

# Generalized Additive Models

David L Miller

# Overview

- What is a GAM?
- What is smoothing?
- How do GAMs work?
- Fitting GAMs using dsm
- Model checking

# What is a GAM?

”gam”

1. *Collective noun used to refer to a group of whales, or rarely also of porpoises; a pod.*
2. *(by extension) A social gathering of whalers (whaling ships).*

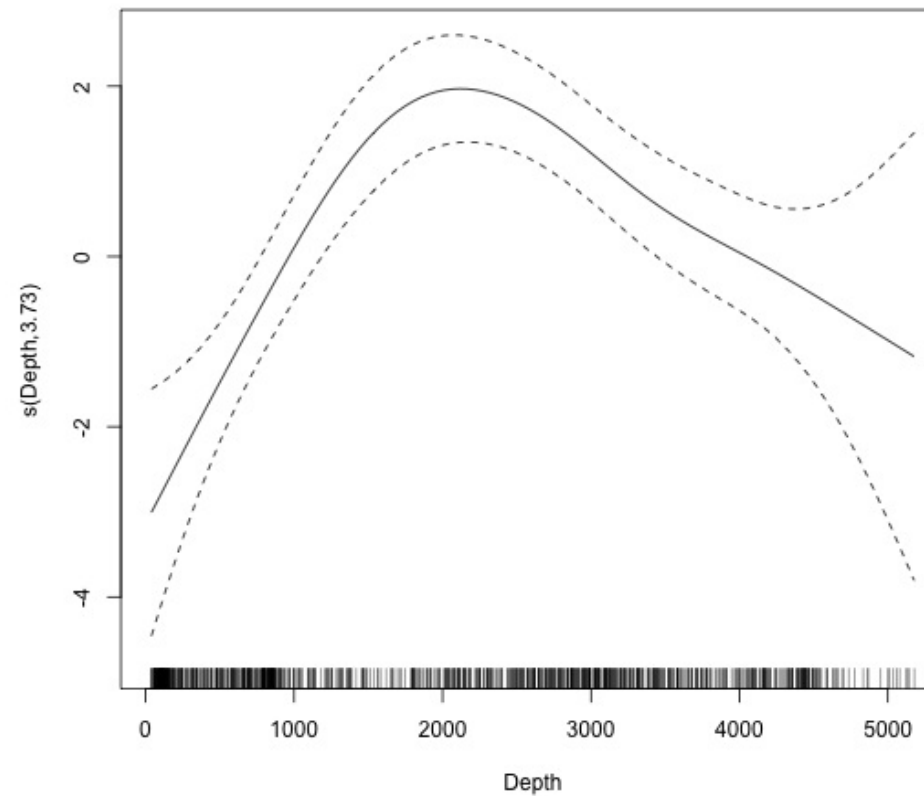
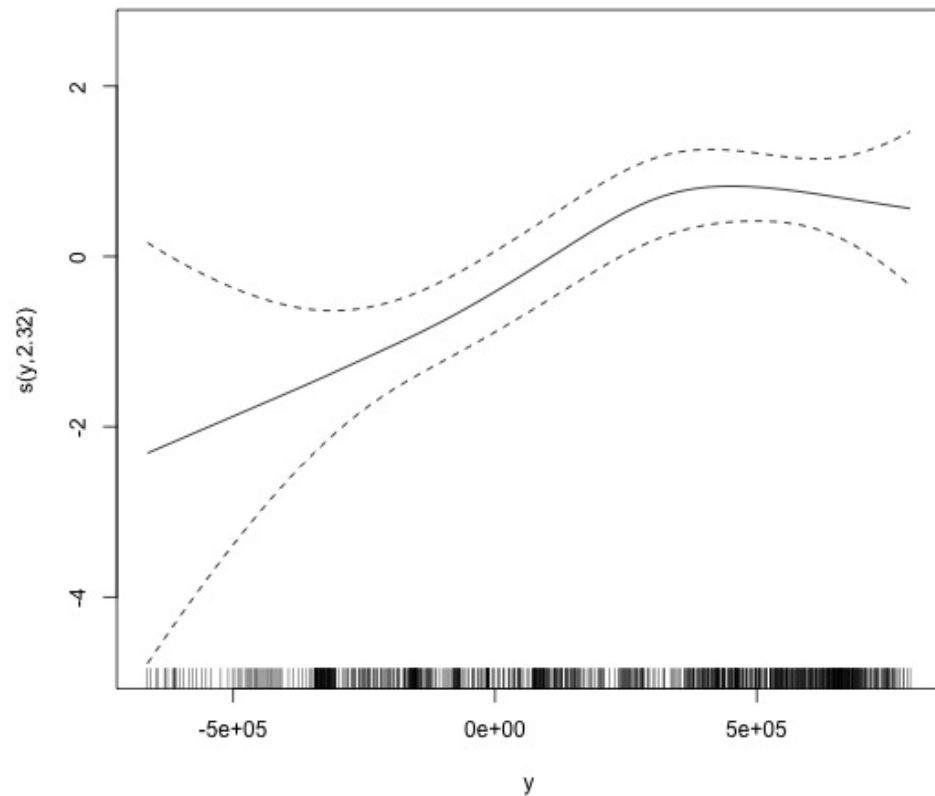
(via Natalie Kelly, CSIRO. Seen in Moby Dick.)

# Generalized Additive Models

- Generalized: many response distributions
- Additive: terms **add** together
- Models: well, it's a model...

# What does a model look like?

- Count  $n_j$  distributed according to some count distribution
- Model as sum of terms



# Mathematically...

Taking the previous example...

$$n_j = A_j \hat{p}_j \exp \left[ \beta_0 + s(y_j) + s(\text{Depth}_j) \right] + \epsilon_j$$

where  $\epsilon_j \sim N(0, \sigma)$ ,  $n_j \sim \text{count distribution}$

# Mathematically...

Taking the previous example...

$$n_j = A_j \hat{p}_j \exp \left[ \beta_0 + s(y_j) + s(\text{Depth}_j) \right] + \epsilon_j$$

where  $\epsilon_j \sim N(0, \sigma)$ ,  $n_j \sim \text{count distribution}$

- area of segment - offset
- probability of detection in segment
- link function
- model terms

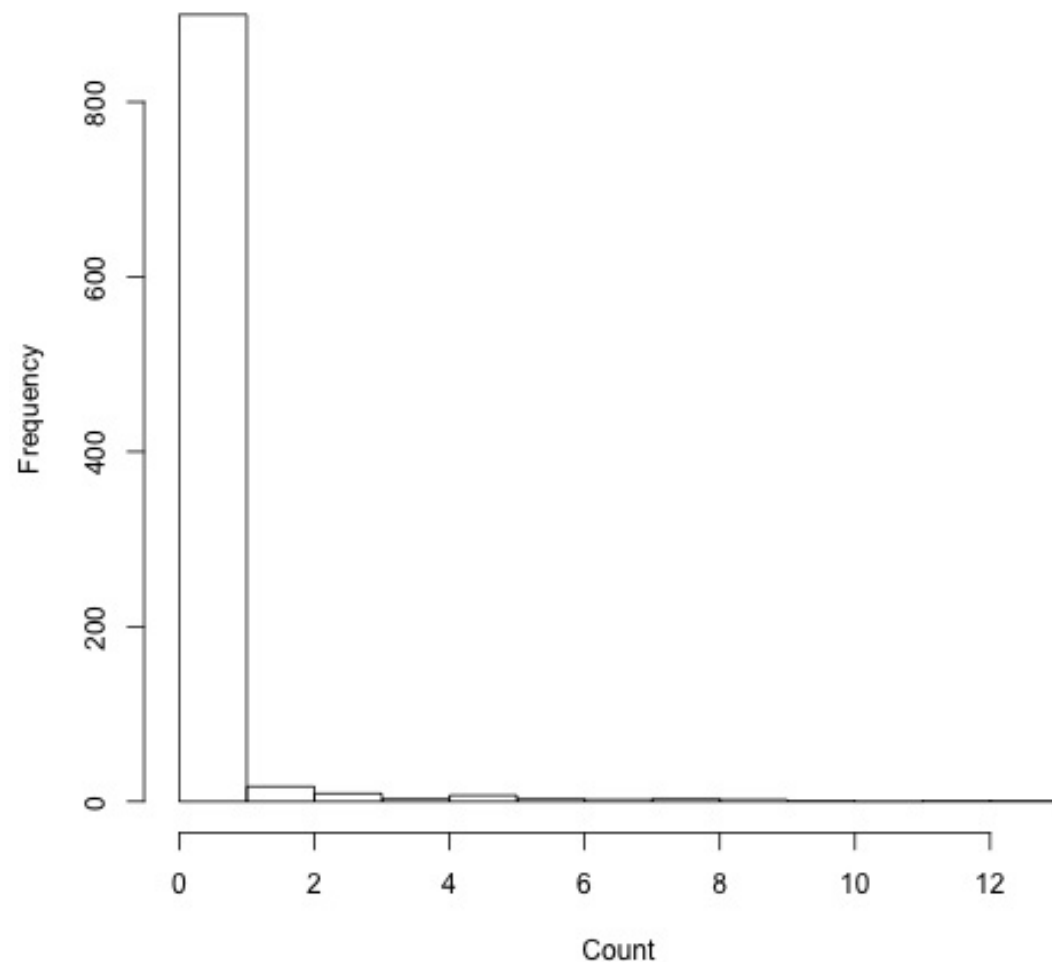


# Response

$$\mathbf{n}_j = A_j \hat{p}_j \exp [\beta_0 + s(y_j) + s(\text{Depth}_j)] + \epsilon_j$$

