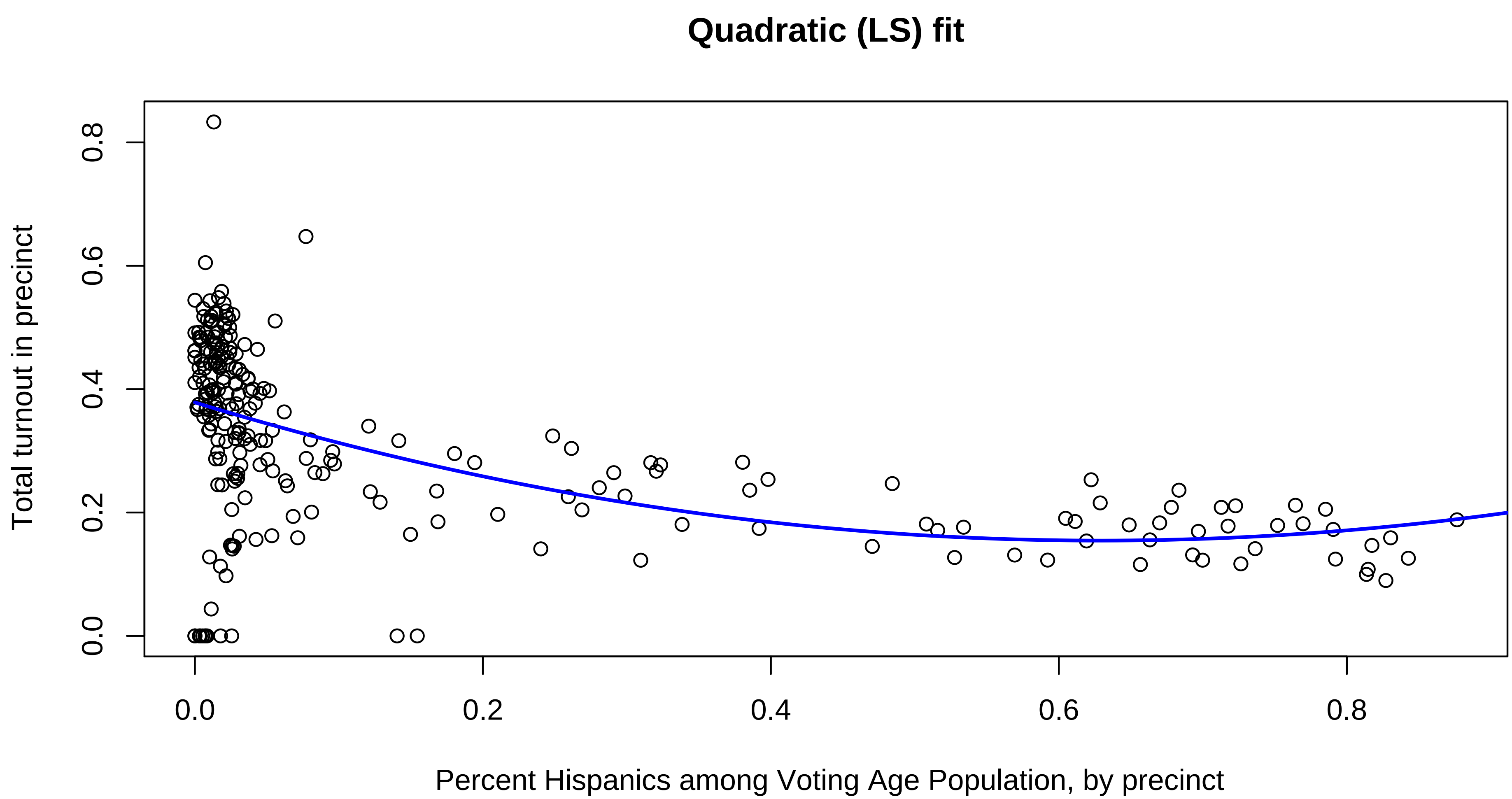
- running averages
- running medians
- loess
- splines
- kernel density (mainly useful for smoothing histograms)

Let's look at how a variety of smoothers deal with a simple dataset:

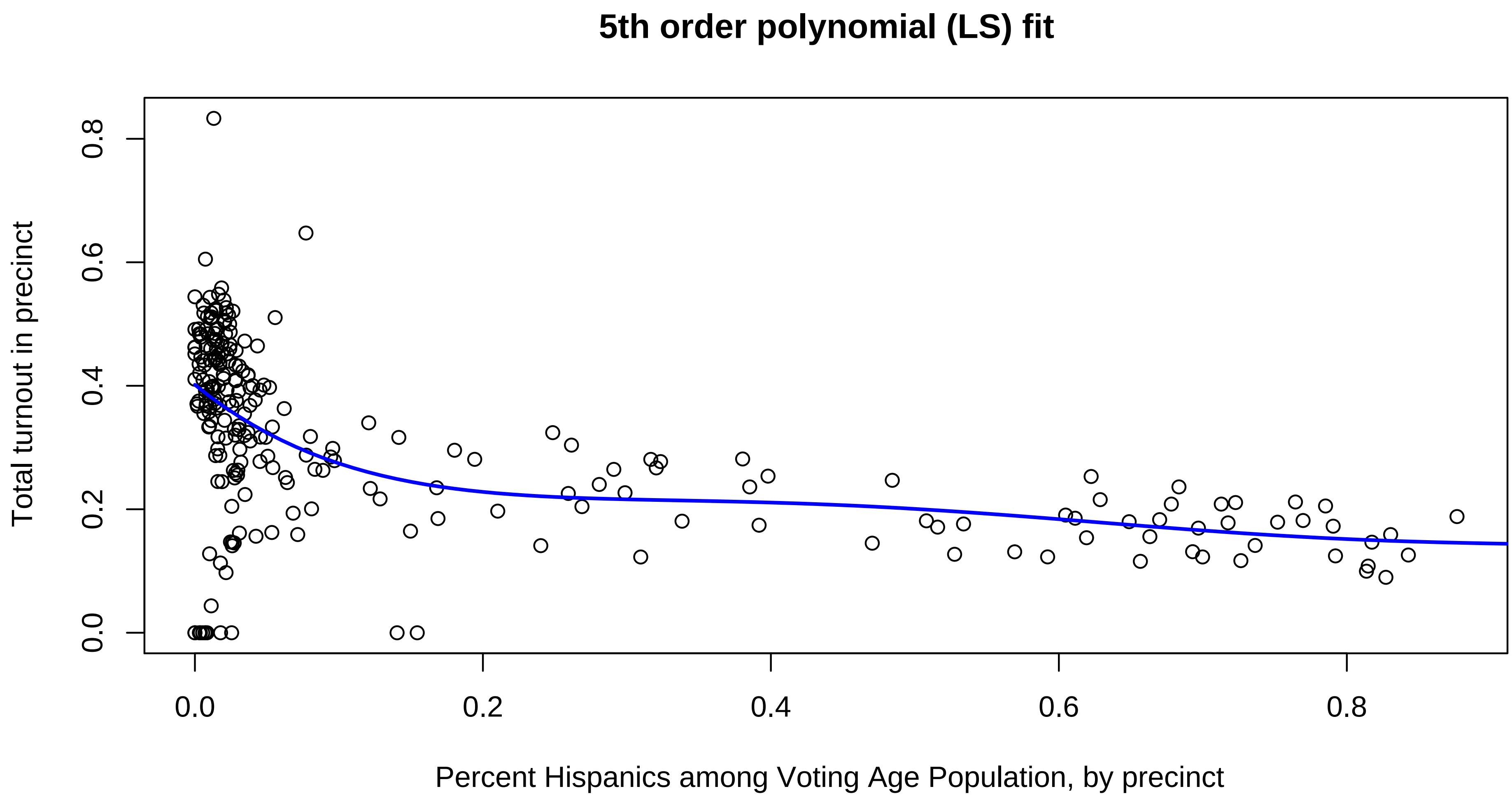
Turnout (`turnout`) as a function of %Hispanic voters (`hisp`)
in a Pennsylvania State Senate election




```
res.lm <- lm(turnout~hisp)
abline(coef(res.lm)[1], coef(res.lm)[2], col="blue", lwd=2)
```

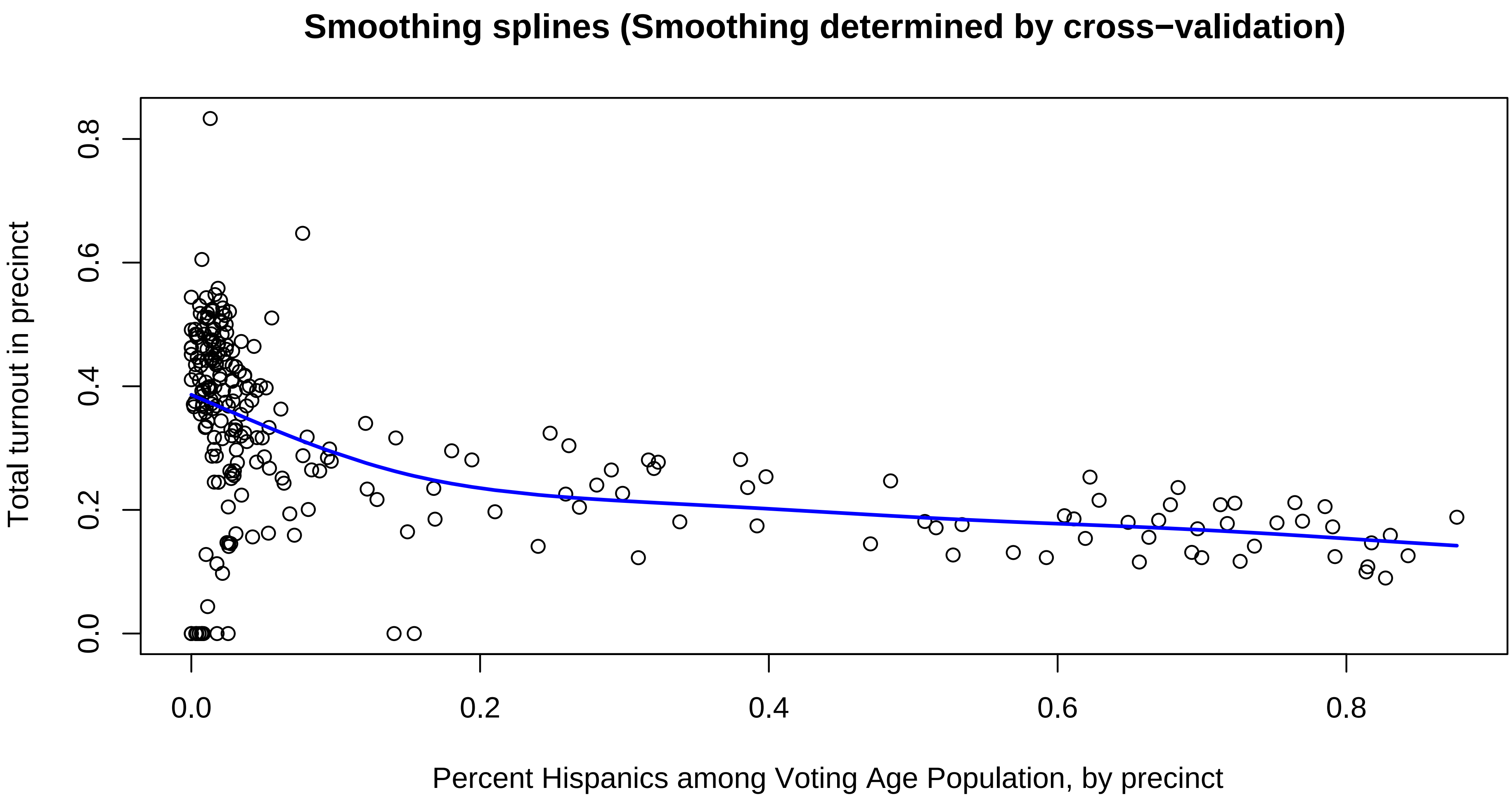



```
res.q <- lm(turnout~hisp + I(hisp^2))
pred.q <- hypHisp <- seq(0,1,0.01)
for (i in 1:length(allx)) {
  pred.q[i] <- coef(res.q)[1] + coef(res.q)[2]*hypHisp[i]
               + coef(res.q)[3]*hypHisp[i]^2
}
```




```
res.q <- lm(t~hisp + I(hisp^2) + I(hisp^3) + I(hisp^4) + I(hisp^5))
```

etc. Danger of overfitting with any polynomial fit.

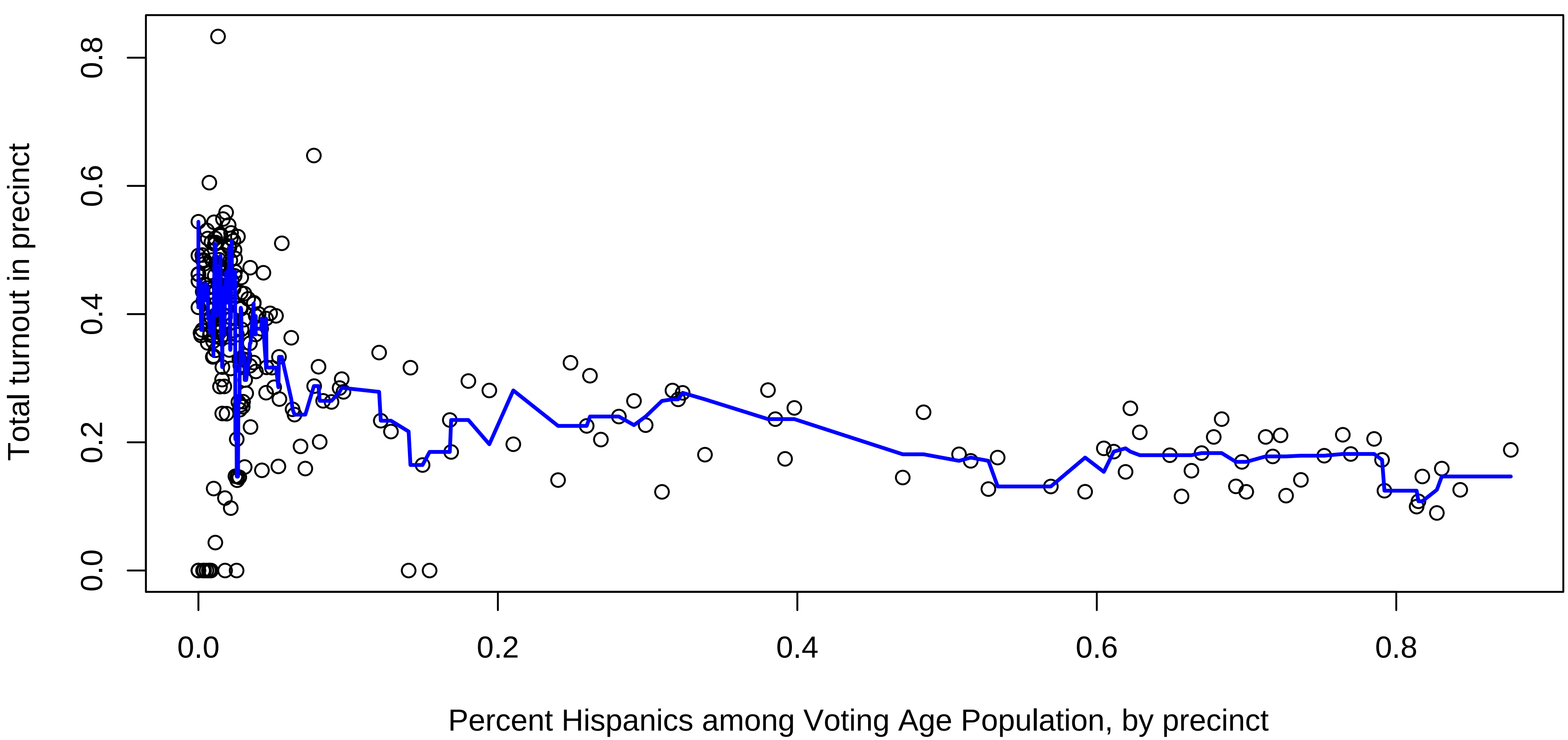



```
lines(smooth.spline(y=turnout, x=hisp), col="blue", lwd=2)
```

We can approximate high order polynomial fits with *smoothing splines*

This function tries to avoid overfitting by selecting the “wiggliness” based on cross-validation