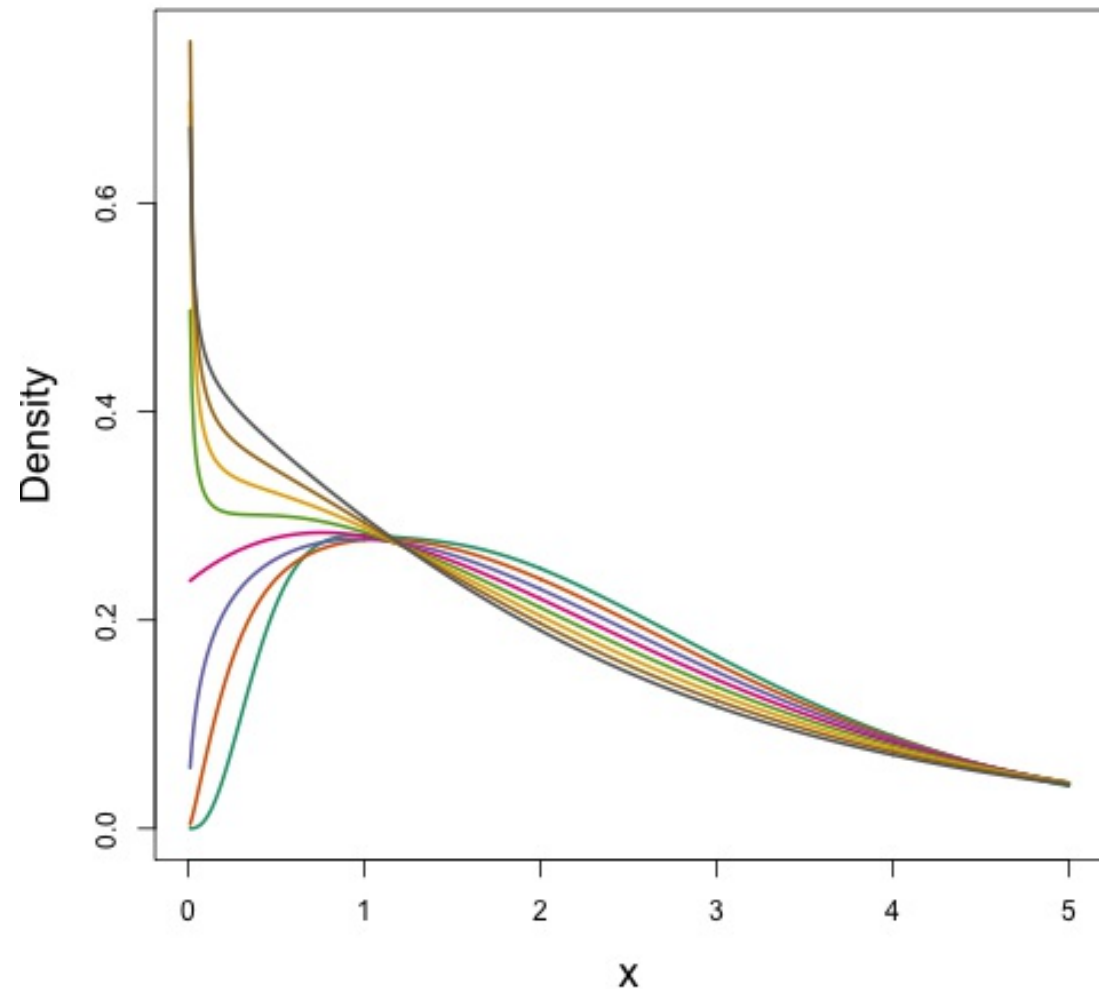
where  $\epsilon_j \sim N(0, \sigma)$ ,  $\mathbf{n}_j \sim \text{count distribution}$

# Count distributions



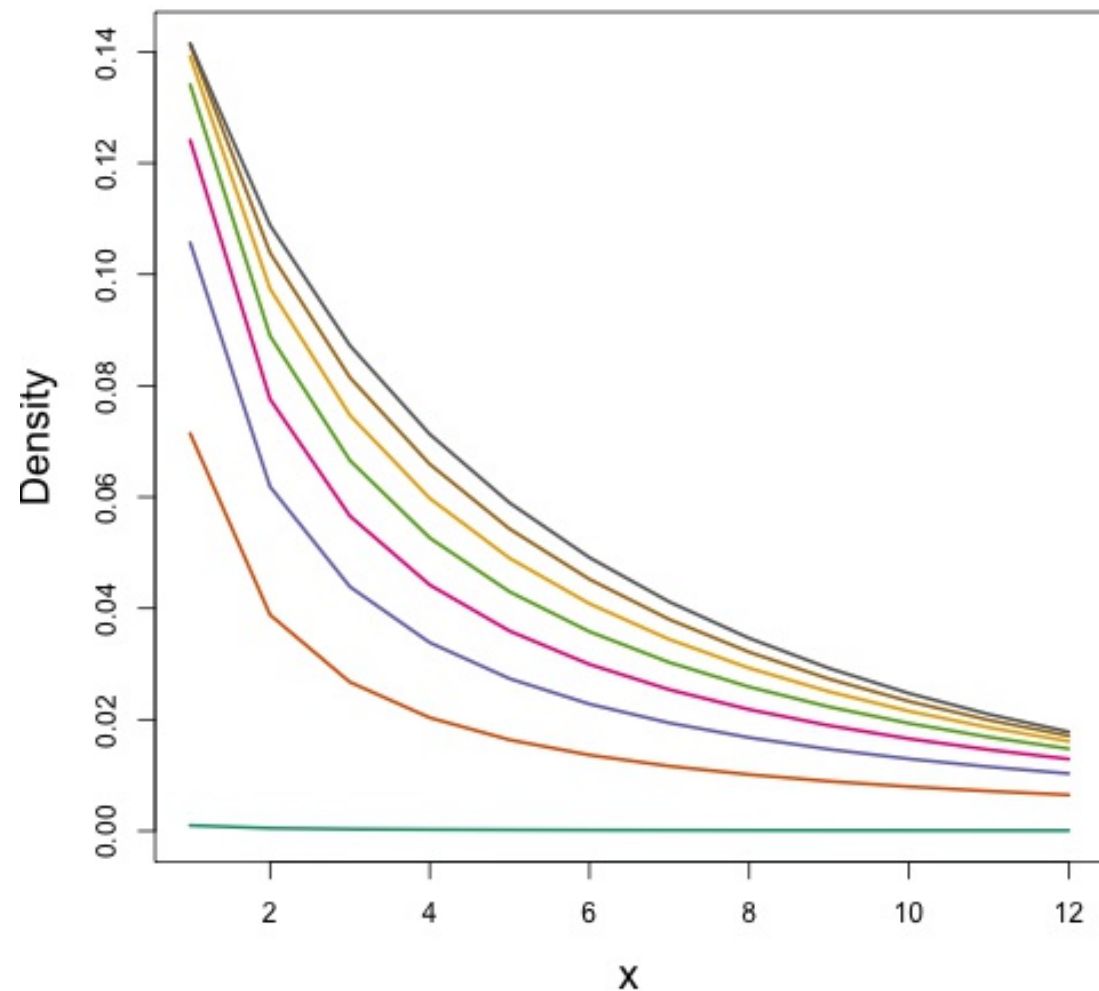
- Response is a count (not not always integer)
- Often, it's mostly zero (that's complicated)
- Want response distribution that deals with that
- Flexible mean-variance relationship

# Tweedie distribution



- $\text{Var}(\text{count}) = \varphi(\text{count})^q$
- Common distributions are sub-cases:
  - $q = 1 \Rightarrow \text{Poisson}$
  - $q = 2 \Rightarrow \text{Gamma}$
  - $q = 3 \Rightarrow \text{Normal}$
- We are interested in  $1 < q < 2$
- (here  $q = 1.2, 1.3, \dots, 1.9$ )

# Negative binomial



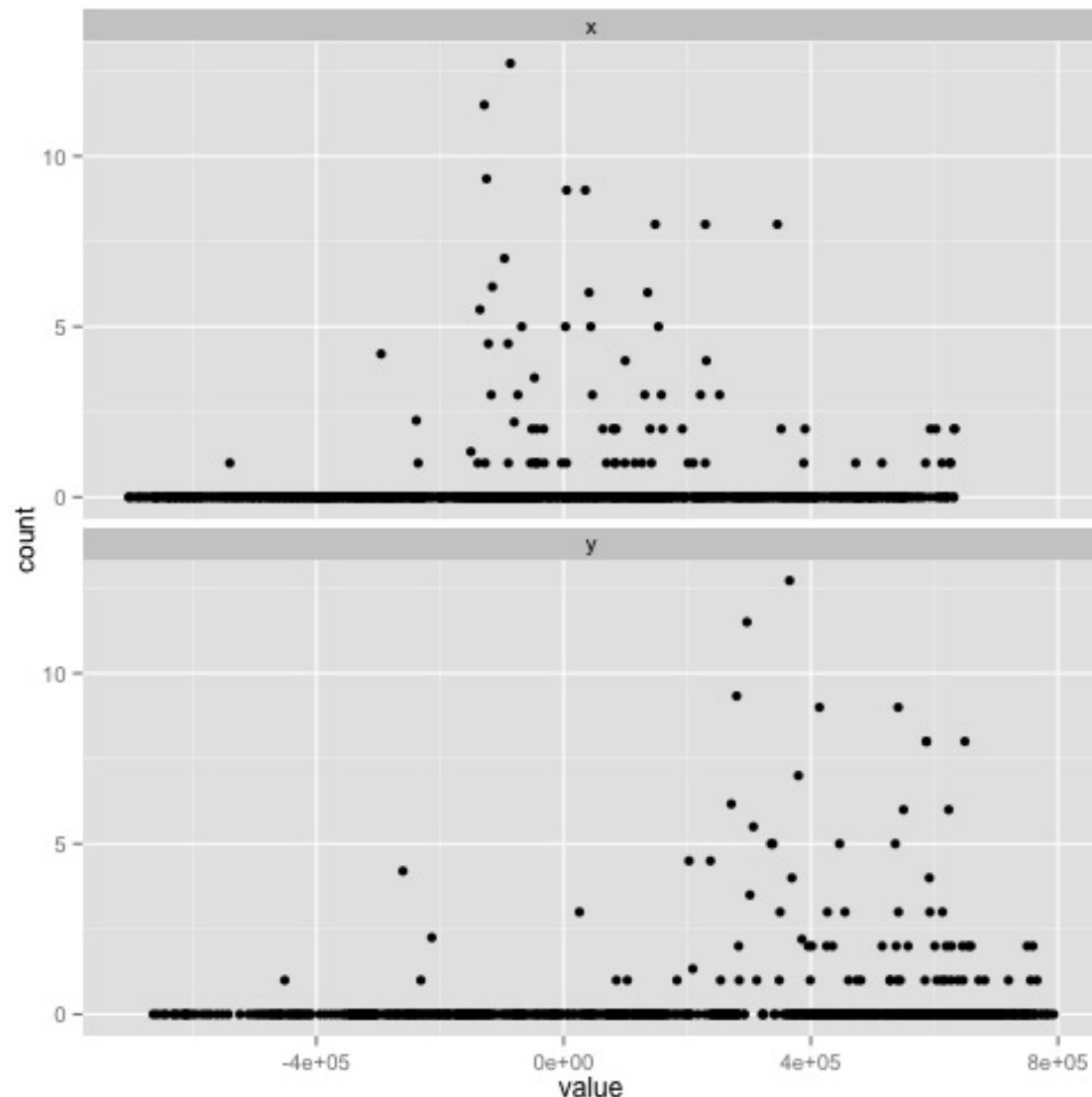
- $\text{Var}(\text{count}) = (\text{count}) + \kappa(\text{count})^2$
- Estimate  $\kappa$
- Is quadratic relationship a “strong” assumption?
- Similar to Poisson:  
 $\text{Var}(\text{count}) = (\text{count})$

# Smooth terms

$$n_j = A_j \hat{p}_j \exp \left[ \beta_0 + s(y_j) + s(\text{Depth}_j) \right] + \epsilon_j$$

where  $\epsilon_j \sim N(0, \sigma)$ ,  $n_j \sim \text{count distribution}$

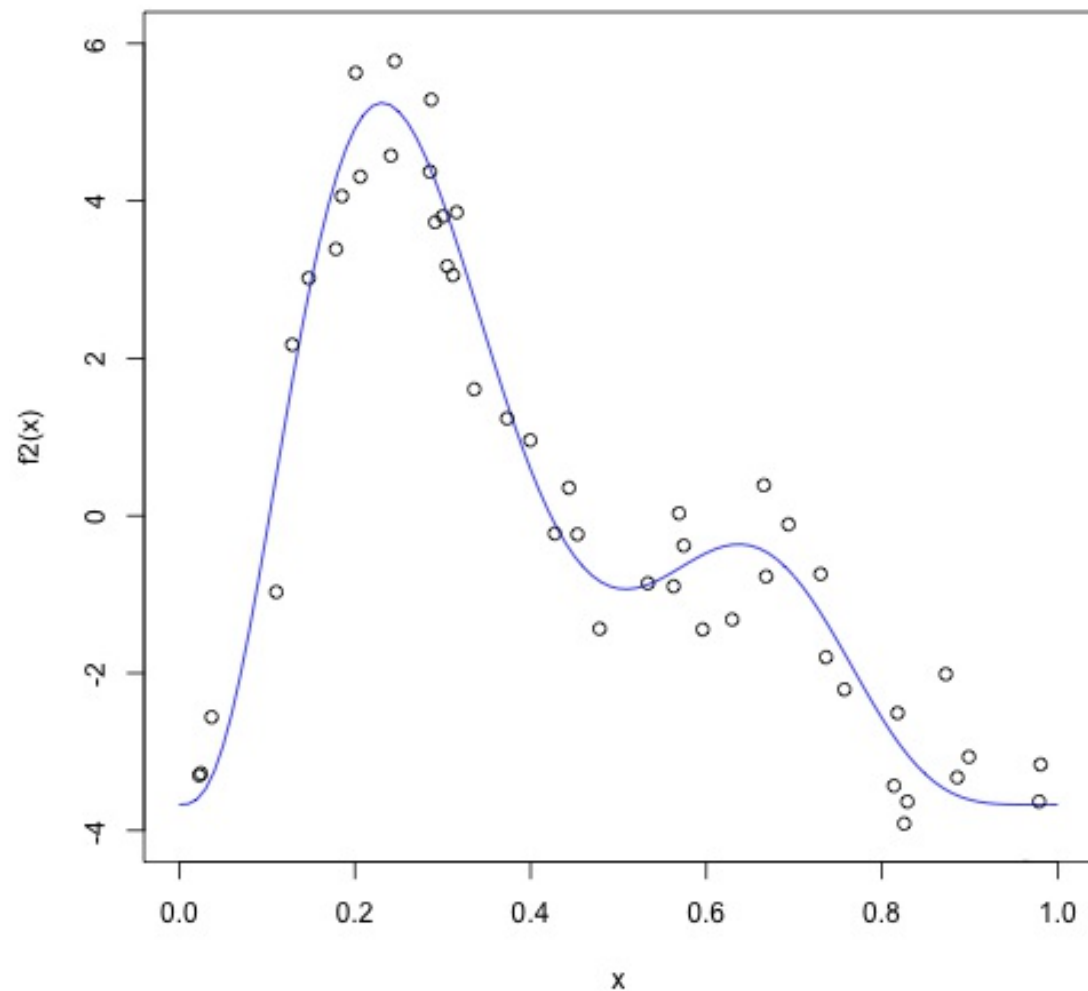
# Okay, but what about these "s" things?



- Think  $s = \text{smooth}$
- Want to model the covariates flexibly
- Covariates and response not necessarily linearly related!
- Want some wiggles

# What is smoothing?

# Straight lines vs. interpolation

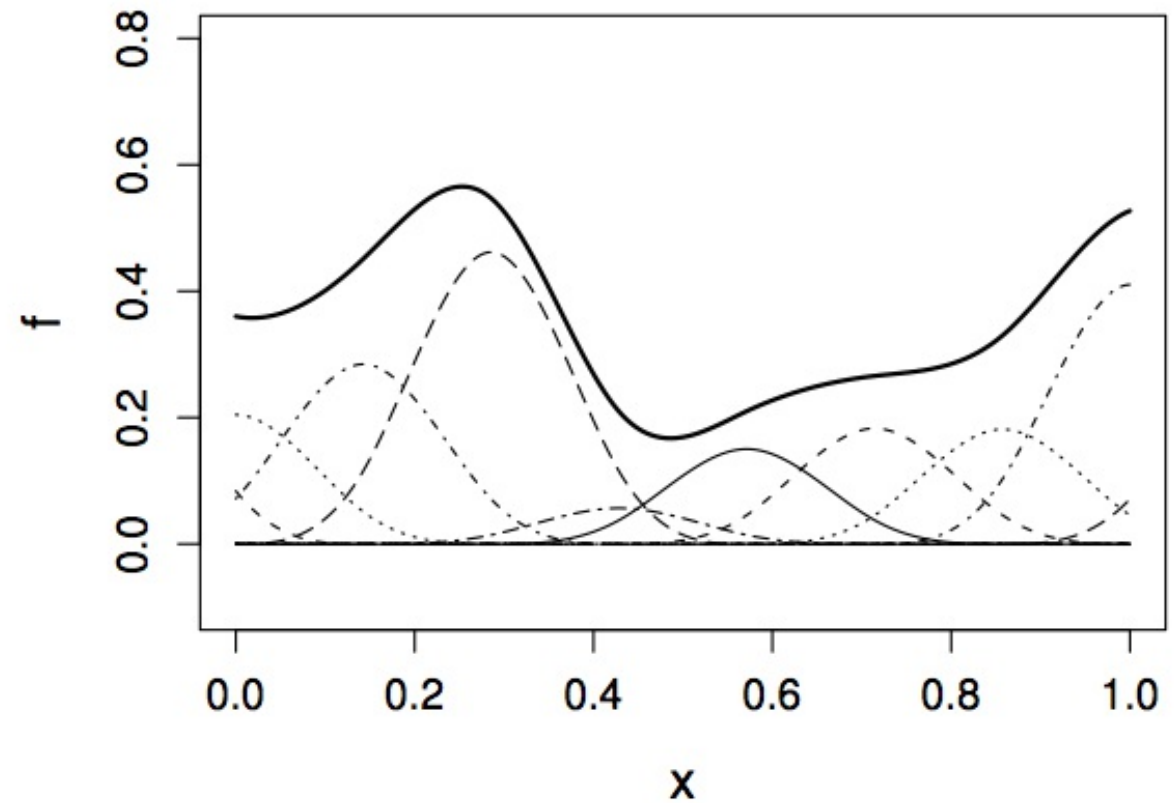
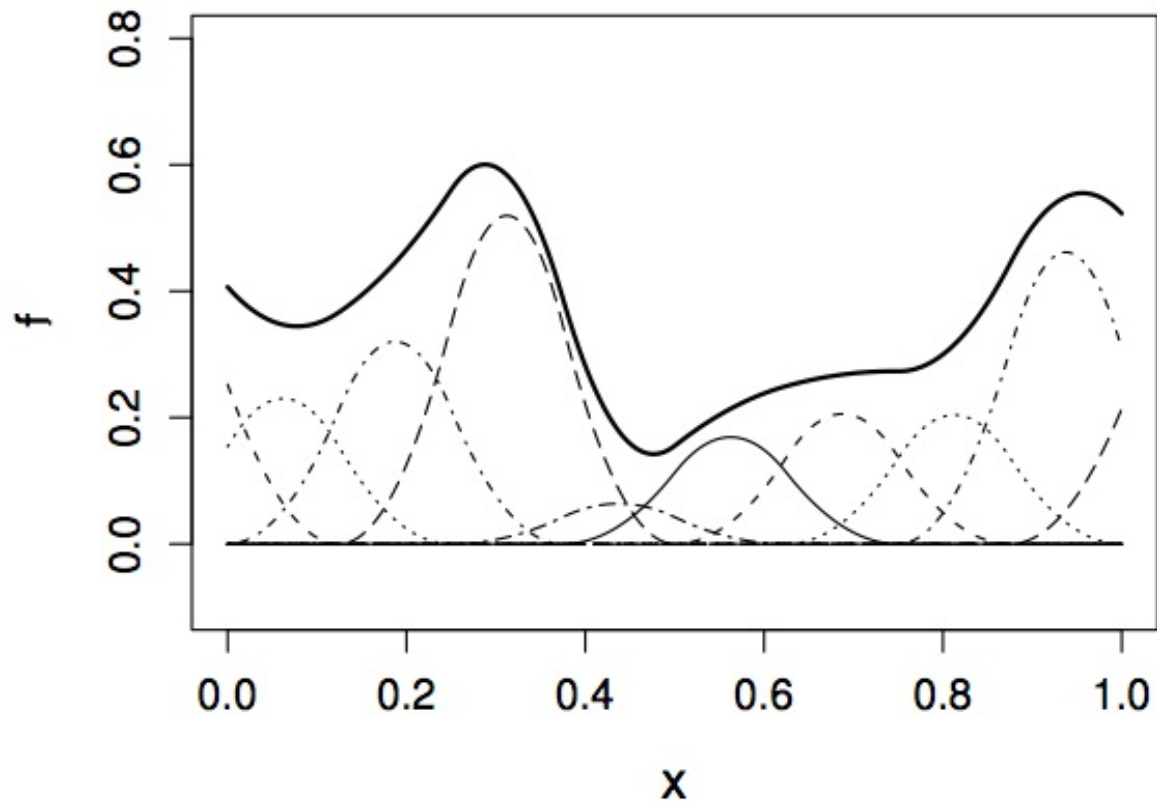