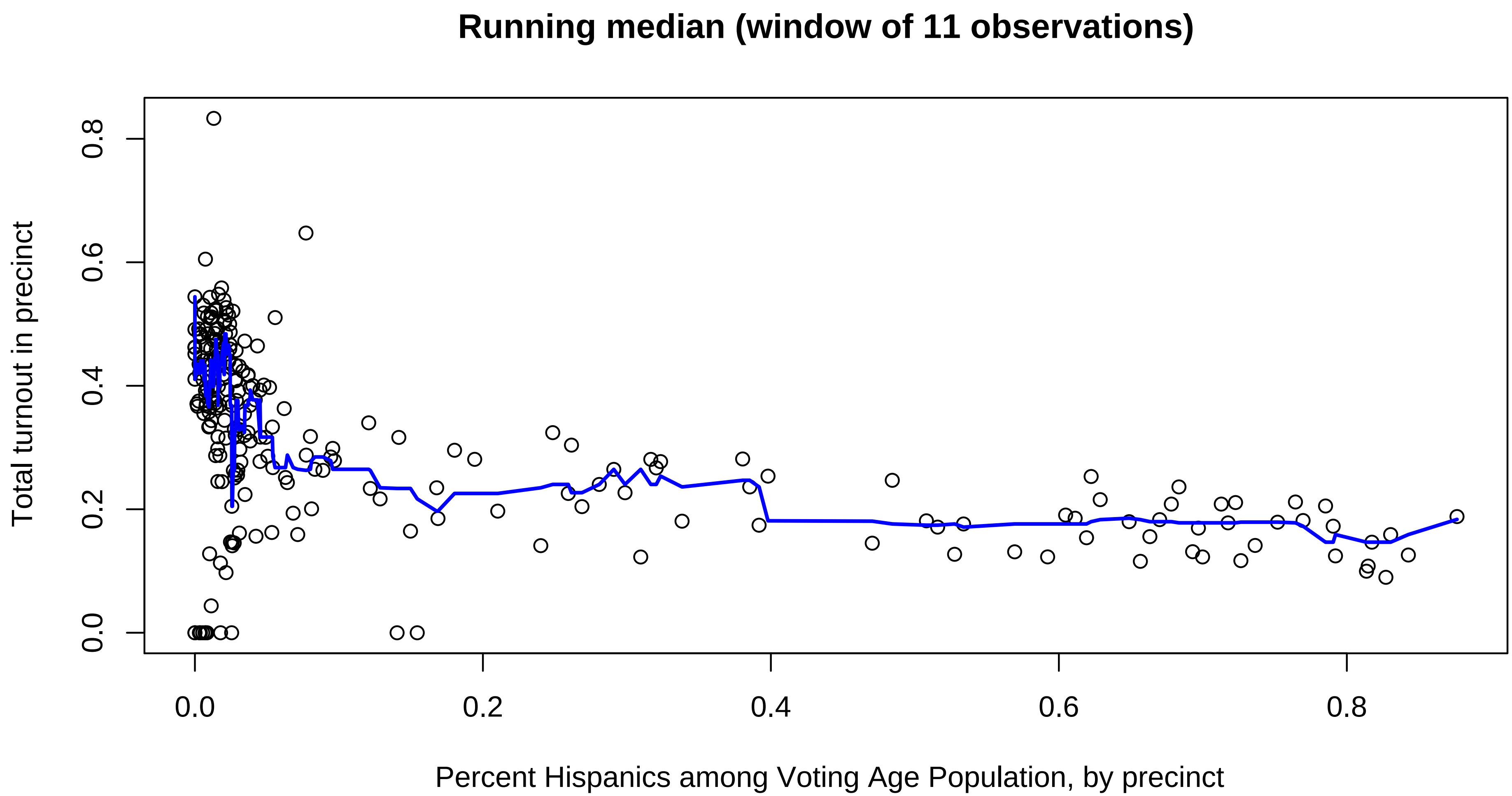
Running median (window of 5 observations)



This works only if `turnout` and `hisp` both sorted by `hisp` *first*:

```
lines(x=hisp, y=runmed(turnout, k=5), lwd=2, col="blue")
```

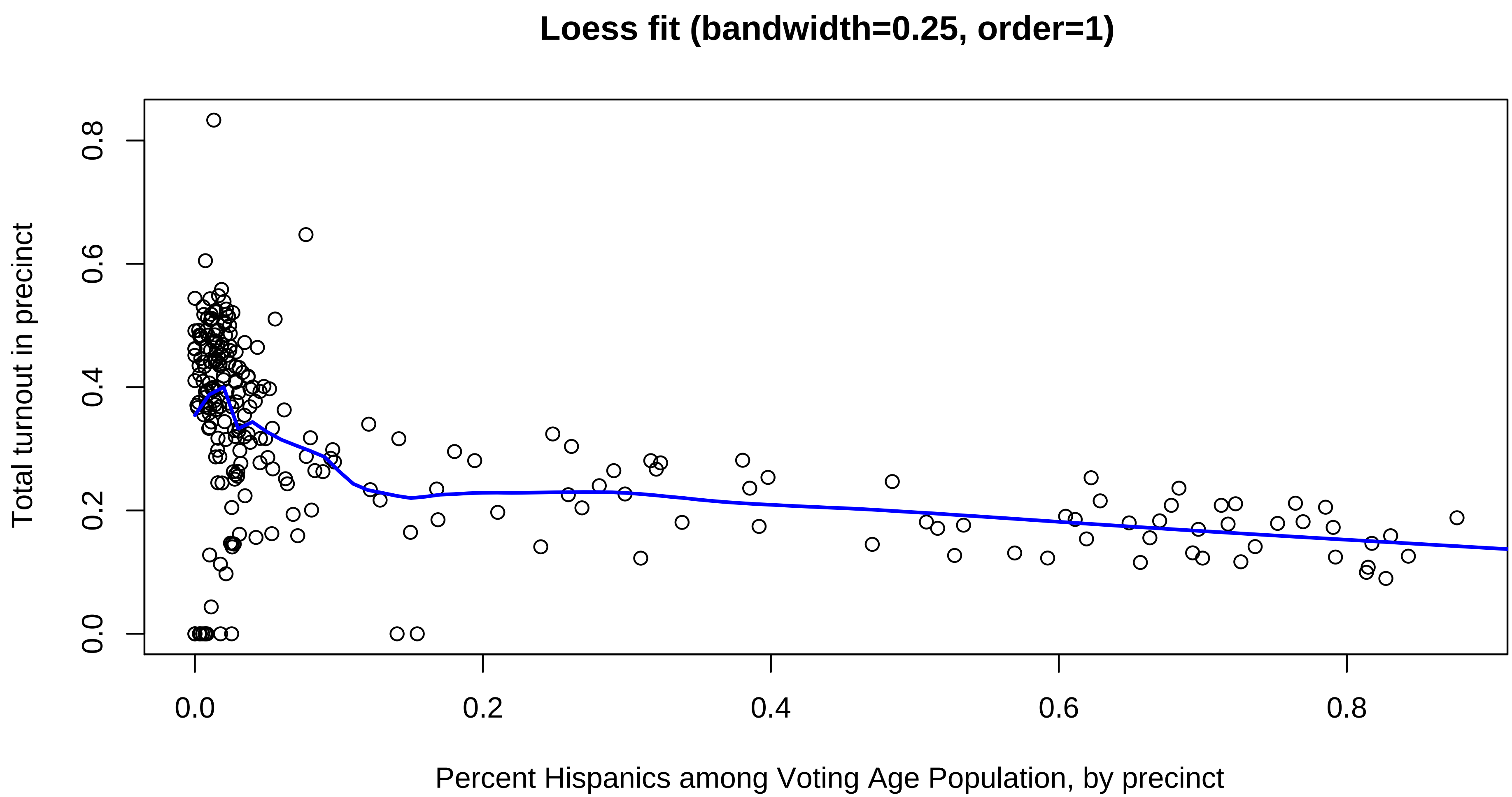
Maximum robustness: not very smooth though



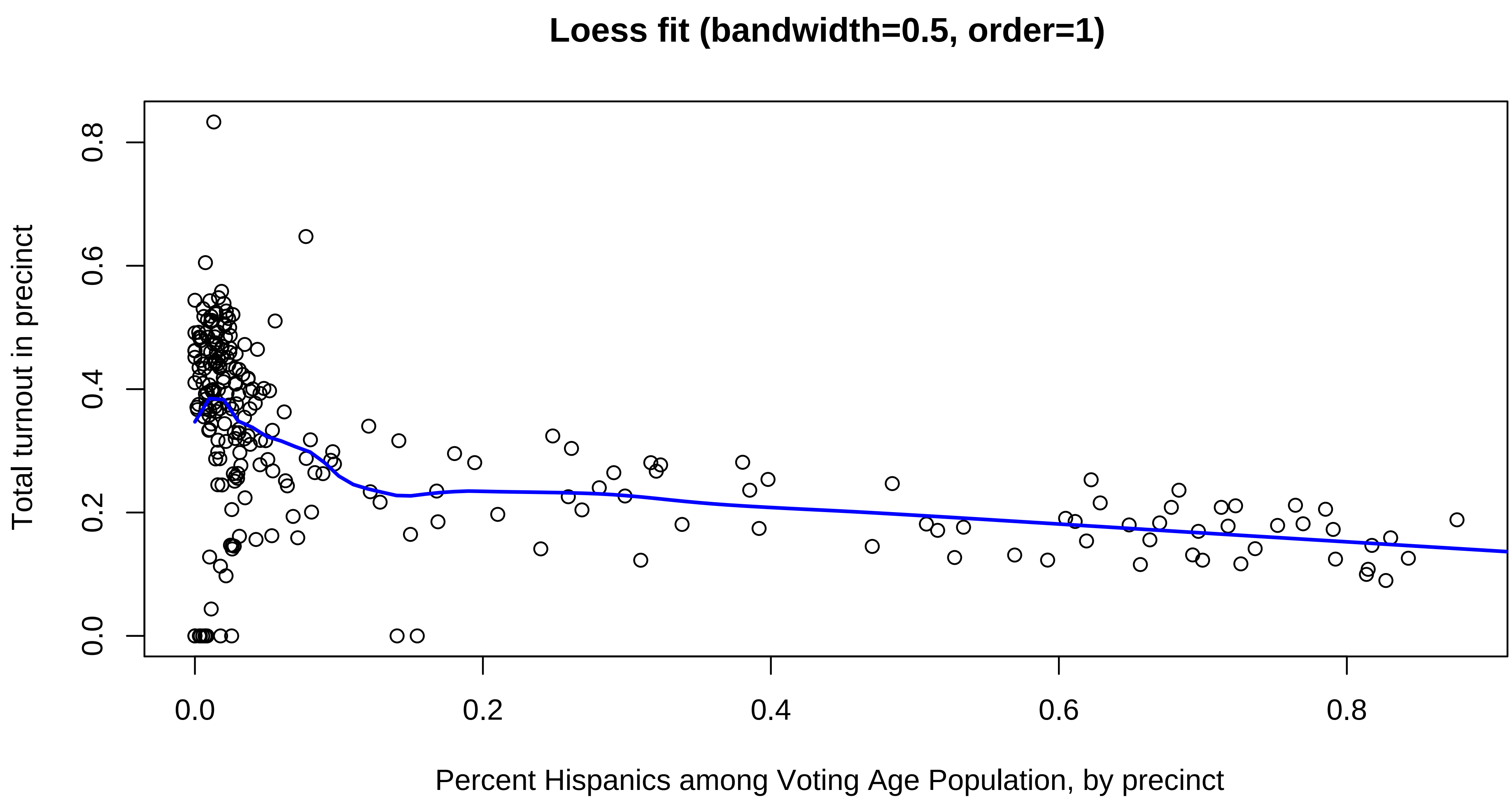
This works only if `t` and `x` both sorted by `x` *first*:

```
lines(x=hisp, y=runmed(turnout, k=11), lwd=2, col="blue")
```

Maximum robustness: A bit smoother, but we'd like another alternative

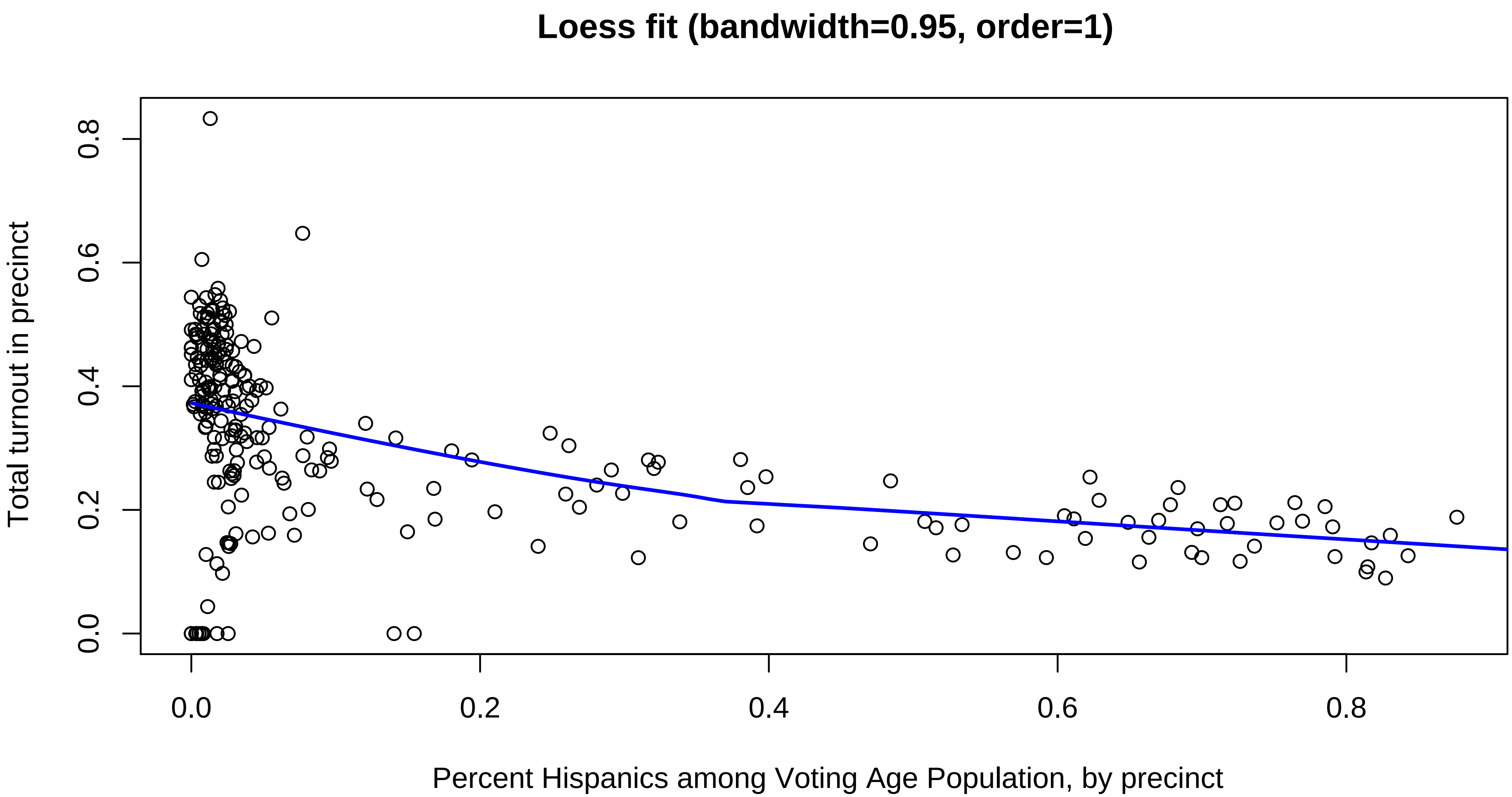



```
res.loess25 <- loess(turnout~hisp,
                     surface="direct",
                     degree=1,
                     span=0.25)
pred.loess25 <- predict(res.loess25, newdata=hypHisp)
lines(x=hypHisp, y=pred.loess25, col="blue", lwd=2)
```




```
res.loess50 <- loess(turnout~hisp,
                     surface="direct",
                     degree=1,
                     span=0.5)
```

Higher bandwidth → a smoother plot




```
res.loess95 <- loess(turnout~hisp,
                     surface="direct",
                     degree=1,
                     span=0.95)
```

This is about as high as we go. Note that we find a kink.

Non-parametric models: smooths

Lots of mathematical details here.