- Want a line that is “close” to all the data
- Don't want interpolation – we know there is “error”
- Balance between interpolation and “fit”



# Splines

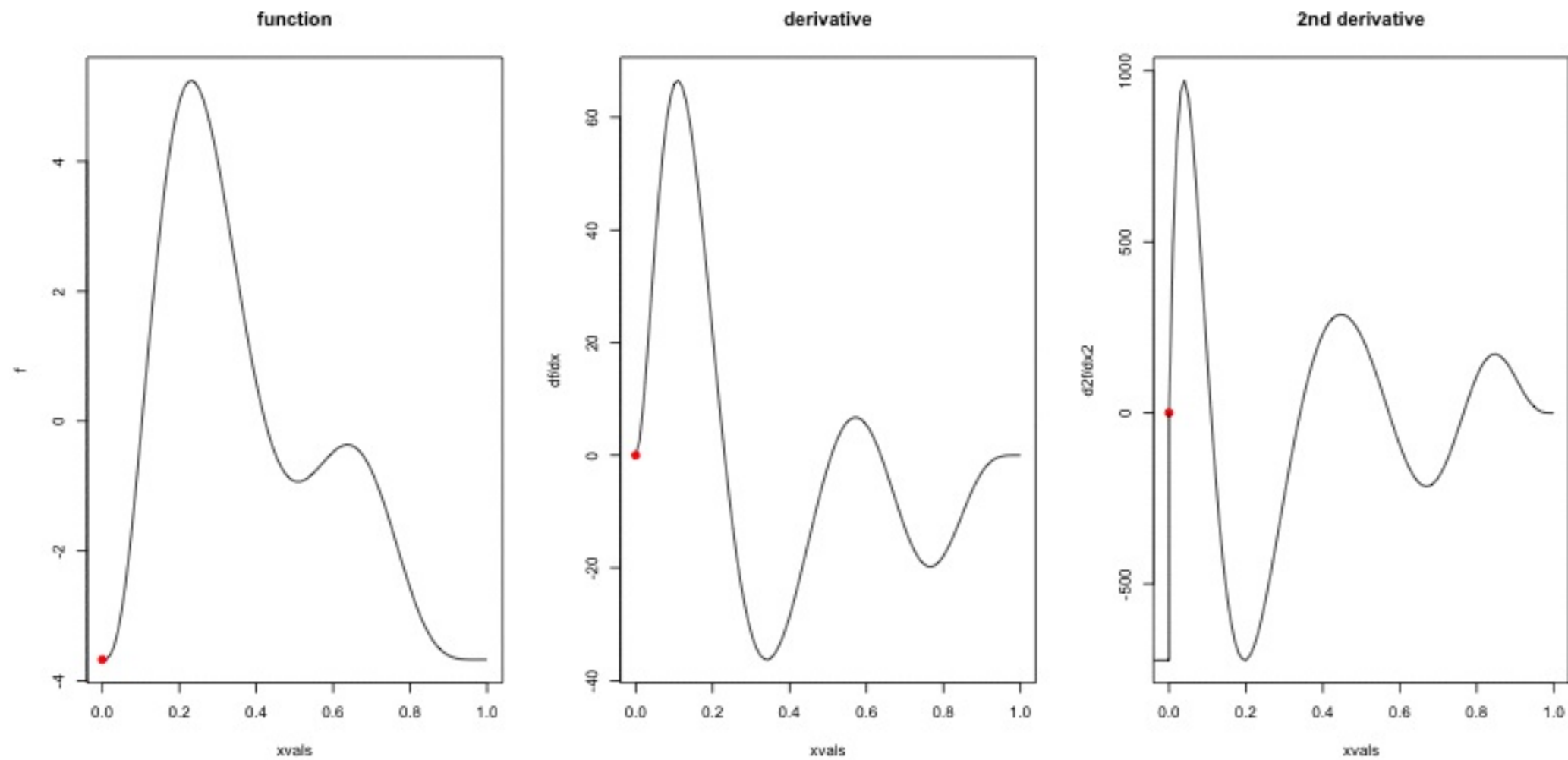
- Functions made of other, simpler functions
- **Basis functions**  $b_k$ , estimate  $\beta_k$
- $s(x) = \sum_{k=1}^K \beta_k b_k(x)$
- Makes the math(s) much easier



# Measuring wigglyness

- Visually:
  - Lots of wiggles == NOT SMOOTH
  - Straight line == VERY SMOOTH
- How do we do this mathematically?
  - Derivatives!
  - (Calculus *was* a useful class afterall)

# Wigglyness by derivatives



# Making wigglyness matter

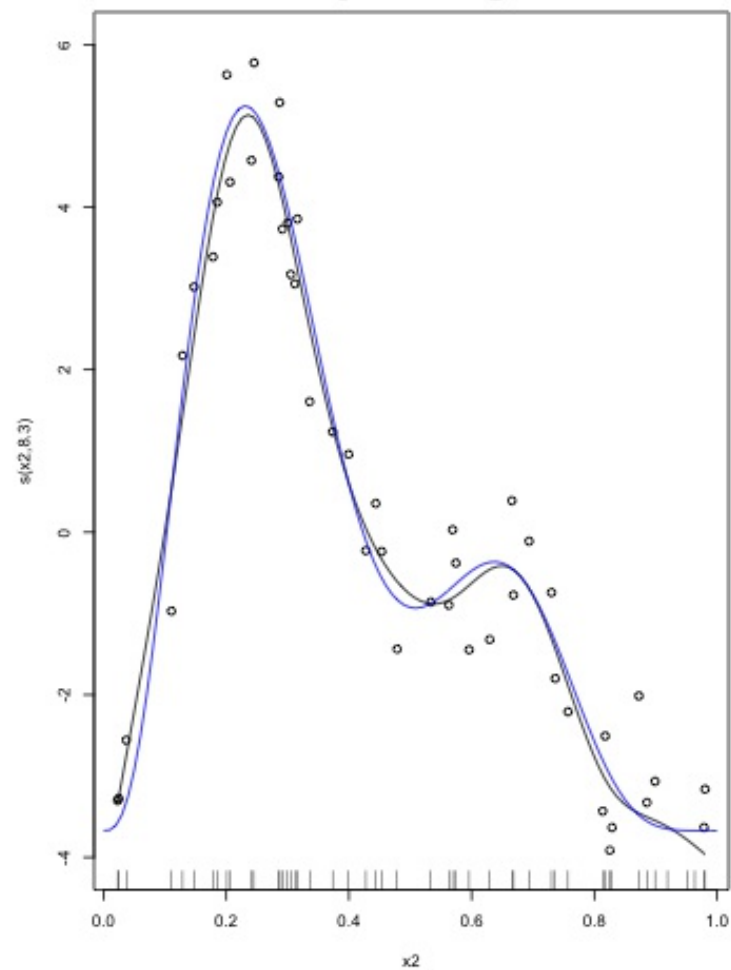
- Integration of derivative (squared) gives wigglyness
- Fit needs to be **penalised**
- **Penalty matrix** gives the wigglyness
- Estimate the  $\beta_k$  terms but penalise objective
  - “closeness to data” + penalty

# Penalty matrix

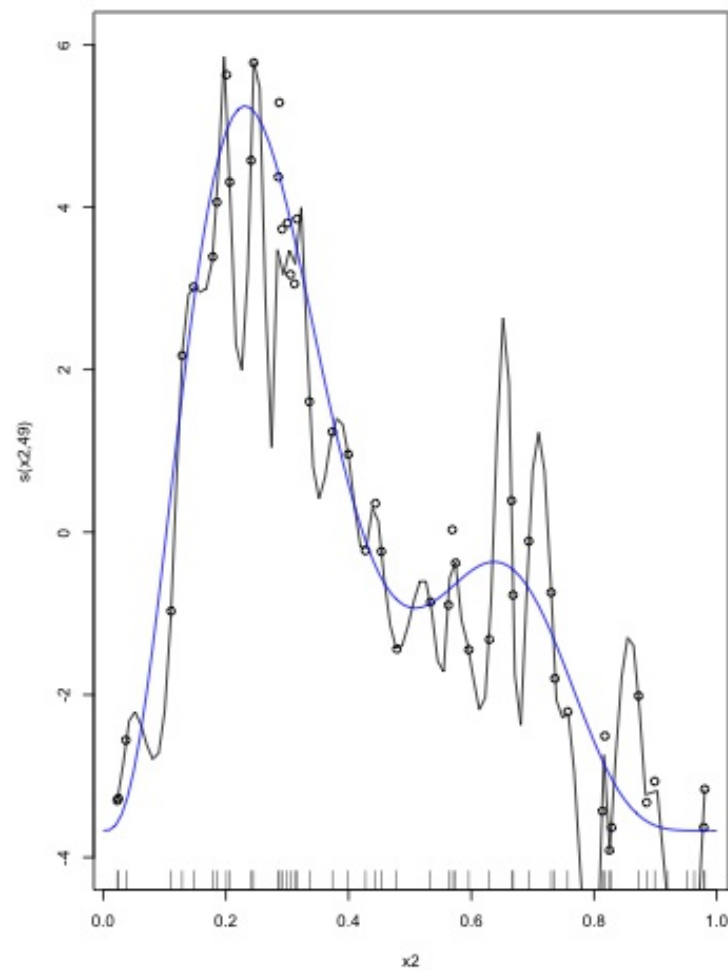
- For each  $b_k$  calculate the penalty
- Penalty is a function of  $\beta$ 
  - $\lambda \beta^T S \beta$
- $S$  calculated once
- smoothing parameter ( $\lambda$ ) dictates influence

# Smoothing parameter

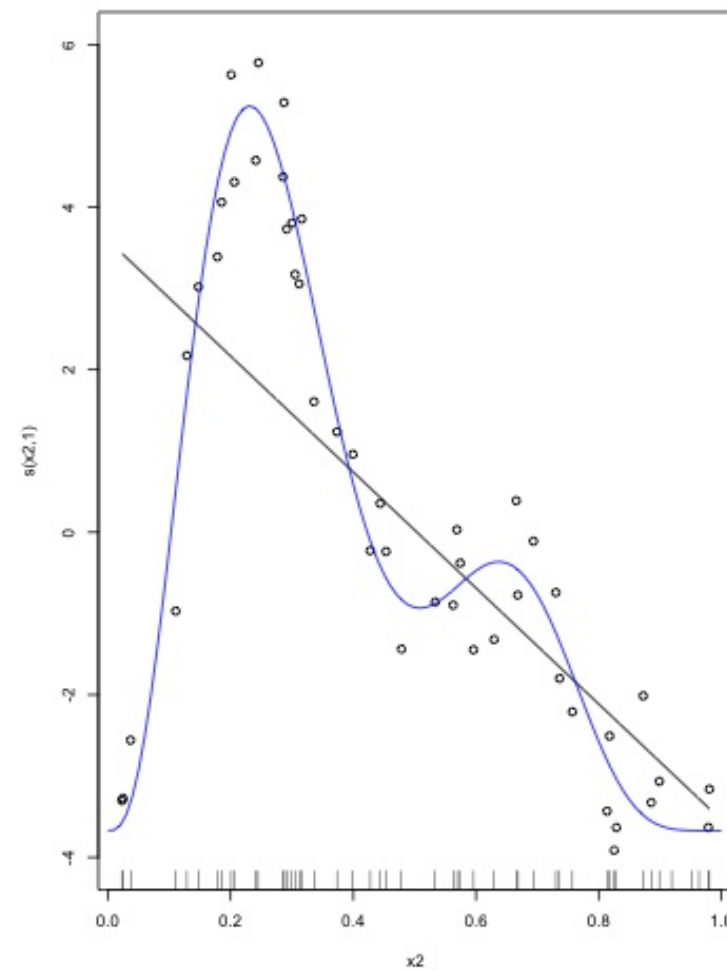
$\lambda = \text{just right}$



$\lambda = 0$



$\lambda = \infty$



# How wiggly are things?

- We can set **basis complexity** or “size” ( $k$ )
  - Maximum wigglyness
- Smoother have **effective degrees of freedom** (EDF)
- $\text{EDF} < k$
- Set  $k$  “large enough”

Okay, that was a lot of theory...



# Fitting GAMs using dsm

# Translating maths into R

$$n_j = A_j \hat{p}_j \exp[\beta_0 + s(y_j)] + \epsilon_j$$

where  $\epsilon_j \sim N(0, \sigma)$ ,  $n_j \sim \text{count distribution}$

- inside the link: `formula=count ~ s(y)`
- response distribution: `family=nb()` or `family=tw()`
- detectability: `ddf.obj=df_hr`
- offset, data: `segment.data=segs,`  
`observation.data=obs`

# Your first DSM

```
library(dsm)
dsm_x_tw <- dsm(count~s(x), ddf.obj=df_hr,
                segment.data=segs, observation.data=obs,
                family=tw(), method="REML")
```

(method="REML" uses REML to select the smoothing parameter)

dsm is based on mgcv by Simon Wood

# What did that do?

```
summary(dsm_x_tw)
```

Family: Tweedie(p=1.326)

Link function: log

Formula:

count ~ s(x) + offset(off.set)

Parametric coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	-19.7008	0.2277	-86.52	<2e-16 ***

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:

	edf	Ref.df	F	p-value
s(x)	4.962	6.047	6.403	1.07e-06 ***

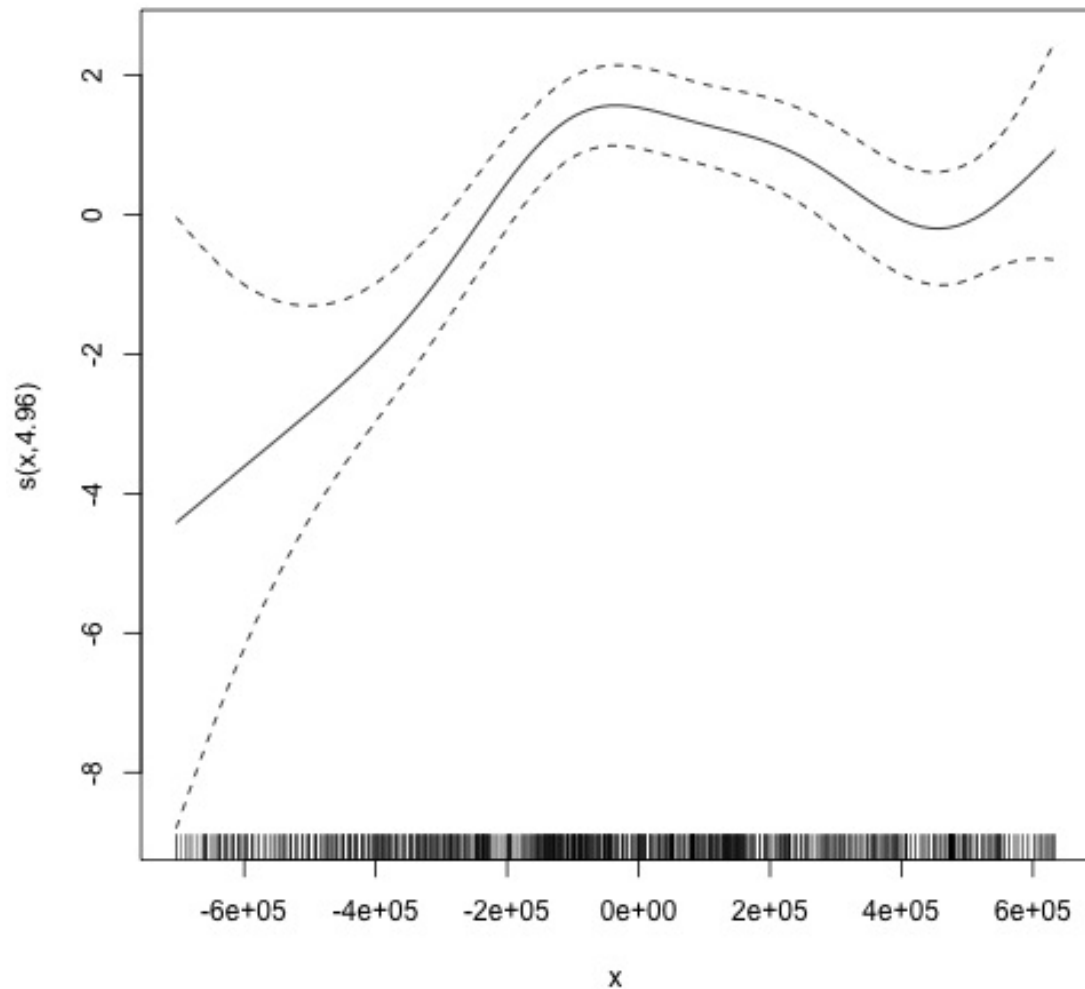
---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) = 0.0283      Deviance explained = 17.7%

-REML = 409.94      Scale est. = 6.0413      n = 949

# Plotting



- `plot(dsm_x_tw)`
- Dashed lines indicate  $\pm 2$  standard errors
- Rug plot
- On the link scale
- EDF on y axis

# Adding a term

- Just use +

```
dsm_xy_tw <- dsm(count ~ s(x) + s(y),  
                 ddf.obj=df_hr,  
                 segment.data=segs, observation.data=obs,  
                 family=tw(), method="REML")
```

# Summary

```
summary(dsm_xy_tw)
```

Family: Tweedie(p=1.306)

Link function: log

Formula:

count ~ s(x) + s(y) + offset(off.set)

Parametric coefficients:

	Estimate	Std. Error	t value	Pr(> t )	
(Intercept)	-19.9801	0.2381	-83.93	<2e-16	***

---

Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:

	edf	Ref.df	F	p-value	
s(x)	4.943	6.057	3.224	0.004239	**
s(y)	5.293	6.419	4.034	0.000322	***