But key issue is how much flexibility to allow

In each model, there is a choice of how flexible the fit should be

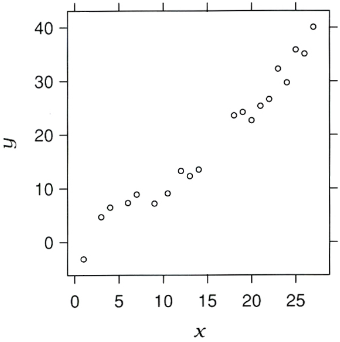
- for loess and kernel, the bandwidth;
- for splines, the degree of smoothing and number of knots
- for running averages, the number and period of the averages

How do we perform this fit? And why is there no β?

$$\hat{\mathbf{y}} = f_{\text{loess}}(\mathbf{x})$$

How Loess Works

Let's fit a loess curve to the data at right

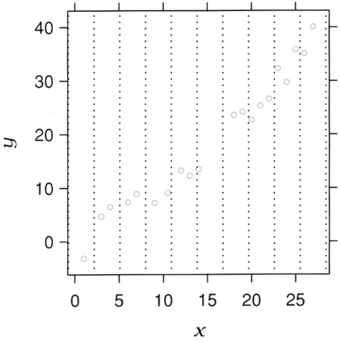


Adapted from: Cleveland, *Visualizing Data*

How Loess Works

Let's fit a loess curve to the data at right

1. Choose a sequence $\mathbf{x}^*$ of k equally spaced points at which to calculate loess fits $\mathbf{y}^*$

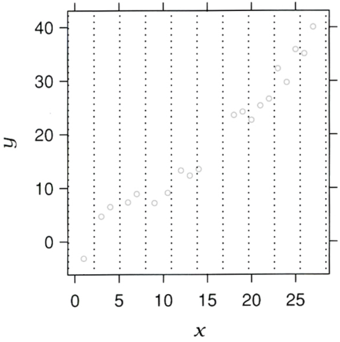


Adapted from: Cleveland, *Visualizing Data*

How Loess Works

Let's fit a loess curve to the data at right

1. Choose a sequence $\mathbf{x}^*$ of k equally spaced points at which to calculate loess fits $\mathbf{y}^*$
2. Choose a smoothing parameter α

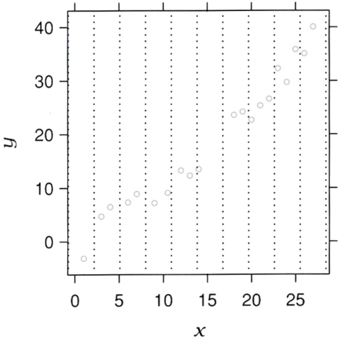


Adapted from: Cleveland, *Visualizing Data*

How Loess Works

Let's fit a loess curve to the data at right

1. Choose a sequence $\mathbf{x}^*$ of k equally spaced points at which to calculate loess fits $\mathbf{y}^*$
2. Choose a smoothing parameter α
3. Choose a polynomial order λ; usually 1 or 2



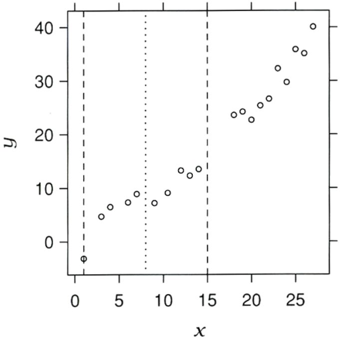
Adapted from: Cleveland, *Visualizing Data*

How Loess Works

Let's fit a loess curve to the data at right

1. Choose a sequence $\mathbf{x}^*$ of k equally spaced points at which to calculate loess fits $\mathbf{y}^*$
2. Choose a smoothing parameter α
3. Choose a polynomial order λ; usually 1 or 2
4. For each element in $\mathbf{x}^*$, x_k^*,

   i. Select the $\alpha n/2$ nearest neighbors to x_k^*,



Adapted from: Cleveland, *Visualizing Data*

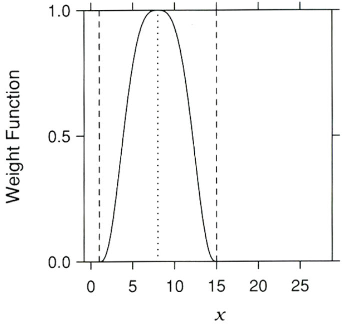
How Loess Works

Let's fit a loess curve to the data at right

1. Choose a sequence $\mathbf{x}^*$ of k equally spaced points at which to calculate loess fits $\mathbf{y}^*$
2. Choose a smoothing parameter α
3. Choose a polynomial order λ; usually 1 or 2
4. For each element in $\mathbf{x}^*$, x_k^*,

   i. Select the $\alpha n/2$ nearest neighbors to x_k^*,

   ii. Apply weights descending in distance from x_k^*



Adapted from: Cleveland, *Visualizing Data*

How Loess Works

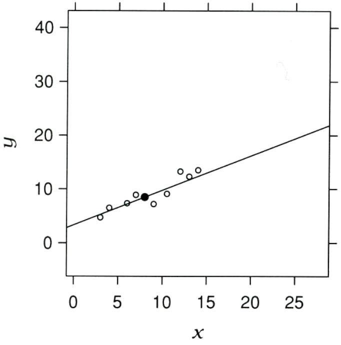
Let's fit a loess curve to the data at right

1. Choose a sequence $\mathbf{x}^*$ of k equally spaced points at which to calculate loess fits $\mathbf{y}^*$
2. Choose a smoothing parameter α
3. Choose a polynomial order λ; usually 1 or 2
4. For each element in $\mathbf{x}^*$, x_k^*,

   i. Select the $\alpha n/2$ nearest neighbors to x_k^*,

   ii. Apply weights descending in distance from x_k^*

   iii. Fit $\mathbf{y}_{selected}$ on the λth-order polynomial of $\mathbf{x}_{selected}$ by WLS



Adapted from: Cleveland, Visualizing Data

How Loess Works

Let's fit a loess curve to the data at right

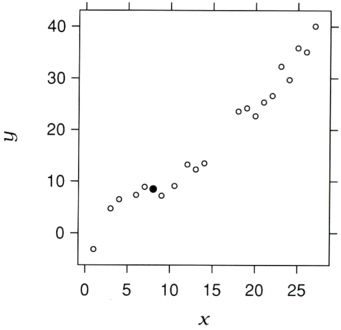
1. Choose a sequence $\mathbf{x}^*$ of k equally spaced points at which to calculate loess fits $\mathbf{y}^*$
2. Choose a smoothing parameter α
3. Choose a polynomial order λ; usually 1 or 2
4. For each element in $\mathbf{x}^*$, x_k^*,

   i. Select the $\alpha n/2$ nearest neighbors to x_k^*,

   ii. Apply weights descending in distance from x_k^*

   iii. Fit $\mathbf{y}_{selected}$ on the λth-order polynomial of $\mathbf{x}_{selected}$ by WLS

   iv. Record only the WLS prediction of $\hat{y}_k^*$ (discard the rest of the line)

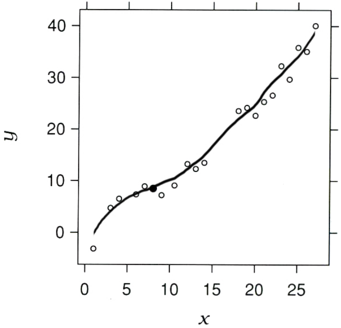


Adapted from: Cleveland, Visualizing Data

How Loess Works

Let's fit a loess curve to the data at right

1. Choose a sequence $\mathbf{x}^*$ of k equally spaced points at which to calculate loess fits $\mathbf{y}^*$
2. Choose a smoothing parameter α
3. Choose a polynomial order λ; usually 1 or 2
4. For each element in $\mathbf{x}^*$, x_k^*,
   i. Select the $\alpha n/2$ nearest neighbors to x_k^*,
   ii. Apply weights descending in distance from x_k^*
   iii. Fit $\mathbf{y}_{selected}$ on the λth-order polynomial of $\mathbf{x}_{selected}$ by WLS
   iv. Record only the WLS prediction of $\hat{y}_k^*$ (discard the rest of the line)
5. Connect the fitted $\hat{y}_k^*$'s to make a curve.



Adapted from: Cleveland, Visualizing Data

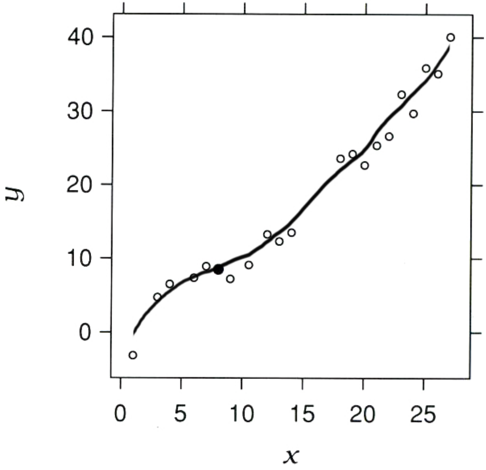
How Loess Works

Key issue here is the bandwidth parameter, $\alpha \in [0, 1]$

Lower values make a jerkier line; higher values a smoother one

If your data seem to curve, use a quadratic fit within loess ($\lambda = 2$) to get a better fit

Note: Our loess example is an artist's impression; let's look at some real ones at `demonstrations.wolfram.com/HowLoessWorks`



Adapted from: Cleveland, *Visualizing Data*

What the heck is a “spline”?

Splines are a concept from carpentry, of all places

Splines let us summarize very complex curves with a few numbers

Basic idea:

- Imagine a flexible piece of wood.
- We pick it up and bend in many places;
- then tack it down (at “knots”) to a board.

The idea behind splines

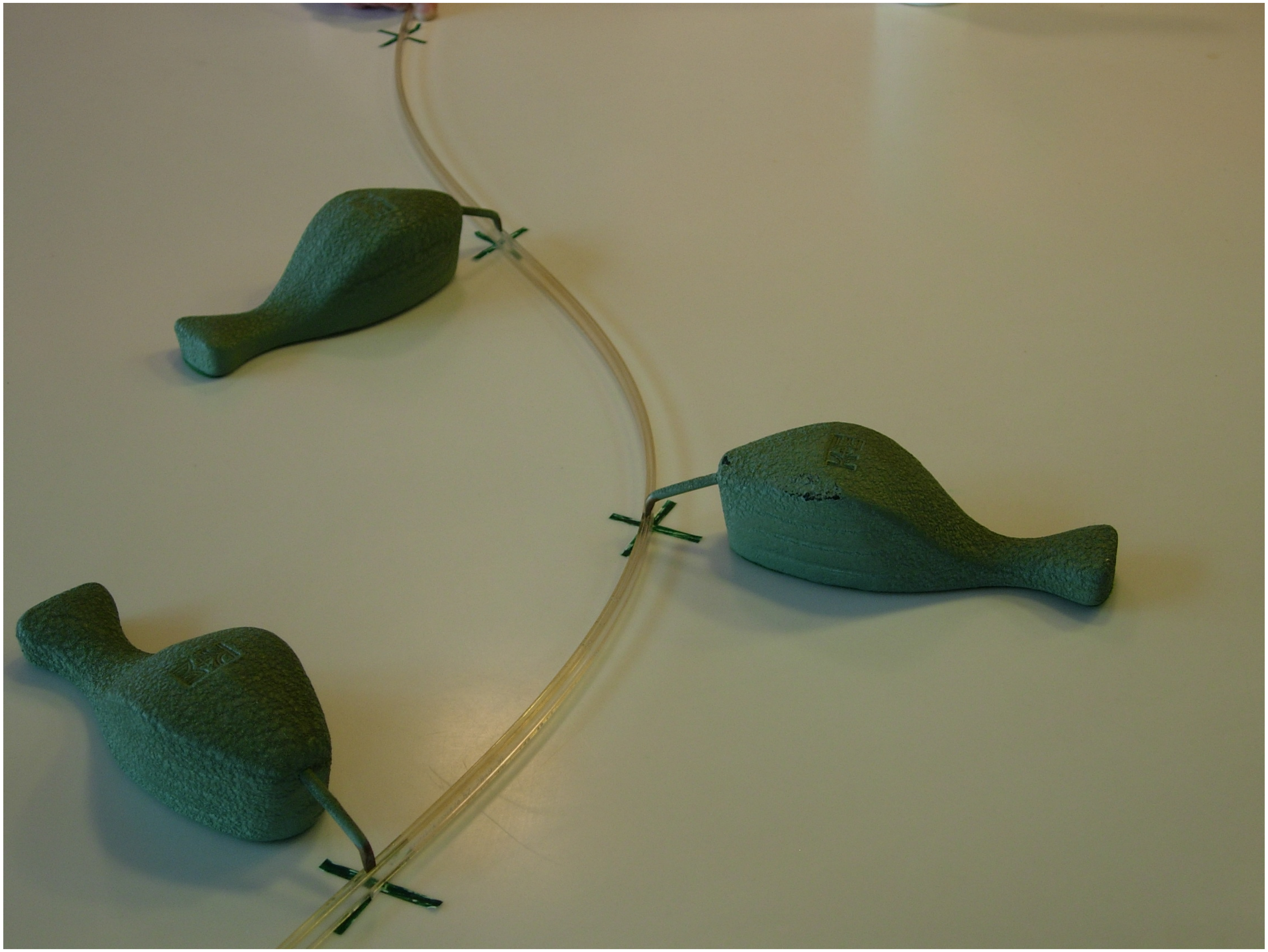


Photo of draftman's spline from Carl de Boor
`www.cs.wisc.edu/d̃eboor/draftspline.html`

Splines in statistics

Many similar shapes can be approximated by local cubic polynomials, which we will call cubic splines

(Note: there are *many* kinds of splines.)

Even more than loess,
Cubic splines rely on simple local structure to create global flexibility

1. Start with a few data points.
2. Connect every point to its nearest neighbor with a (twice differentiable) cubic polynomial
3. To make a "smoothing" or approximate spline,
   take the weighted average of the spline and the least squares fit
4. Choose knots & amt of smoothing subject to a penalized likelihood criterion, e.g.,

$$\max(2 \times \text{ log-likelihood } - \text{ trade-off } \times \text{ smoothness penalty })$$

Spline examples

Using demonstration software

`http://www.particleincell.com/blog/2012/bezier-splines/`

More complex example: Democracy & Development

The relationship between democracy, dictatorship, and economic development is well-explored in political science

Key recent work: Przeworksi, Alvarez, Cheibub & Limongi. *Democracy and Development: Political Insitutions and Well-being in the World, 1950–1990*

We'll borrow their data, but run *much* cruder models

In particular, PACL investigate selection bias and the difference between creation and sustenance of democracies.

We'll ignore these issues, and consider covariates of

the degree of civil liberties (CIVLIB: 1-7 scale)

the presence of democracy (REG: 0-1 binary)

More complex example: Democracy & Development

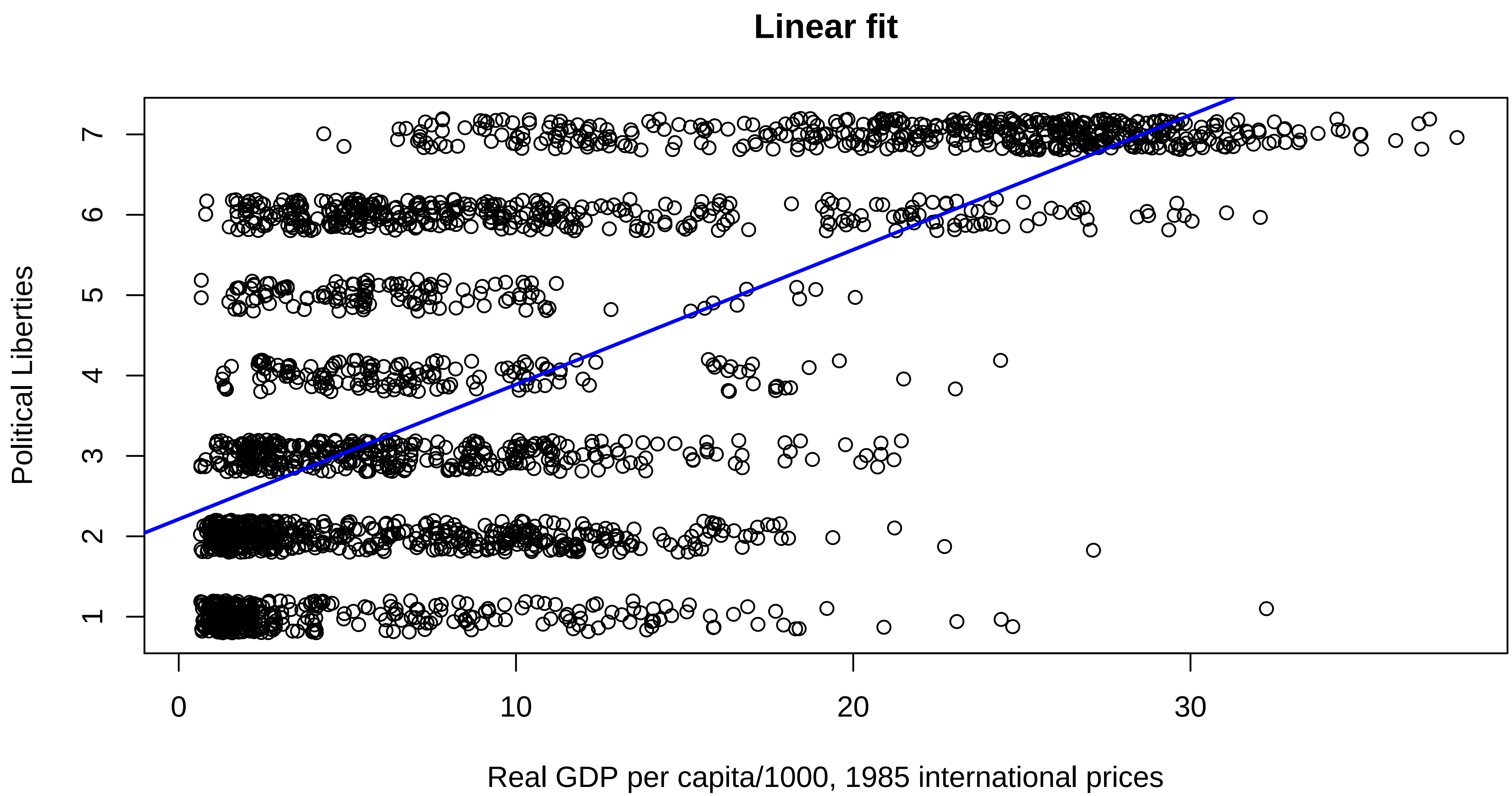
Our candidate covariates

GDPW	GDP per capita in real international prices
EDT	average years of education
ELF60	ethnolinguistic fractionalization
MOSLEM	percentage of Muslims in country
OIL	whether oil accounts for 50+% of exports
STRA	count of recent regime transitions
NEWC	whether county was created after 1945
BRITCOL	whether country was a British colony