---

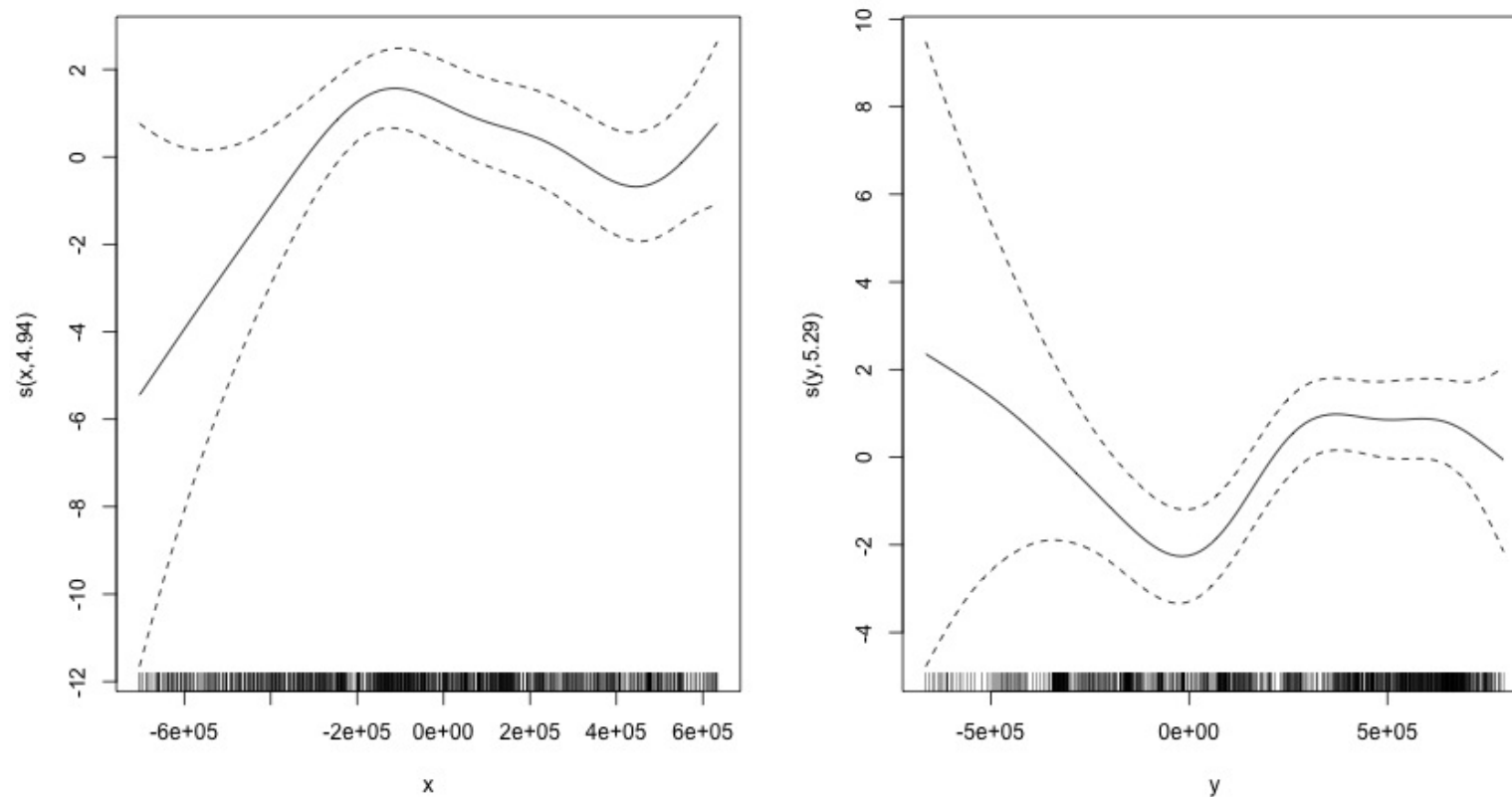
Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) = 0.0678      Deviance explained = 27.3%

-REML = 399.84      Scale est. = 5.3157      n = 949

# Plotting

```
plot(dsm_xy_tw, scale=0, pages=1)
```



- `scale=0`: each plot on different scale
- `pages=1`: plot together



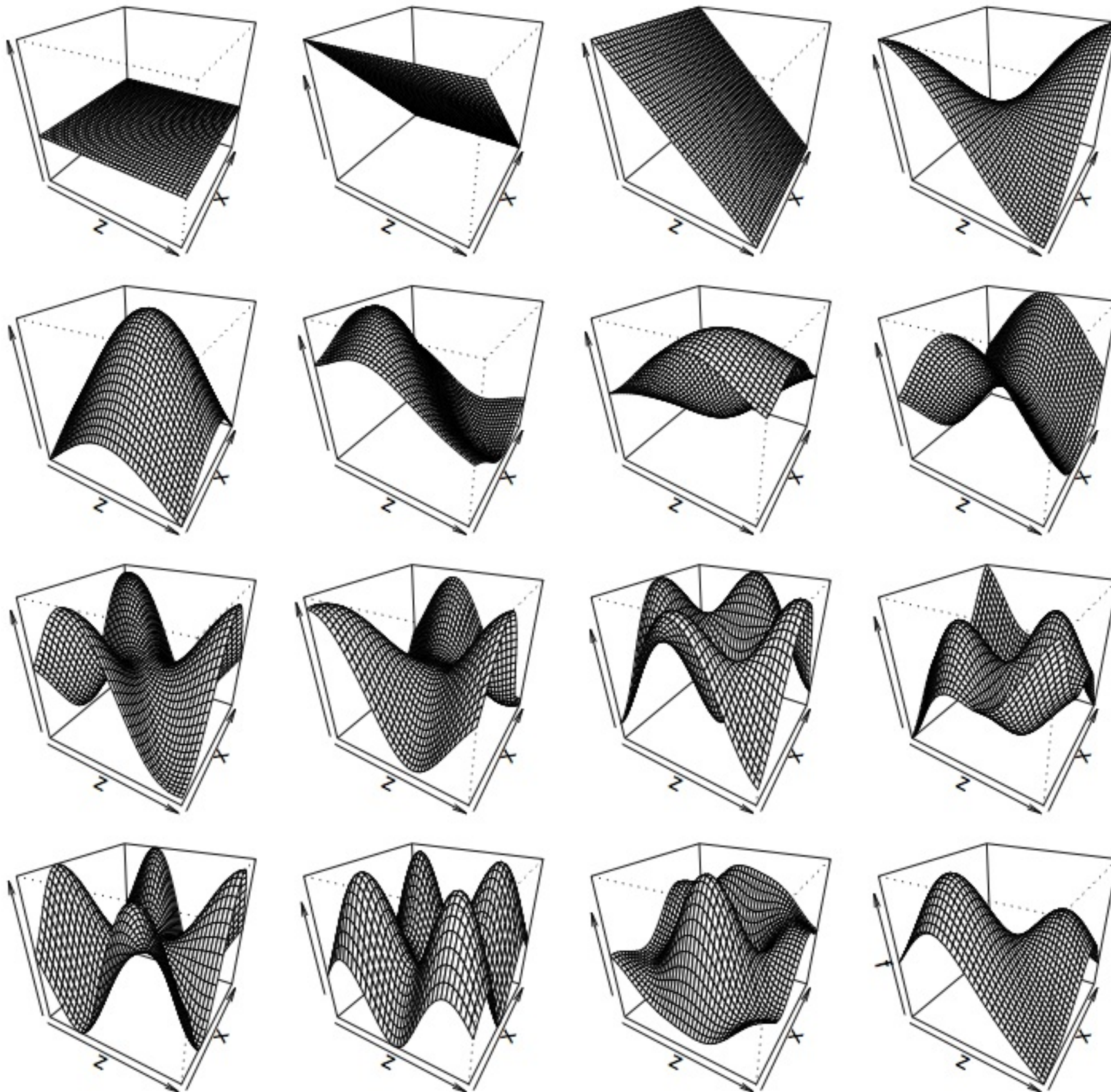
# Bivariate terms

- Assumed an additive structure
- No interaction
- We can specify  $s(x, y)$  (and  $s(x, y, z, \dots)$ )

# Thin plate regression splines

- Default basis
- One basis function per data point
- Reduce # basis functions (eigendecomposition)
- Fitting on reduced problem
- Multidimensional

# Thin plate splines (2-D)



# Bivariate spatial term

```
dsm_xyb_tw <- dsm(count ~ s(x, y),  
                  ddf.obj=df_hr,  
                  segment.data=segs, observation.data=obs,  
                  family=tw(), method="REML")
```

# Summary

```
summary(dsm_xyb_tw)
```

```
Family: Tweedie(p=1.29)
Link function: log
```

```
Formula:
count ~ s(x, y) + offset(off.set)
```

```
Parametric coefficients:
```

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	-20.1638	0.2477	-81.4	<2e-16 ***

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

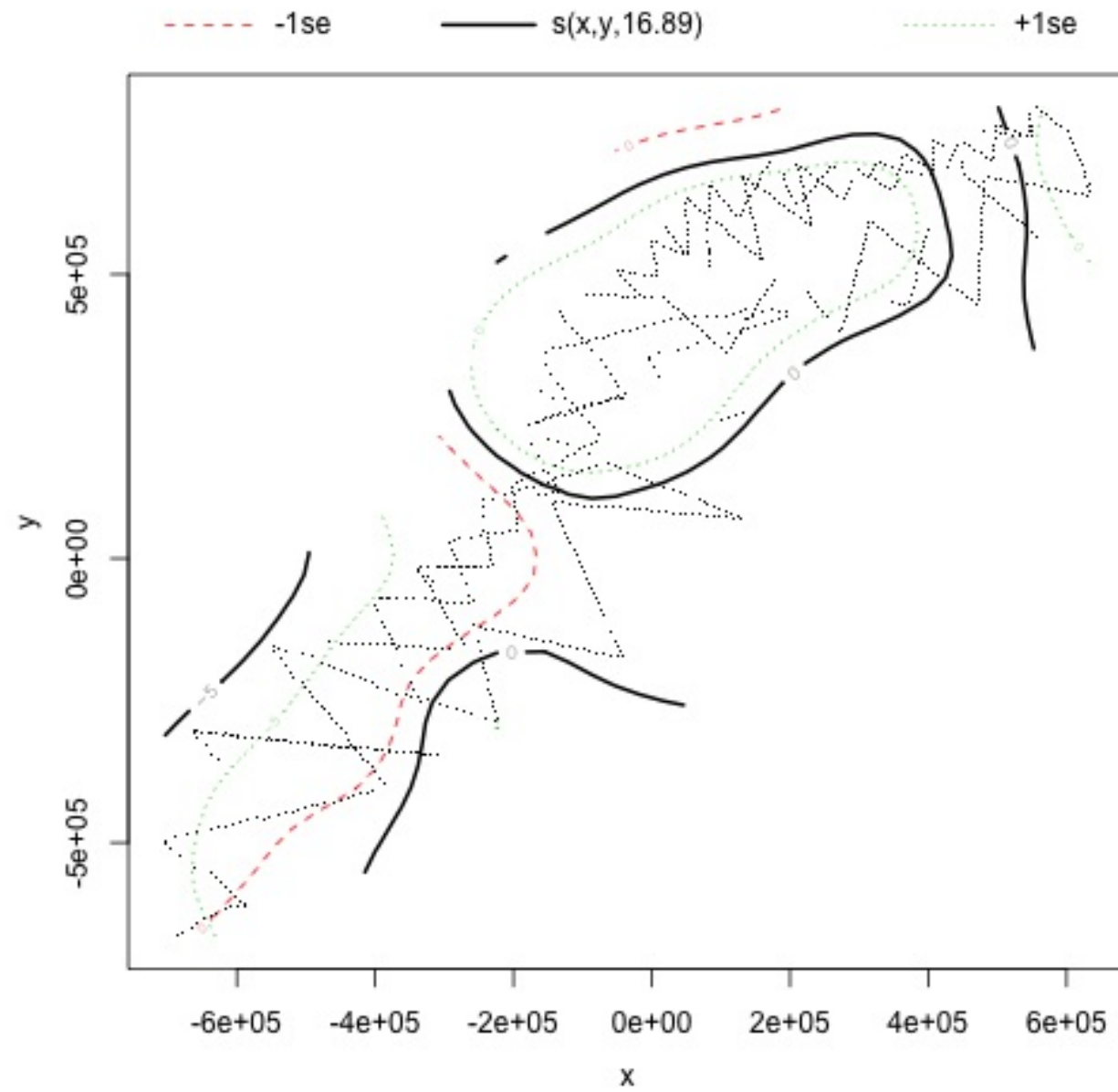
```
Approximate significance of smooth terms:
```

	edf	Ref.df	F	p-value
s(x,y)	16.89	21.12	4.333	3.73e-10 ***

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

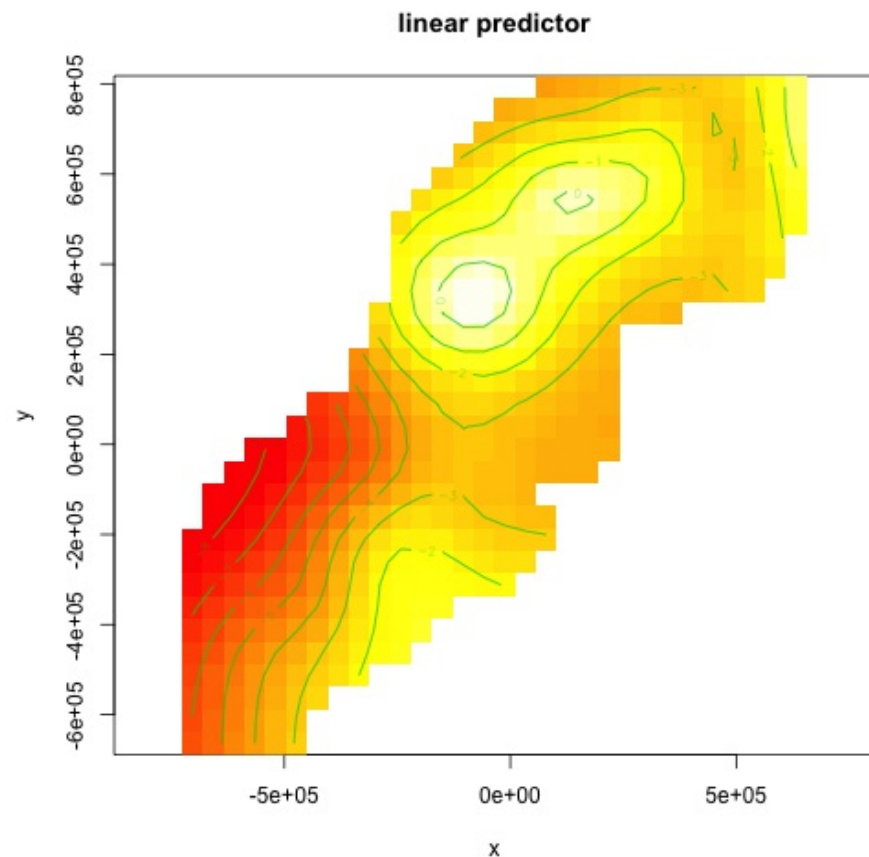
```
R-sq.(adj) = 0.102    Deviance explained = 34.5%
-REML = 394.86    Scale est. = 4.8248    n = 949
```

# Plotting... erm...



# Let's try something different

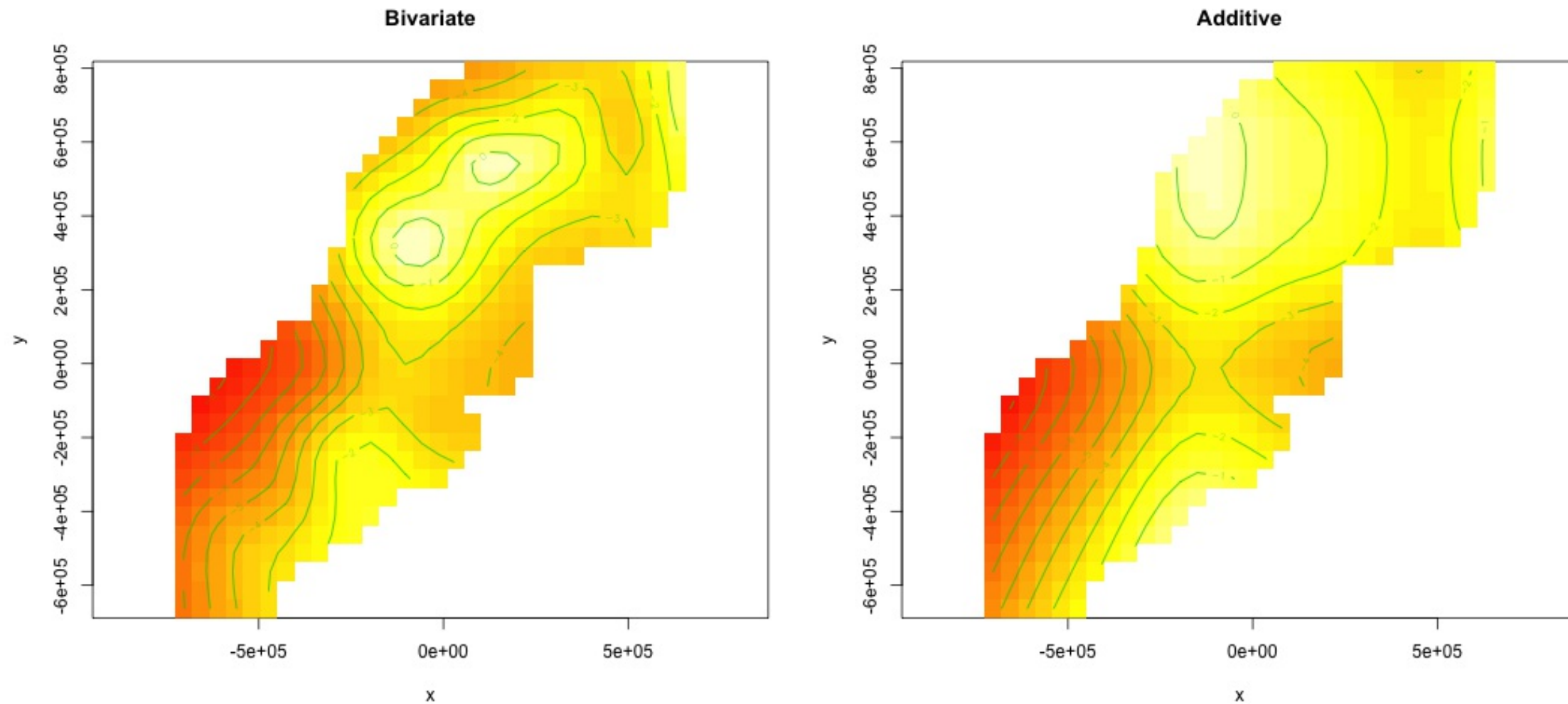
```
vis.gam(dsm_xyb_tw, view=c("x", "y"), plot.type="contour",  
too.far=0.1, asp=1)
```



- Still on link scale
- `too.far` excludes points far from data



# Comparing bivariate and additive models





# Model checking

*“perhaps the most important part of applied statistical modelling”*

Simon Wood

# Model checking

- As with detection function, checking is important
- Want to know the model conforms to assumptions
- What assumptions should we check?