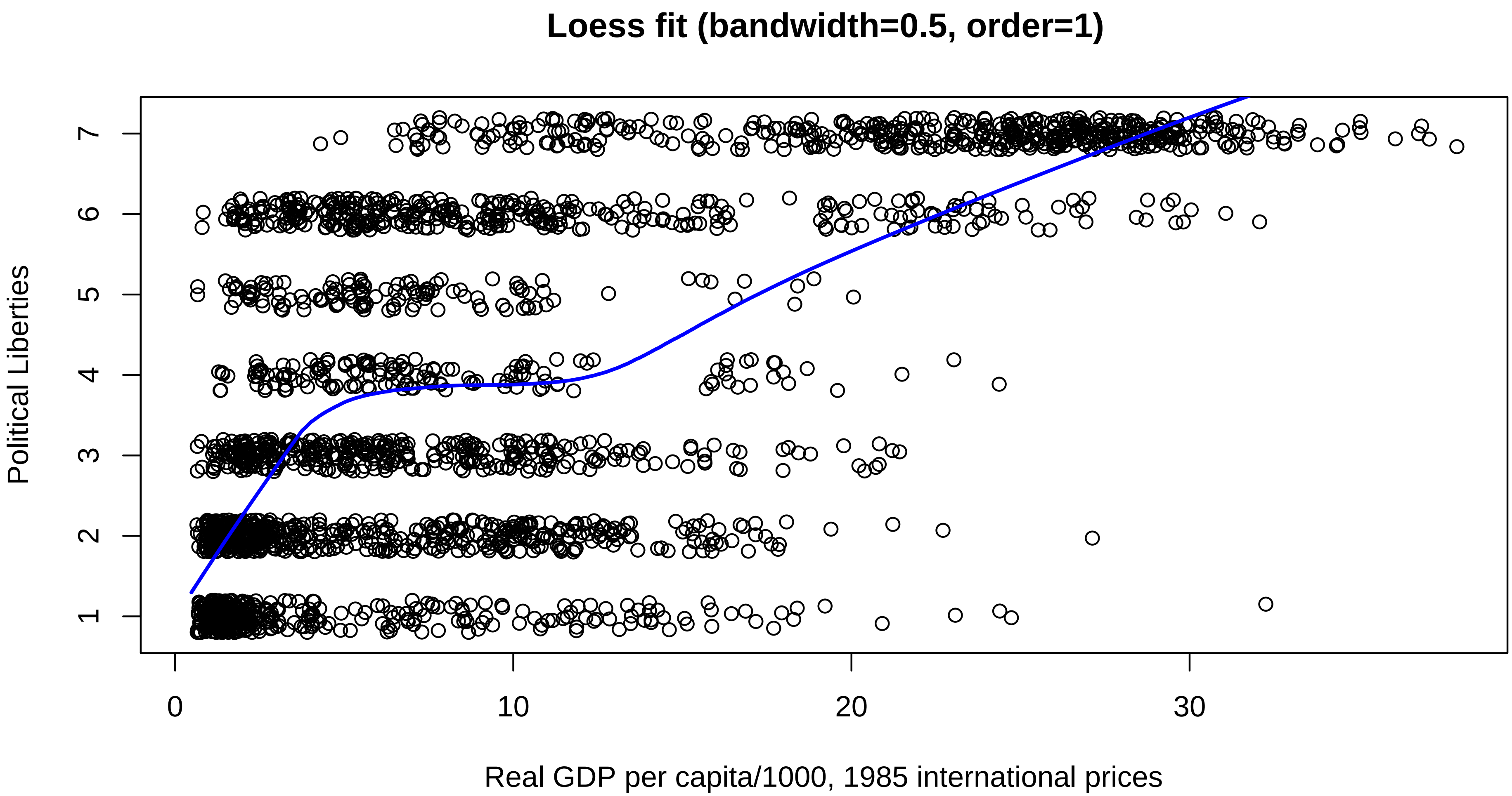
Let's start with the simplest model we can run.

Is there a relationship between civil liberties and economic development?

Let's pretend linear models are appropriate for this categorical variable



Leaving aside the categorical nature of the data, does this fit look "right"?



Loess reveals "thresholds".

A problem: We haven't controlled for *anything*

Smoothers won't be much use if we're restricted to bivariate models

Fortunately, there is a generalization to the multivariate case. . .

Generalized Additive Models (GAMs)

Generalized Additive Models are a generalization of GLMs

incorporate smooths into multiple regression, and into logit, probit, etc.

GAMs take the following form:

$$g^{-1}(\mu) = \alpha + \sum_{j=1}^{p} \beta_j X_j \quad + \sum_{k=1}^{q} f_k(Z_k)$$

where y is a response, X_j and Z_k are covariates, and f_k is a smoothing function

Note we can estimate parametric relations for some covariates, and non-parametric for others

f_k are often splines or loess fits

GAM for Democracy

Let's control for the rest of our variables while letting the degree of political liberties remain a smooth function of the level of development

Sample code:

```
library(mgcv)
res.gam1 <- gam(POLLIB~s(GDPW)+EDT+NEWC+BRITCOL+STRA+ELF60+OIL
                  +MOSLEM)
summary(res.gam1)
```

Notes:

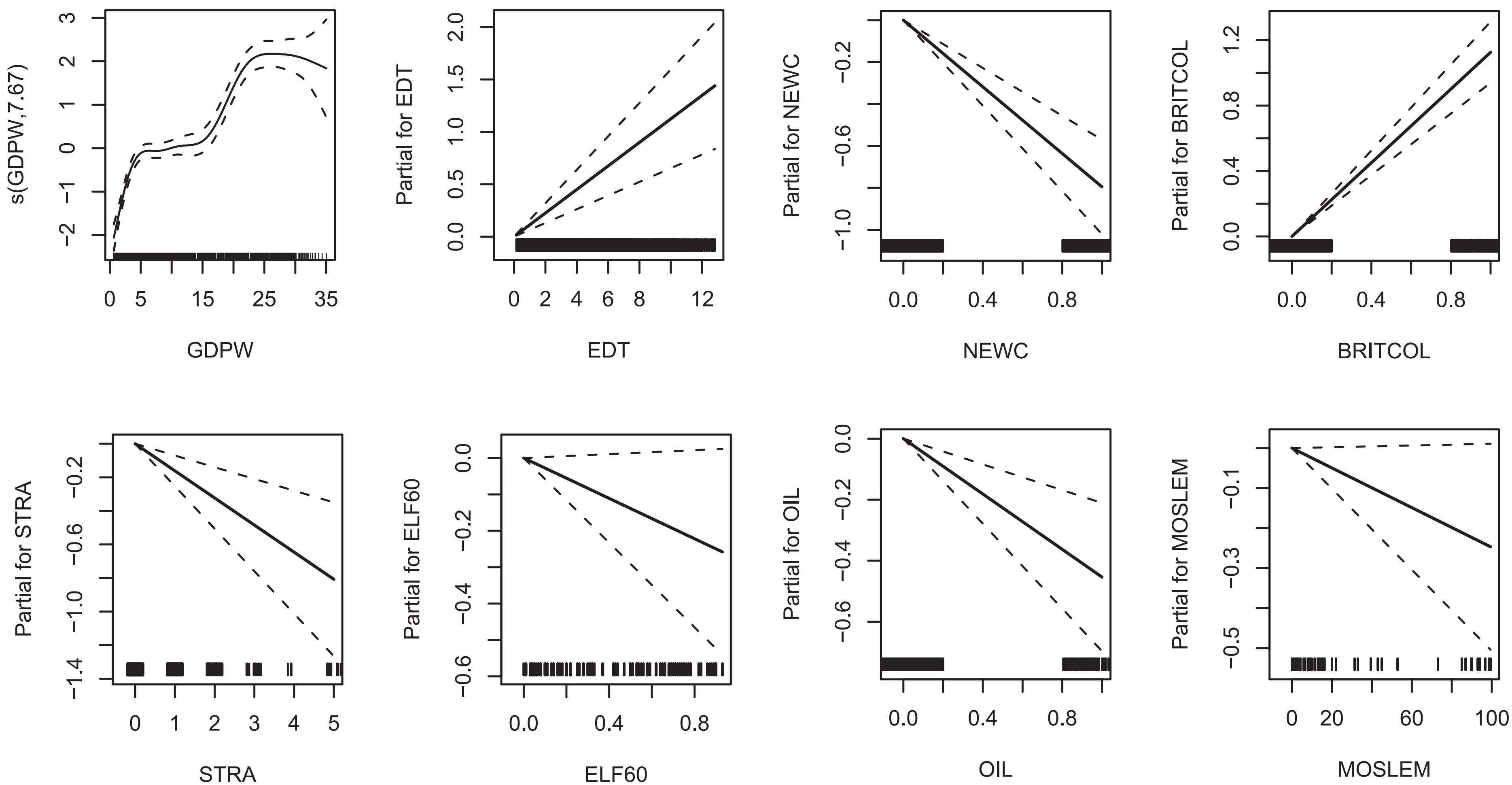
- This is the `gam` function in `mgcv`.
  There is another `gam` in library `gam`; slightly different
- Without specifying a distribution `family`,
  gam defaults to a linear Normal model
- We could specify more details of the smooth;
  this code just let's R find the best smoothing spline by cross-validation

Output of `summary(res.gam1)`

```
Family: gaussian
Link function: identity

Formula:
POLLIB ~ s(GDPW) + EDT + NEWC + BRITCOL + STRA + ELF60 + OIL +
    MOSLEM

Parametric coefficients:
              Estimate  std. err.    t ratio    Pr(>|t|)
(Intercept)     3.8475     0.1552       24.8    < 2.22e-16
        EDT    0.11236    0.02354      4.773    1.9840e-06
       NEWC   -0.79545     0.1125      -7.07    2.3322e-12
    BRITCOL     1.1263    0.09354      12.04    < 2.22e-16
       STRA   -0.16153    0.04582     -3.525    0.00043583
      ELF60    -0.2774     0.1521     -1.824    0.068387
        OIL   -0.45368     0.1214     -3.737    0.00019278
     MOSLEM -0.0024823   0.001295     -1.917    0.055408

Approximate significance of smooth terms:
               edf         chi.sq     p-value
s(GDPW)      7.666         328.44     < 2.22e-16
```

```
R-sq.(adj) =  0.638   Deviance explained = 64.1%
GCV score = 1.8114   Scale est. = 1.7934    n = 1580
```

`mgcv`'s `gam` has preset graphics



Above made with: `plot(res.gam1,pages=1,all.terms=T)`
plus editing in Illustrator

We can smooth over two dimensions

Suppose we wanted to smooth over both development & education:

One way is to let each smooth be additive to the response:

```
library(mgcv)
res.gam2 <- gam(POLLIB~s(GDPW)+s(EDT)+NEWC+BRITCOL+STRA+ELF60+OIL
                  +MOSLEM)
summary(res.gam2)
```

which lets more of the above plots be smooths

Another way is to let the smooths "interact." Sample code:

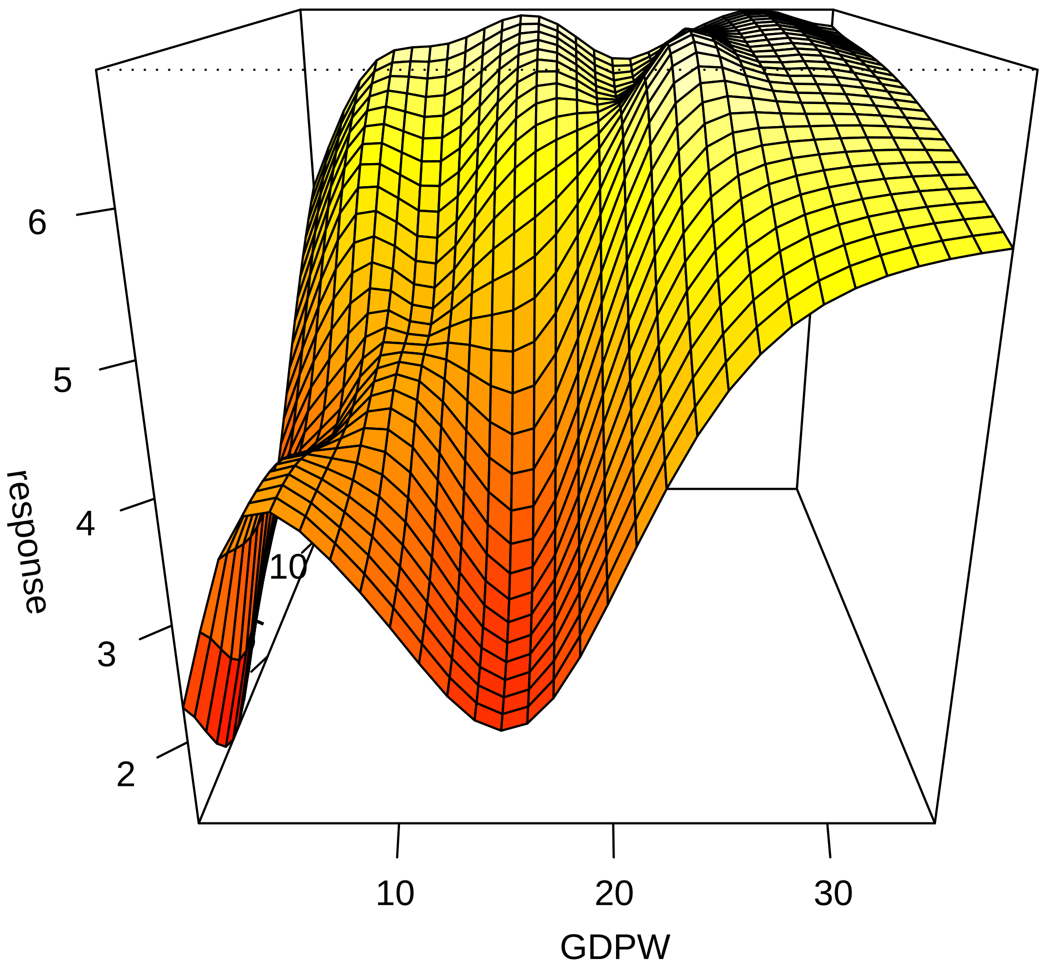
```
library(mgcv)
res.gam2 <- gam(POLLIB~s(GDPW,EDT)+NEWC+BRITCOL+STRA+ELF60+OIL+MOSLEM)
summary(res.gam2)
```

Now there is a smooth *surface* relating the joint levels of GDPW & EDT to POLLIB

We can visualize this with

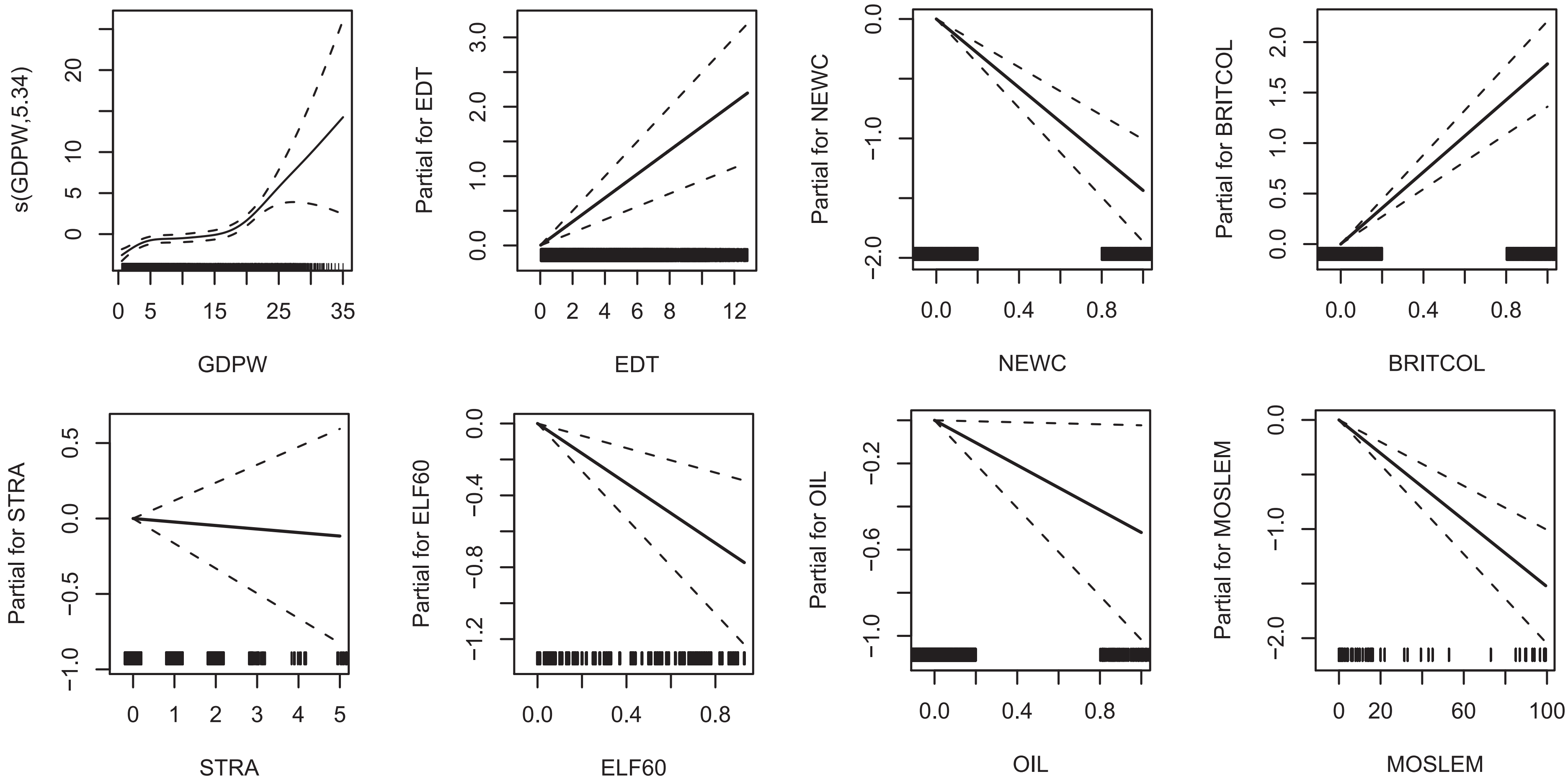
```
vis.gam(res.gam2, view=c("GDPW","EDT"), type="response",
        ticktype="detailed", theta=0, phi=20)
```

A smooth over two covariates together



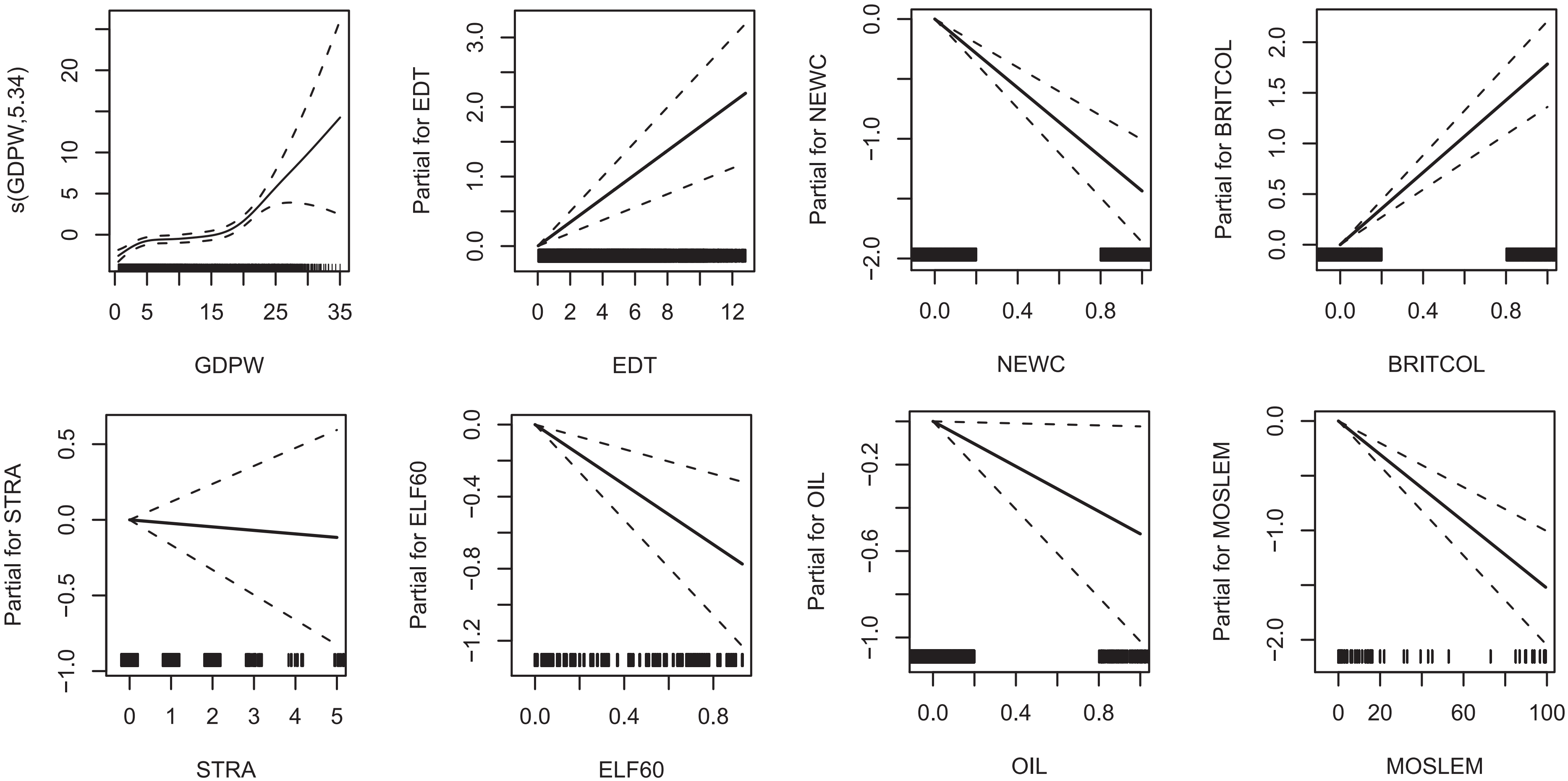
Interpretation is challenging, but this may be occassionally enlightening

A logit example using GAM



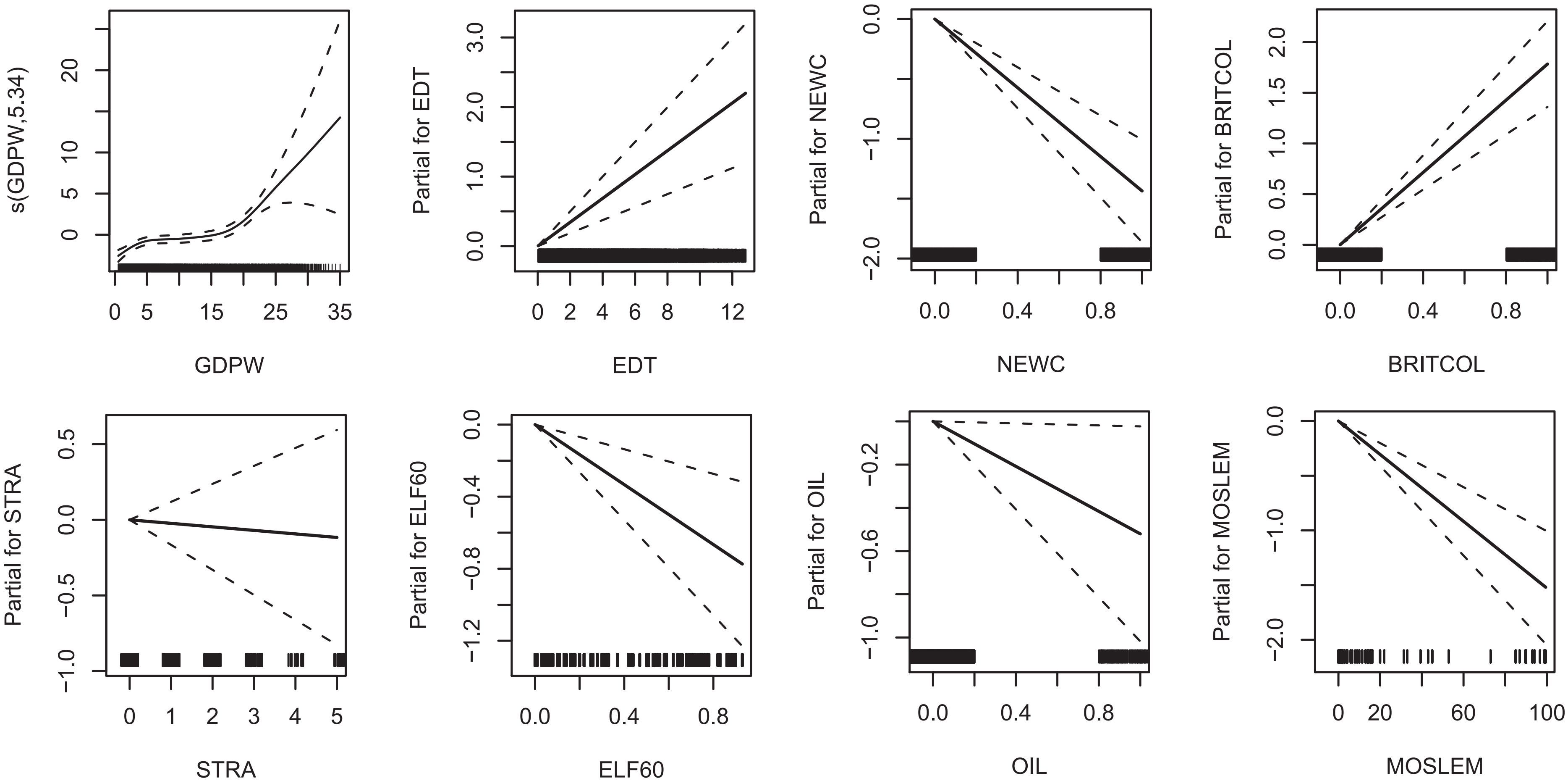
We can use GAM to fit GLM type models. What are the correlates of Democracy?

A logit example using GAM




```
res.gam4 <- gam(REG~s(GDPW)+EDT+NEWC+BRITCOL+STRA+ELF60+OIL
                    +MOSLEM, family=binomial())
summary(res.gam4)
plot(res.gam4, pages=1, all.terms=TRUE)
```

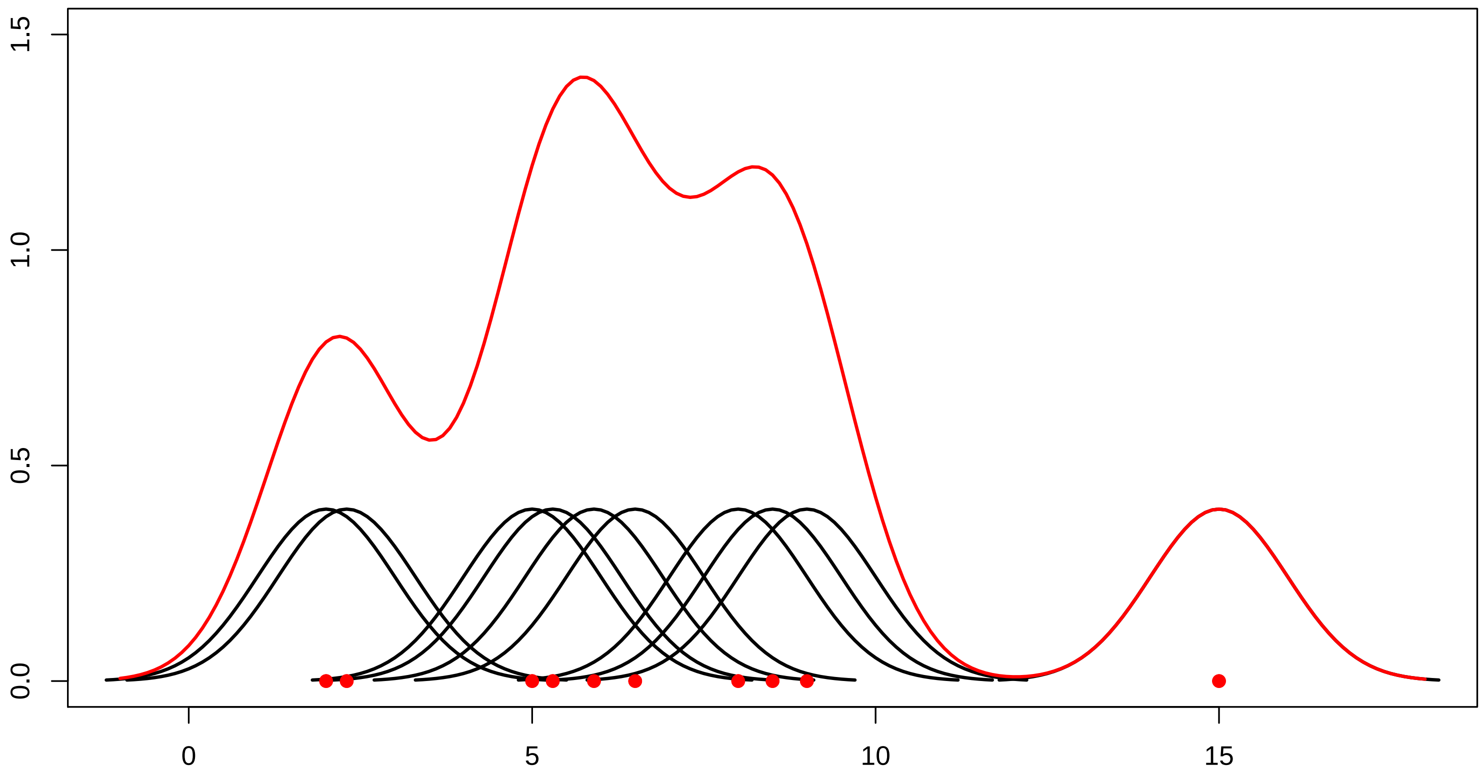
A logit example using GAM



When the GAM is a nonlinear model (e.g., logit), the scale is difficult to interpret

It would be better if the y-axes were in the units of Pr(REG)

One last smooth: kernel density estimation



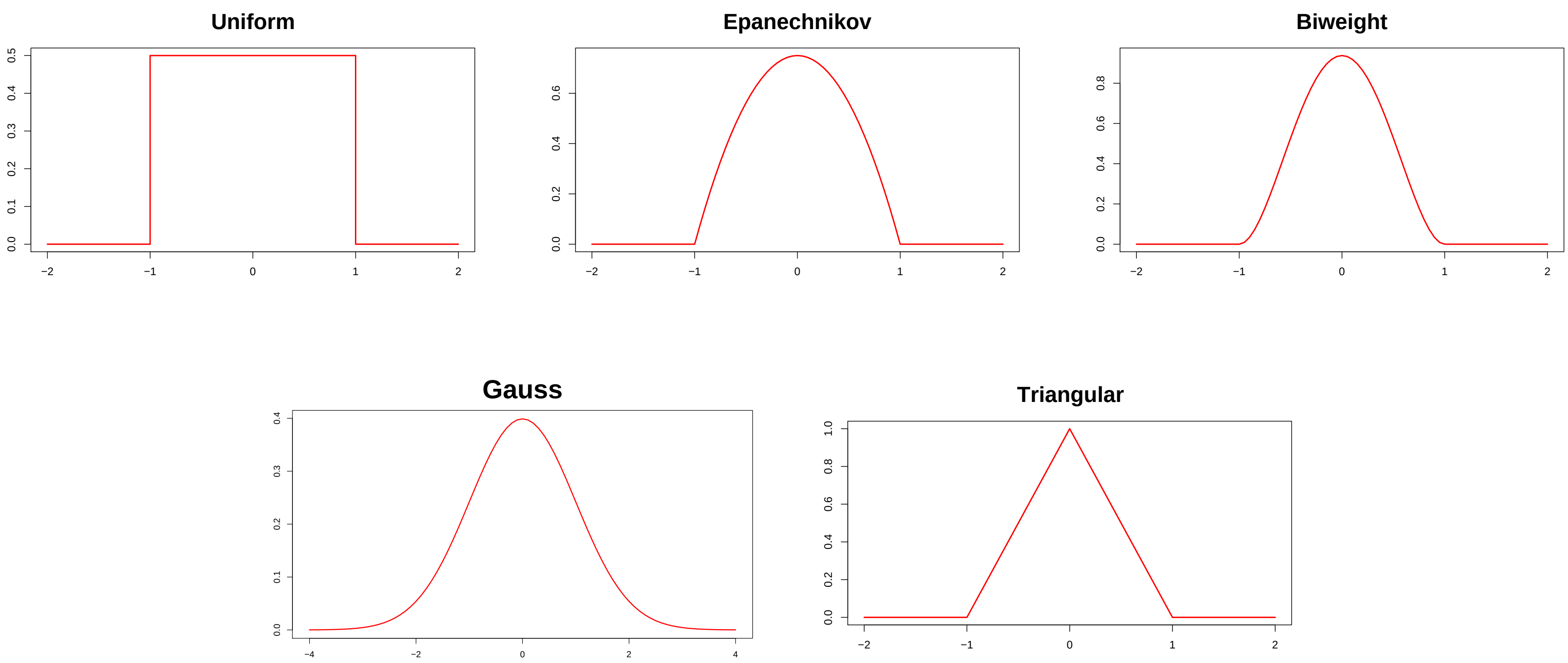
(Source for KDE diagrams: Stefanie Scheid - Introduction to Kernel Smoothing)

Another popular smoother is kernel density estimation (KDE)

KDE treats each data point as the center of a “kernel”, and adds those kernels up

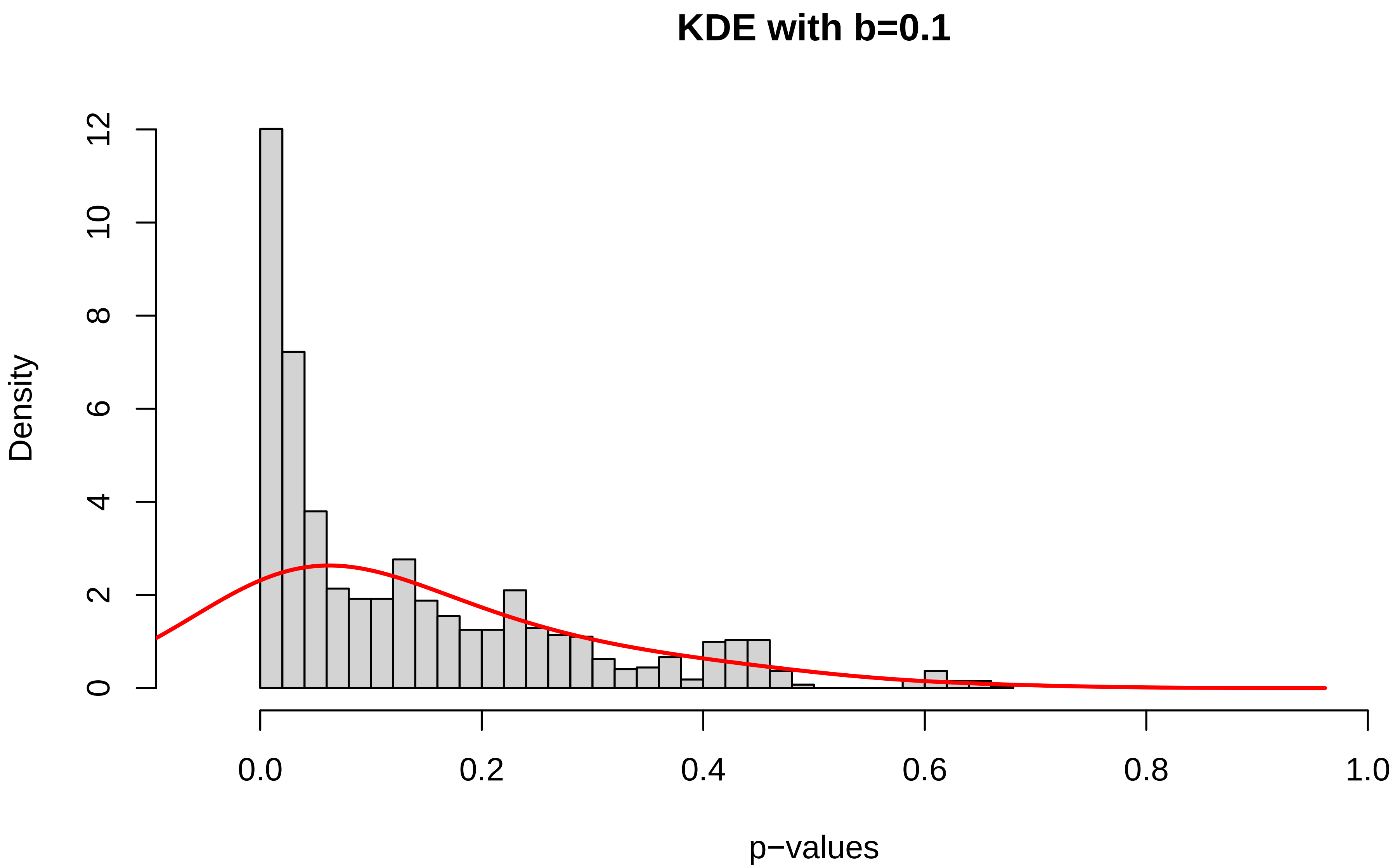
This let’s the effect of each datapoint “smooth out”.
Shape of smoothing out is the kernel; degree of smoothing is the bandwidth

Choices of kernels to place around datapoints



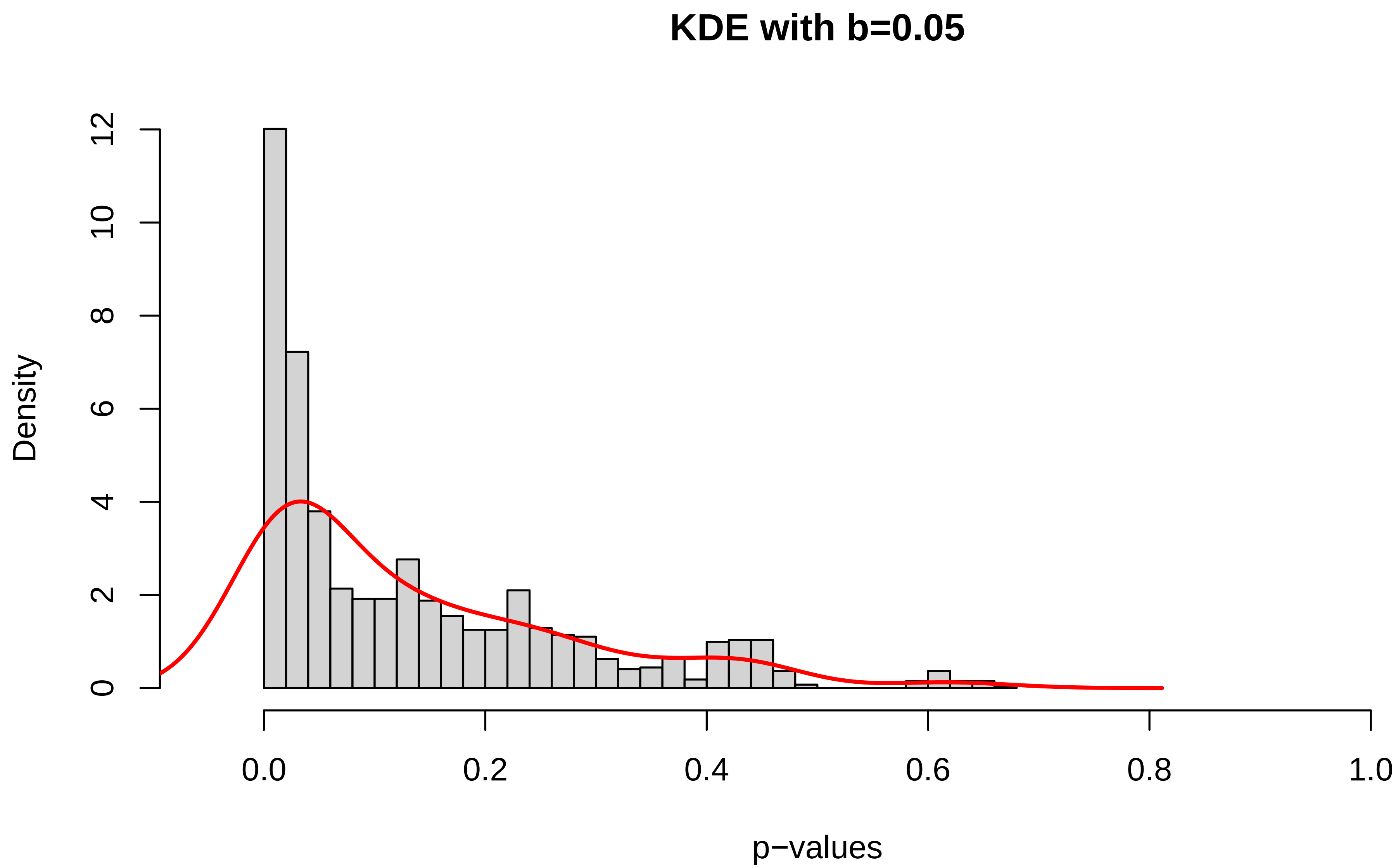
Epanechnikov minimizes error. But not usually important which you choose.

Smoothing a histogram



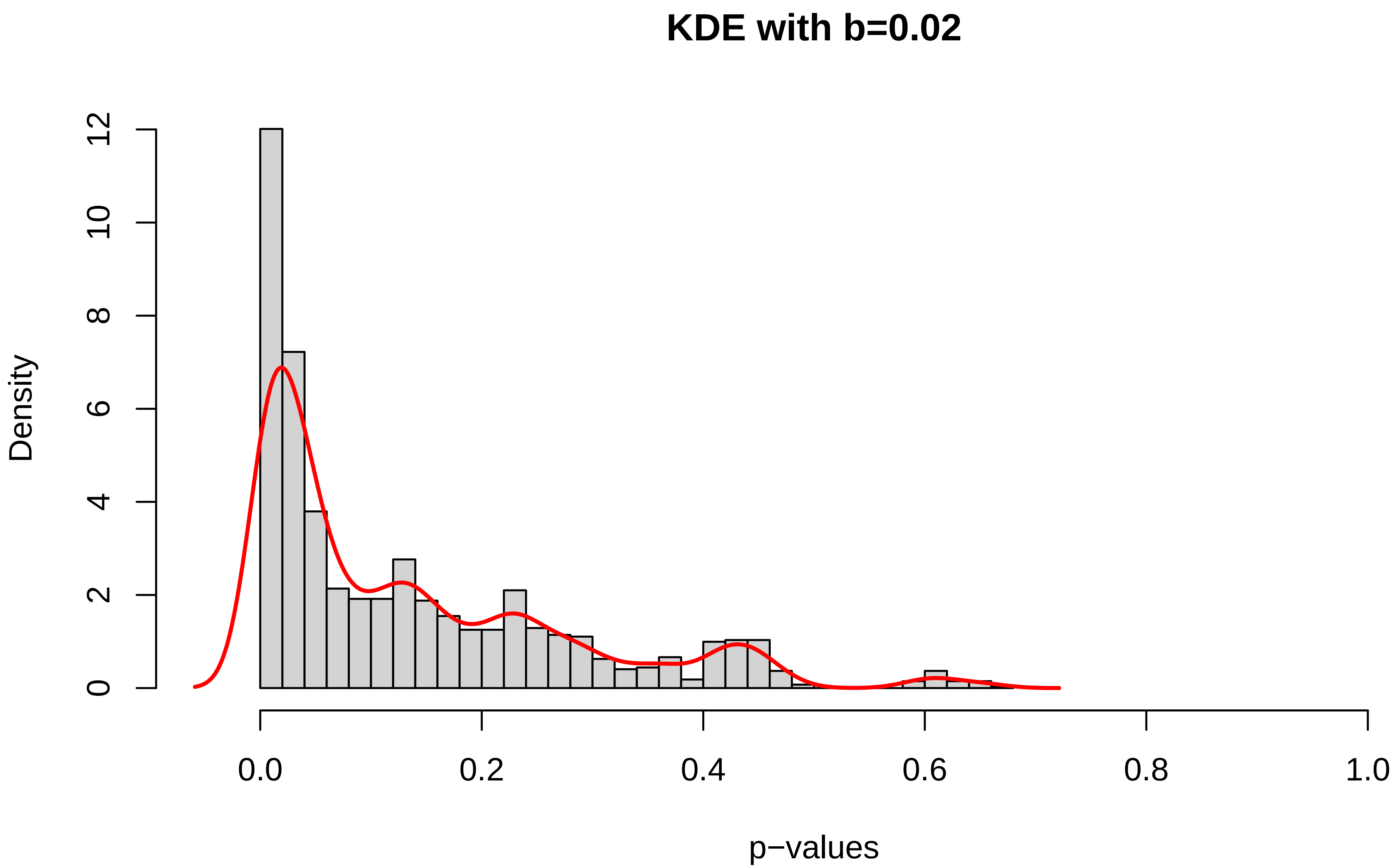
Bandwith *is* an important choice, as with any smoother.

Smoothing a histogram



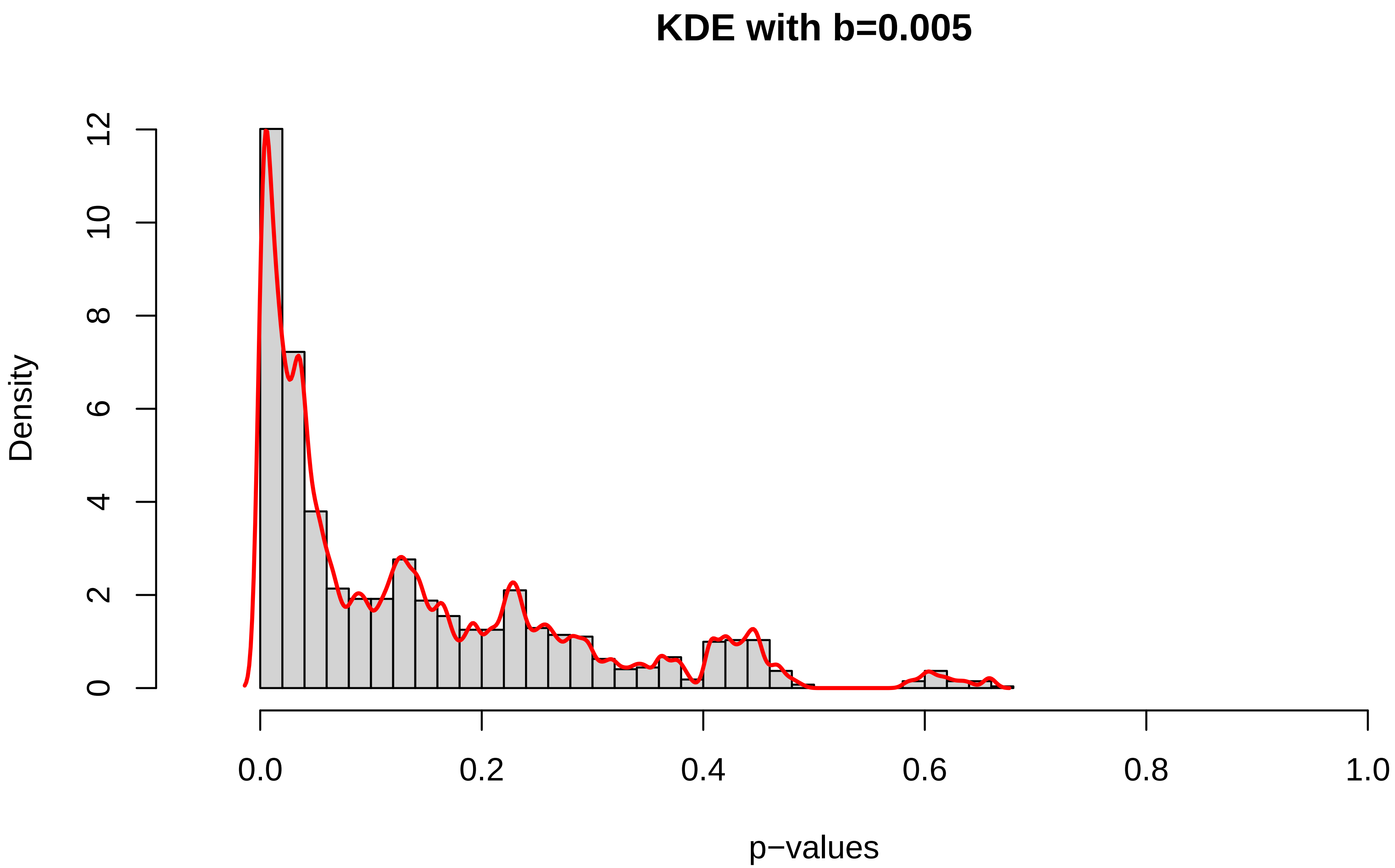
Bandwith *is* an important choice, as with any smoother.

Smoothing a histogram



Bandwith *is* an important choice, as with any smoother.

Smoothing a histogram



Bandwith *is* an important choice, as with any smoother.

Uses of KDE

KDE is very useful for smoothing histograms of all kinds

If you draw from the predictive or posterior distribution of a model,
you can smooth with KDE

This works even in two dimensions
(e.g., you want the joint confidence region of two parameters)

Generally, if you have a 2D histogram, and you want contours, consider KDE

In `R`, see the `density()` command for one dimensional KDE

See `kde2d` in the `MASS` library for 2D version

The curse of dimensionality

The biggest problem in visual display is that humans can only perceive 3 dimensions

Most data, esp in social sciences, have many variables; potentially many dimensions

Paper & computer screens are even more limiting: 2D

Three approaches to fighting the curse of dimensionality:

- Cleverly display all the data.
- Use models to reduce the number of dimensions
- Use time & motion

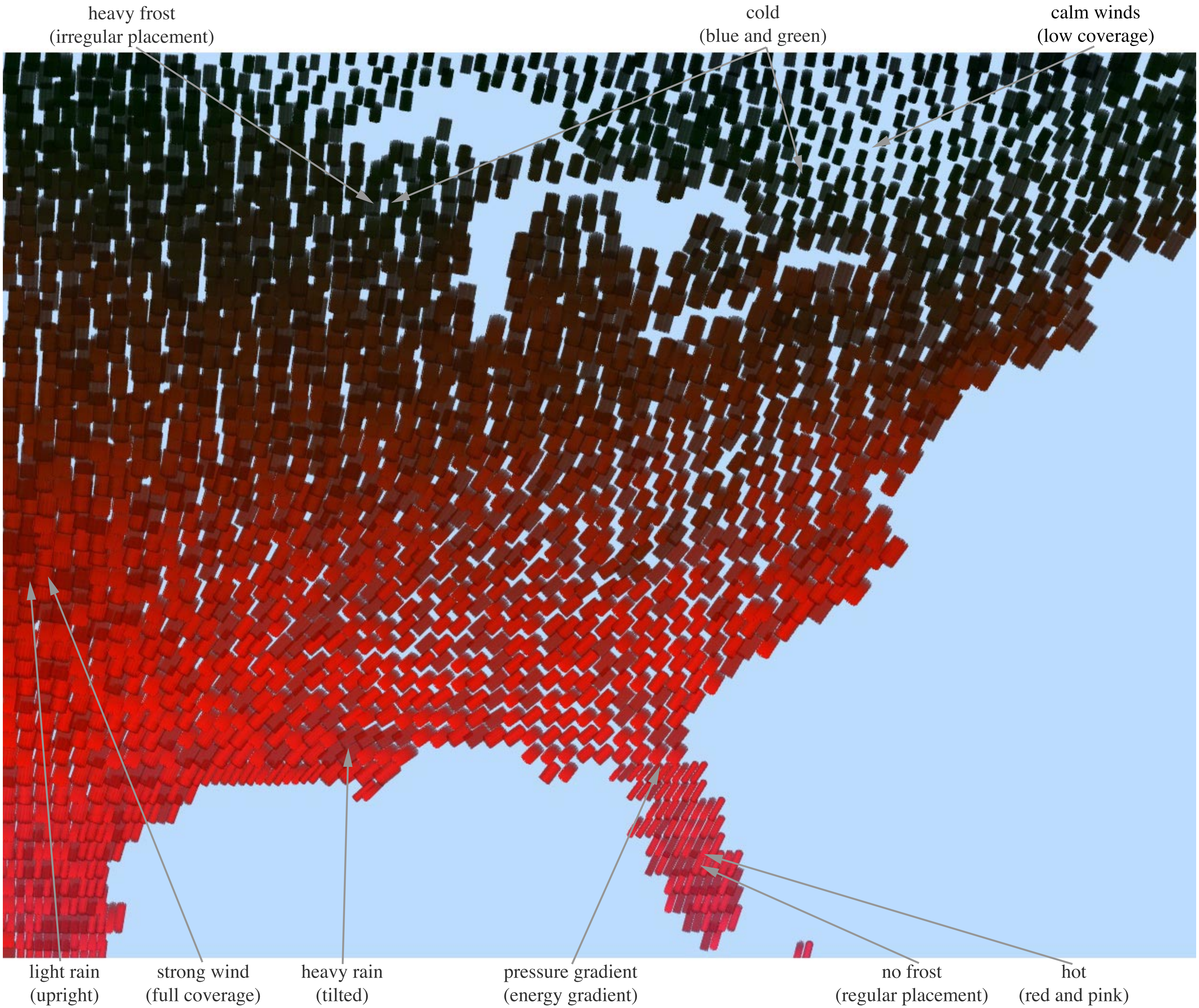
Clever display of the whole dataset

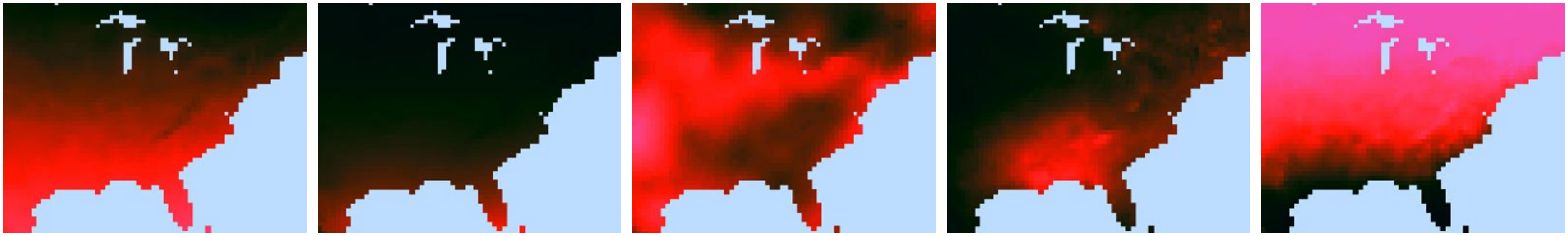
This is what we've done all quarter:

1. Small multiples
2. Glyphs
3. Layer pre-attentive elements

We've seen lots of examples.

Here is a final example that pushes the envelope


heavy frost
(irregular placement)
cold
(blue and green)
calm winds
(low coverage)
light rain
(upright)
strong wind
(full coverage)
heavy rain
(tilted)
pressure gradient
(energy gradient)
no frost
(regular placement)
hot
(red and pink)



temp *pressure* *wind* *precip* *frost*

Source: Christopher G. Healey, "Combining Perception and Impressionist Techniques for Nonphotorealistic Visualization of Multidimensional Data", *SIGGRAPH 2001 Course 32: Nonphotorealistic Rendering in Scientific Visualization* 2001, `http://www.csc.ncsu.edu/faculty/healey/download/sig-course.01.pdf`

Does the 5-dimensional version work "better" than the small multiples?

For look-up, no. But for gestalt impressions, perhaps worth looking at both.

I've experimented with the same 5 dimensions independently. Probably near the limit for a single diagram. Challenging to process.

So let's try to reduce dimensions while keeping most of the data

Principal Components Analysis (PCA)

PCA is related to factor analysis and multidimensional scaling

Details involve lots of linear algebra; instead, a conceptual summary

Imagine a dataset with k continuous variables plotted in k-space

Principal Components Analysis (PCA)

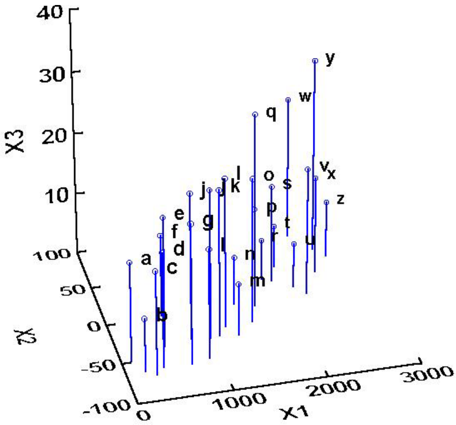
PCA is related to factor analysis and multidimensional scaling

Details involve lots of linear algebra; instead, a conceptual summary

Imagine a dataset with k continuous variables plotted in k-space

For example, $k = 3$

Note the different scales for each variable



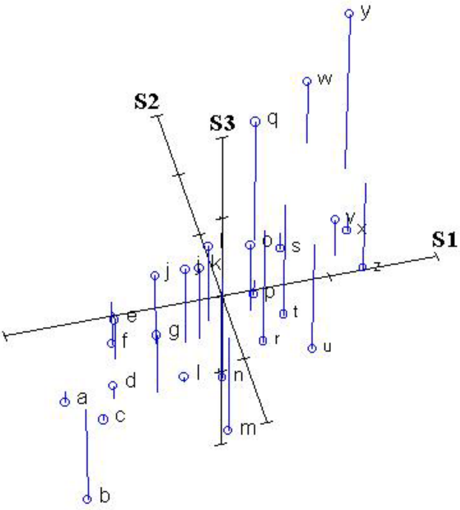
Source: Michael Palmer,

ordination.okstate.edu/PCA.htm

Principal Components Analysis (PCA)

It will help to normalize these variables to have mean zero and unit variance

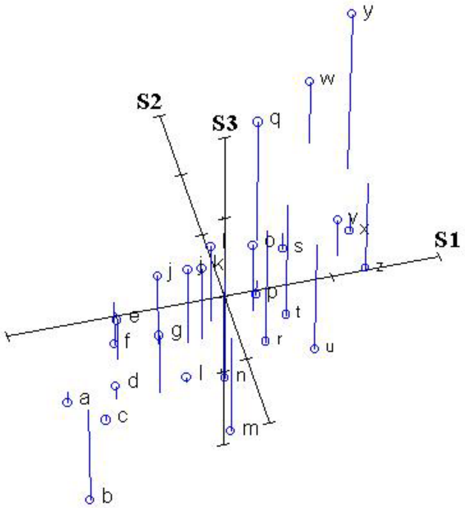
PCA finds a new set of dimensions, $\leq k$, that best explain and separate the variance in these data



It will help to normalize these variables to have mean zero and unit variance

PCA finds a new set of dimensions, $\leq k$, that best explain and separate the variance in these data

We search for new orthogonal axes (components), which we label:



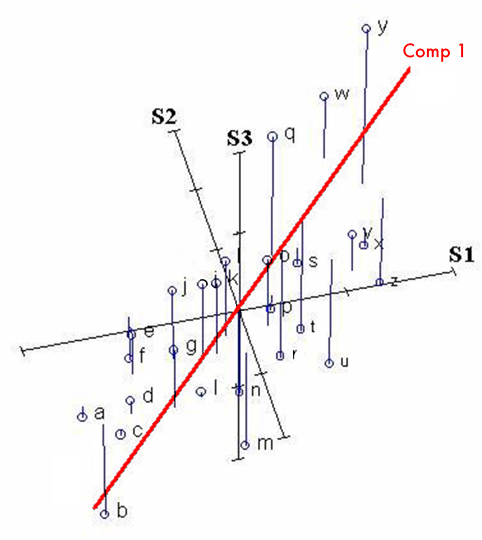
Principal Components Analysis (PCA)

It will help to normalize these variables to have mean zero and unit variance

PCA finds a new set of dimensions, $\leq k$, that best explain and separate the variance in these data

We search for new orthogonal axes (components), which we label:

Component 1



Principal Components Analysis (PCA)

It will help to normalize these variables to have mean zero and unit variance

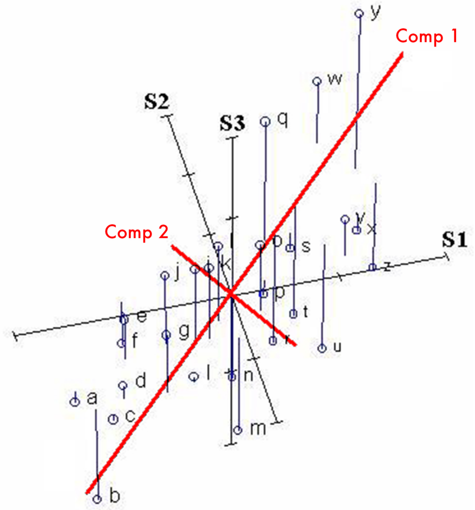
PCA finds a new set of dimensions, $\leq k$, that best explain and separate the variance in these data

We search for new orthogonal axes (components), which we label:

Component 1

Component 2

etc.

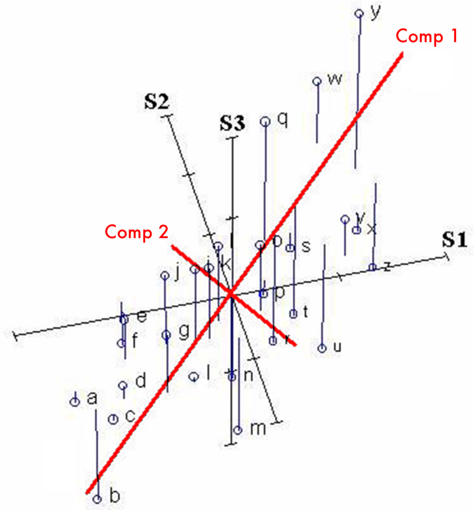


Principal Components Analysis (PCA)

The first principal component is the axis that explains the most variation in the data

The second (third, etc.) principal component is the line orthogonal to the prior components that explains the greatest part of the remaining variation

Two principal components are often (not always) a nearly complete summary of variation on the *k* original dimensions



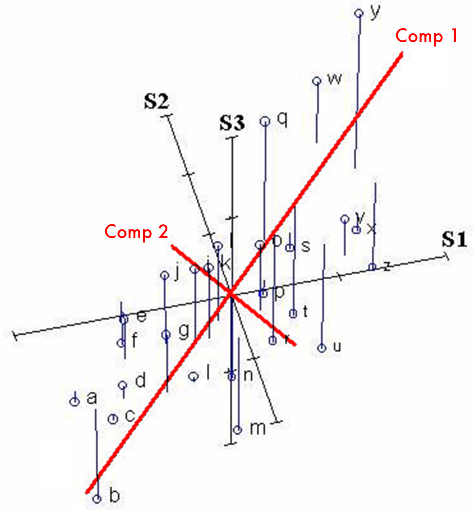
Principal Components Analysis (PCA)

Each principal component can be seen as a linear combination of the original *k* axes

Very useful lower dimensional replacements for multidimensional data

Widely used to create "index" variables

Less arbitrary – and more informative – than averaging the underlying variables



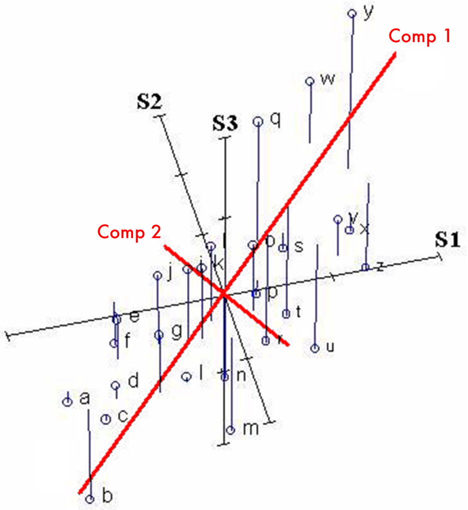
Principal Components Analysis (PCA)

Note the graph at right has *two sets* of coordinate systems

The original variables are currently the plot coordinates, while the PCA coordinates are "plotted" as if they were data

What if we *switched* these coordinate systems, so the principal components are the plot coordinates?

And the original data and original axes are "plotted" in PCA space?

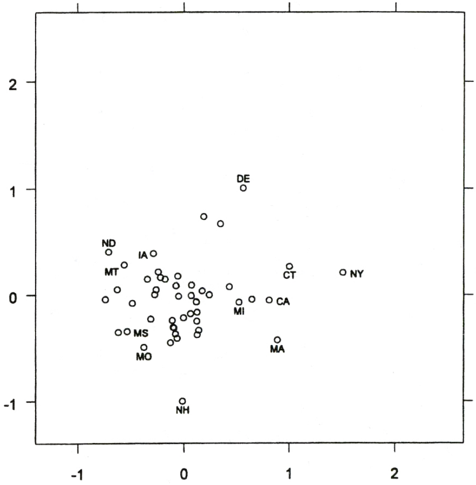


Example: FY 1992 state budget priorities

E.g., suppose a state divided its budget:

- 8% highways
- 30% education
- 35% health
- 12% corrections
- 3% law enforcement
- 12% welfare

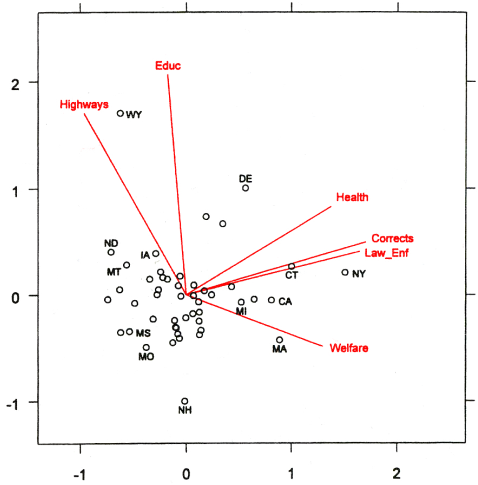
We wish to plot one point for each of the 50 states, but that takes 6 dimensions!



Solution: Use PCA to reduce the 6 budget dimensions to 2 principal components

Plot the 50 states locations in 2d PCA space

Source: William G. Jacoby, *Statistical Graphics for Visualizing Multivariate Data*, Sage Paper 07-120



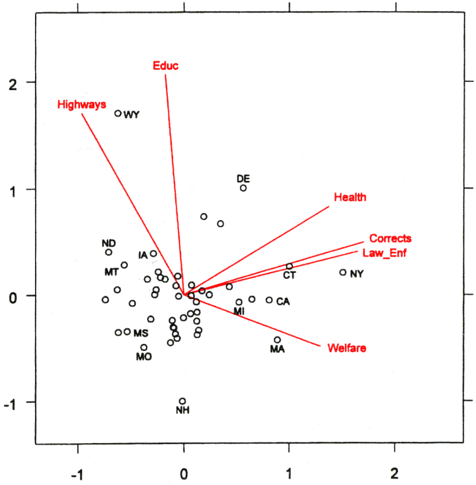
Solution: Use PCA to reduce the 6 budget dimensions to 2 principal components

Plot the 50 states locations in 2d PCA space

Then *add* to the plot the projection into 2d PCA space of the original 6 budget dimensions

This simultaneous plot of dimensions and data is called the **biplot**

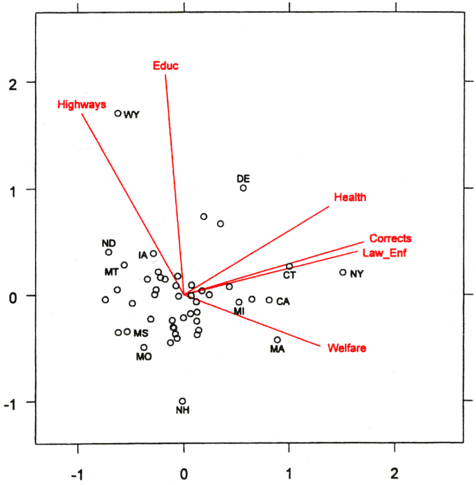
Source: William G. Jacoby, *Statistical Graphics for Visualizing Multivariate Data*, Sage Paper 07-120



Source: William G. Jacoby, *Statistical Graphics for Visualizing Multivariate Data*, Sage Paper 07-120

Tips for interpreting biplots:

Overlapping dimensions are perfectly correlated

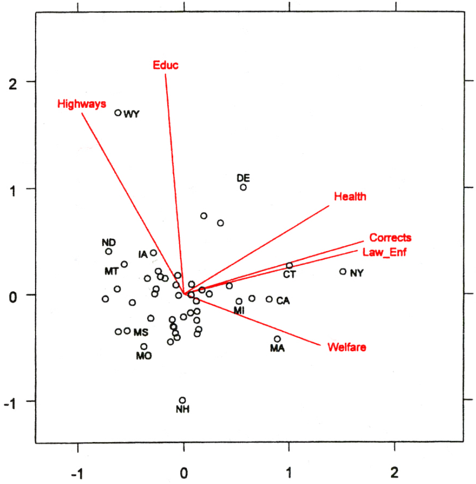


Tips for interpreting biplots:

Overlapping dimensions are perfectly correlated

Dimensions form an acute angle $\Rightarrow$ somewhat positive correlation

Source: William G. Jacoby, *Statistical Graphics for Visualizing Multivariate Data*, Sage Paper 07-120



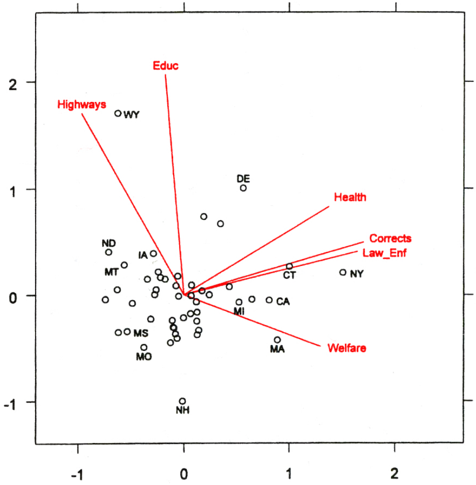
Source: William G. Jacoby, *Statistical Graphics for Visualizing Multivariate Data*, Sage Paper 07-120

Tips for interpreting biplots:

Overlapping dimensions are perfectly correlated

Dimensions form an acute angle
$\Rightarrow$ somewhat positive correlation

Dimensions form a 90° angle
$\Rightarrow$ orthogonal



Tips for interpreting biplots:

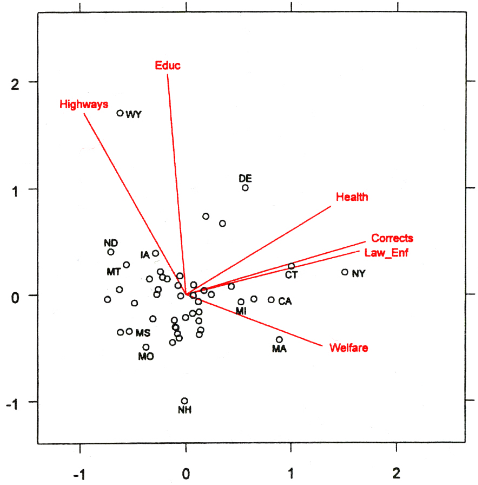
Overlapping dimensions are perfectly correlated

Dimensions form an acute angle $\Rightarrow$ somewhat positive correlation

Dimensions form a 90° angle $\Rightarrow$ orthogonal

Dimensions form an obtuse angle $\Rightarrow$ somewhat negative correlation

Source: William G. Jacoby, *Statistical Graphics for Visualizing Multivariate Data*, Sage Paper 07-120



Tips for interpreting biplots:

Overlapping dimensions are perfectly correlated

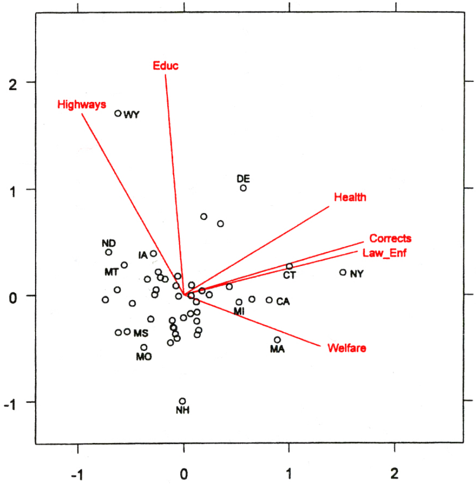
Dimensions form an acute angle
$\Rightarrow$ somewhat positive correlation

Dimensions form a 90° angle
$\Rightarrow$ orthogonal

Dimensions form an obtuse angle
$\Rightarrow$ somewhat negative correlation

Dimensions form a 180° line
$\Rightarrow$ perfect inverse correlation

Source: William G. Jacoby, *Statistical Graphics for Visualizing Multivariate Data*, Sage Paper 07-120



More tips for interpreting biplots:

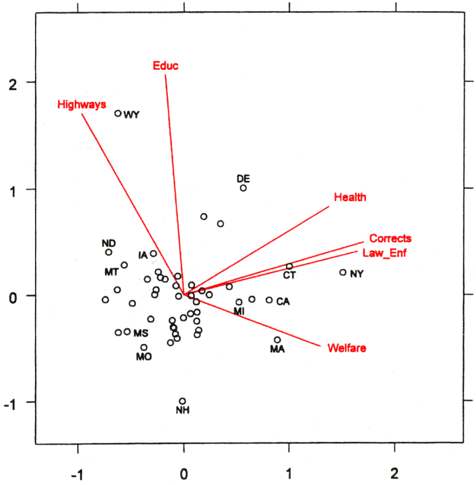
Dimensions meet at the origin (0,0). (Why?)

"Short" dimensions load poorly on components. (Why?)

Data far from the origin are outliers

Distances between points are Mahalanobis distances (variables standardized to have unit variances)

Source: William G. Jacoby, *Statistical Graphics for Visualizing Multivariate Data*, Sage Paper 07-120



Source: William G. Jacoby, *Statistical Graphics for Visualizing Multivariate Data*, Sage Paper 07-120

How to do this in R:

```
res <- princomp(~ health
                 + correct
                 + lawenf
                 + welfare
                 + highways
                 + educ)
biplot(res)
```

`biplot()` has several options, but is surprisingly inflexible

Emily Kalah Gade (UW–Political Science) provides data on the number of documents published on federal websites mentioning topics related to climate change

We have counts of the number of documents mentioning each *topic* by each *agency* by year

How could we visualize these data?

Topics (Dimensions)

```
freshwater
pollution
IPCC
global warming
food security
climate change
natural disaster
greenhouse gas
anthropogenic
desertification
forest conservation
security of food
climate research unit
ocean acidification
anthropocene
climategate
```

Agencies (Cases)

```
usda.gov
house.gov
ed.gov
hhs.gov
doi.gov
senate.gov
dot.gov
whitehouse.gov
usdoj.gov
va.gov
dol.gov
state.gov
treasury.gov
energy.gov
dhs.gov
commerce.gov
defense.gov
dod.gov
other
```

How could we visualize these data?

Topics (Dimensions)	Agencies (Cases)
freshwater	usda.gov
pollution	house.gov
IPCC	ed.gov
global warming	hhs.gov
food security	doi.gov
climate change	senate.gov
natural disaster	dot.gov
greenhouse gas	whitehouse.gov
anthropogenic	usdoj.gov
desertification	va.gov
forest conservation	dol.gov
security of food	state.gov
climate research unit	treasury.gov
ocean acidification	energy.gov
anthropocene	dhs.gov
climategate	commerce.gov
	defense.gov
	dod.gov
	other

How could we visualize these data?

1. Could make an agency $\times$ topic contingency table.

Hard to read

Topics (Dimensions)	Agencies (Cases)
freshwater	usda.gov
pollution	house.gov
IPCC	ed.gov
global warming	hhs.gov
food security	doi.gov
climate change	senate.gov
natural disaster	dot.gov
greenhouse gas	whitehouse.gov
anthropogenic	usdoj.gov
desertification	va.gov
forest conservation	dol.gov
security of food	state.gov
climate research unit	treasury.gov
ocean acidification	energy.gov
anthropocene	dhs.gov
climategate	commerce.gov
	defense.gov
	dod.gov
	other

How could we visualize these data?

1. Could make an agency $\times$ topic contingency table.

Hard to read

2. Could make a heatmap, as we did with trade flows.

Topics (Dimensions)

freshwater
pollution
IPCC
global warming
food security
climate change
natural disaster
greenhouse gas
anthropogenic
desertification
forest conservation
security of food
climate research unit
ocean acidification
anthropocene
climategate

Agencies (Cases)

usda.gov
house.gov
ed.gov
hhs.gov
doi.gov
senate.gov
dot.gov
whitehouse.gov
usdoj.gov
va.gov
dol.gov
state.gov
treasury.gov
energy.gov
dhs.gov
commerce.gov
defense.gov
dod.gov
other

How could we visualize these data?

1. Could make an agency $\times$ topic contingency table.

Hard to read

2. Could make a heatmap, as we did with trade flows.

Can we instead use a biplot to reduce the number of plotted dimensions to 2?

YES, using **correspondence analysis**, an analog of PCA for contingency tables

Topics (Dimensions)	Agencies (Cases)
`freshwater`	`usda.gov`
`pollution`	`house.gov`
`IPCC`	`ed.gov`
`global warming`	`hhs.gov`
`food security`	`doi.gov`
`climate change`	`senate.gov`
`natural disaster`	`dot.gov`
`greenhouse gas`	`whitehouse.gov`
`anthropogenic`	`usdoj.gov`
`desertification`	`va.gov`
`forest conservation`	`dol.gov`
`security of food`	`state.gov`
`climate research unit`	`treasury.gov`
`ocean acidification`	`energy.gov`
`anthropocene`	`dhs.gov`
`climategate`	`commerce.gov`
	`defense.gov`
	`dod.gov`
	`other`

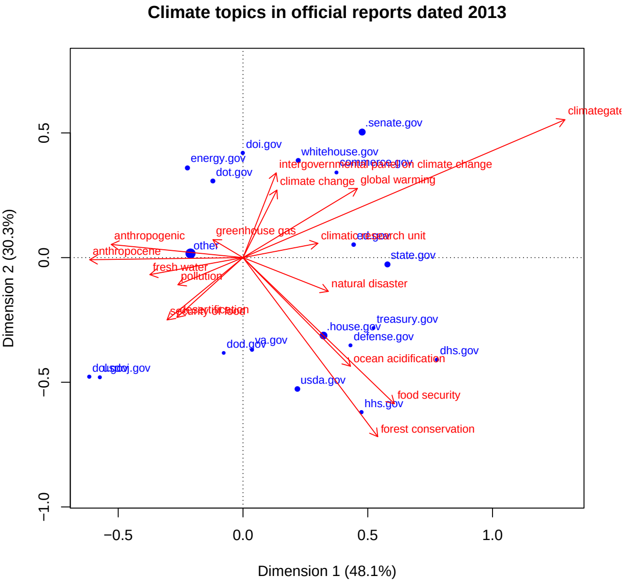
Just as with PCA, a correspondence analysis biplot shows a 2D view of a high dimensional space

Here we see a cloud of points relative to 14(!) axes

Which topics (dimensions) are similar?

Which agencies (points) are similar?

Which agencies load strongly on which dimensions? How sure are you of this?

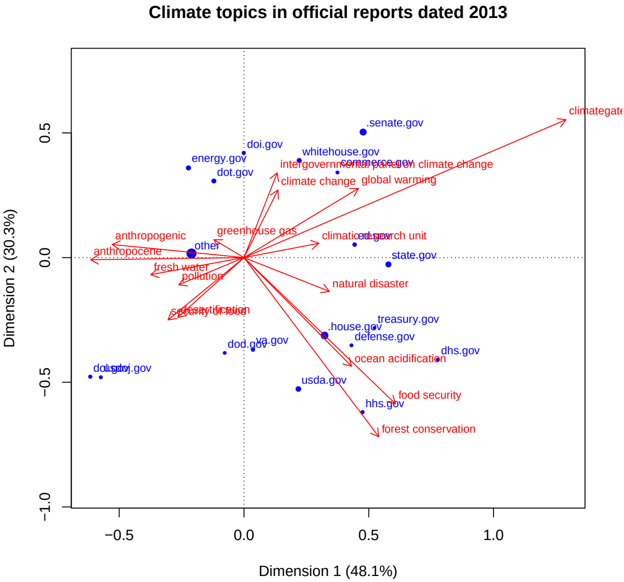


Data: Emily Kalah Gade (UW)

Figure: Will Lowe (Univ. of Mannheim)

How to do it in R:

```
# Roughly, code is:
res <- ca(data)
plot(res,
     mass=c(TRUE, TRUE)
     )

# but see plot.ca()
# for more options
# -- better than
#    biplot()
```



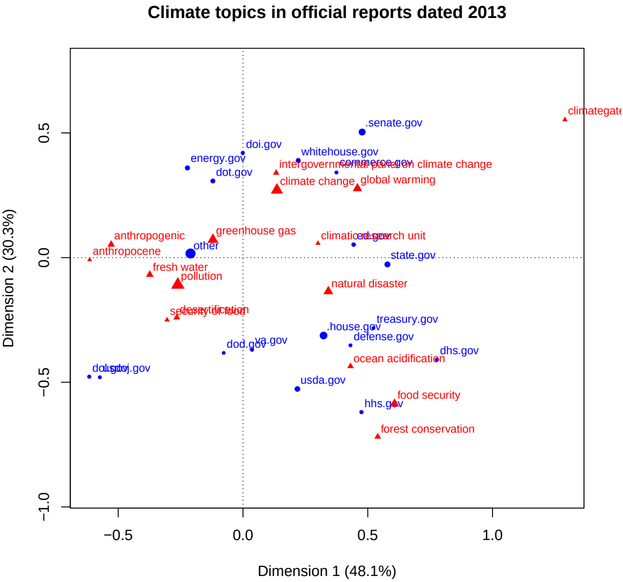
Data: Emily Kalah Gade (UW)

Figure: Will Lowe (Univ. of Mannheim)

With large numbers of dimensions, the arrows can get distracting

Once you understand the biplot, you don't need them

Not an easy plot to explain, but a powerful way to explore many dimensions of data at once



Data: Emily Kalah Gade (UW)

Figure: Will Lowe (Univ. of Mannheim)