# What to check

- Convergence (not usually an issue)
- Basis size is big enough
- Residuals

# Basis size

# Basis size (k)

- Set k per term
- e.g.  $s(x, k=10)$  or  $s(x, y, k=100)$
- Penalty removes “extra” wigglyness
  - *up to a point!*
- (But computation is slower with bigger k)

# Checking basis size

```
gam.check(dsm_x_tw)
```

```
Method: REML    Optimizer: outer newton  
full convergence after 7 iterations.  
Gradient range [-3.08755e-06,4.928062e-07]  
(score 409.936 & scale 6.041307).  
Hessian positive definite, eigenvalue range [0.7645492,302.127].  
Model rank = 10 / 10
```

Basis dimension (k) checking results. Low p-value (k-index<1) may indicate that k is too low, especially if edf is close to k'.

	k'	edf	k-index	p-value
s(x)	9.000	4.962	0.763	0.44

# Increasing basis size

```
dsm_x_tw_k <- dsm(count~s(x, k=20), ddf.obj=df_hr,  
                  segment.data=segs, observation.data=obs,  
                  family=tw(), method="REML")  
gam.check(dsm_x_tw_k)
```

Method: REML    Optimizer: outer newton  
full convergence after 7 iterations.  
Gradient range [-2.301246e-08,3.930757e-09]  
(score 409.9245 & scale 6.033913).  
Hessian positive definite, eigenvalue range [0.7678456,302.0336].  
Model rank = 20 / 20

Basis dimension (k) checking results. Low p-value (k-index<1) may indicate that k is too low, especially if edf is close to k'.

	k'	edf	k-index	p-value
s(x)	19.000	5.246	0.763	0.36

# Sometimes basis size isn't the issue...

- Generally, double  $k$  and see what happens
- Didn't increase the EDF much here
- Other things can cause low “p-value” and “k-index”
- Increasing  $k$  can cause problems (nullspace)

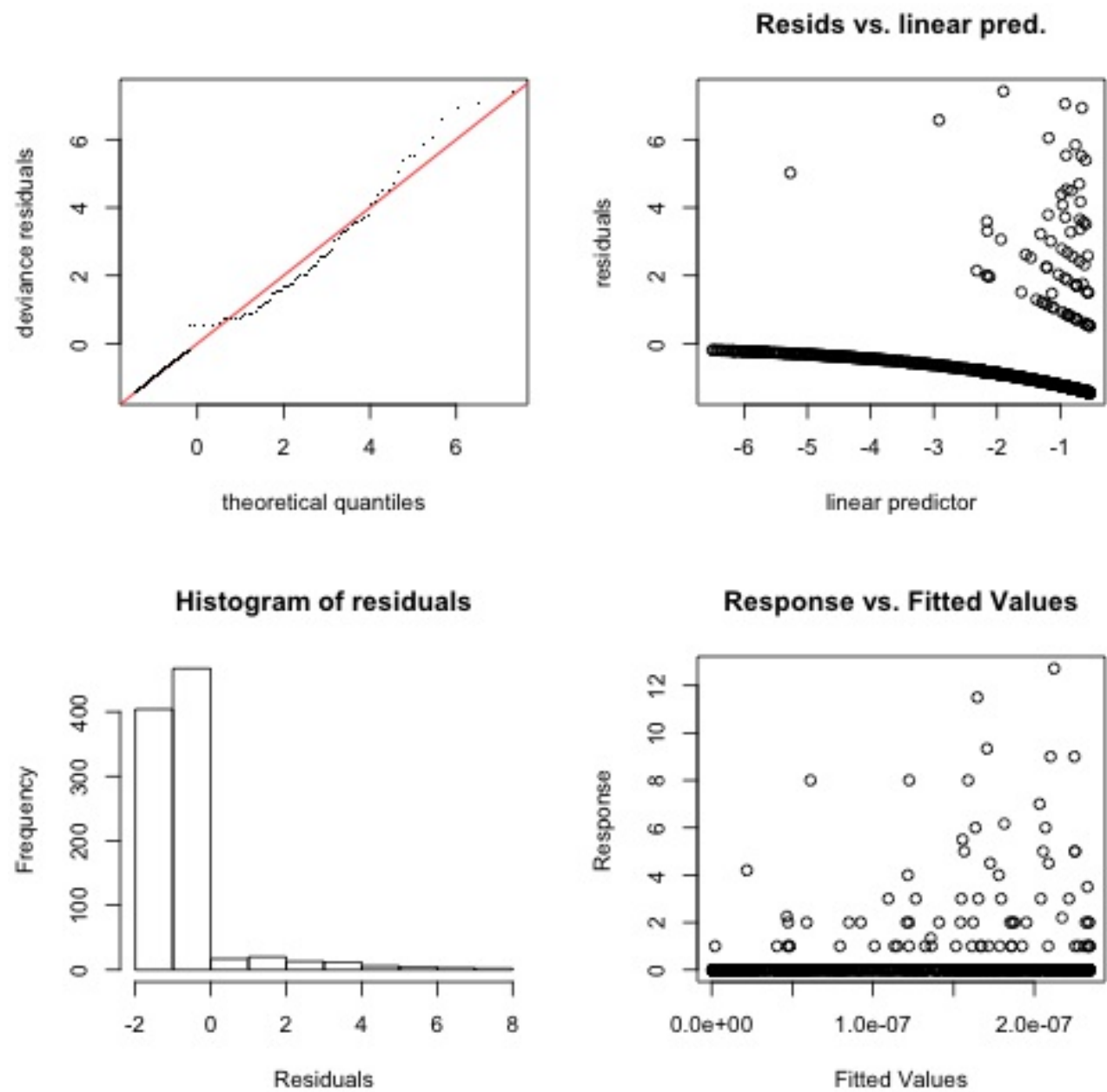


Don't throw away your  
residuals!

# What are residuals?

- Generally residuals = observed value - fitted value
- BUT hard to see patterns in these “raw” residuals
- Need to standardise – **deviance residuals**
- Residual sum of squares  $\Rightarrow$  linear model
  - deviance  $\Rightarrow$  GAM
- Expect these residuals  $\sim N(0, 1)$

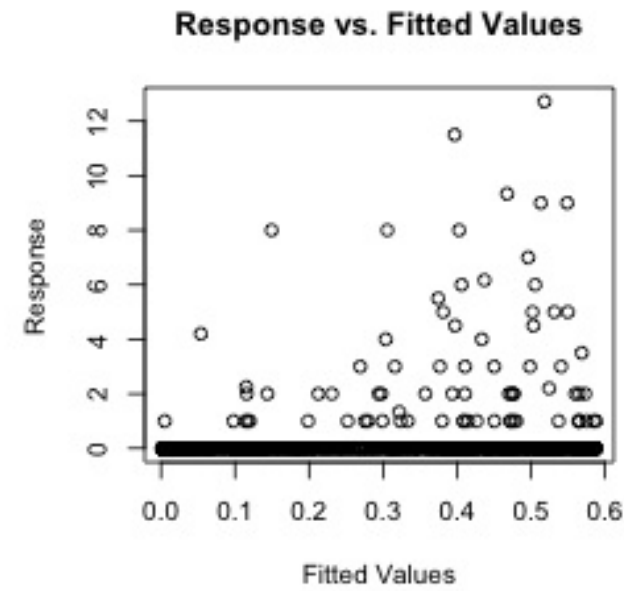
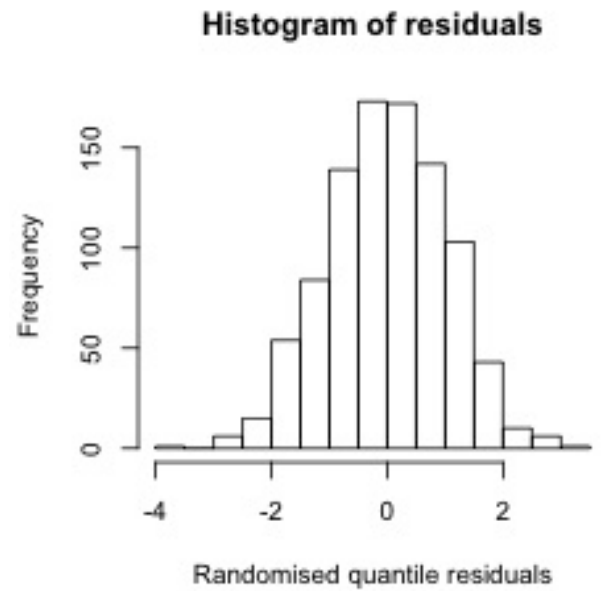
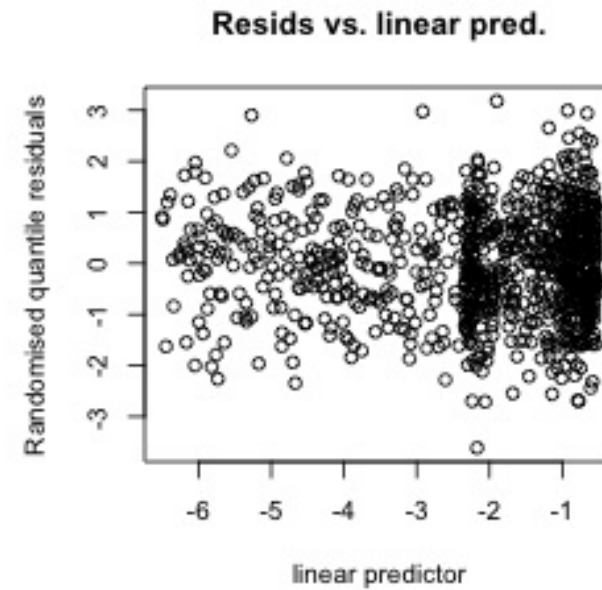
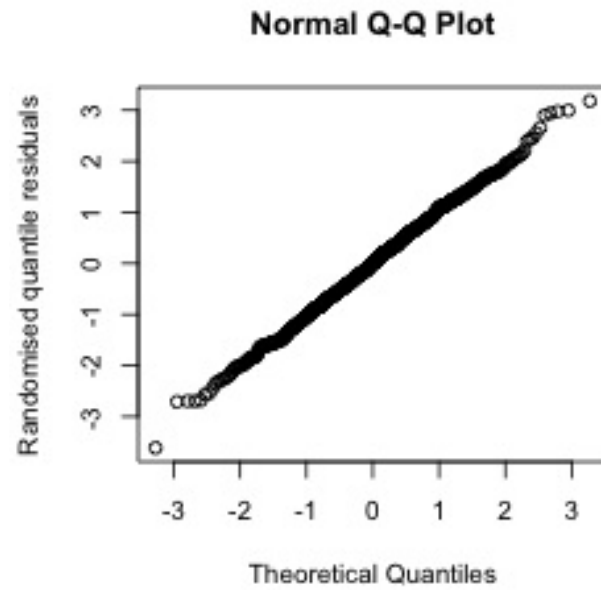
# Residual checking



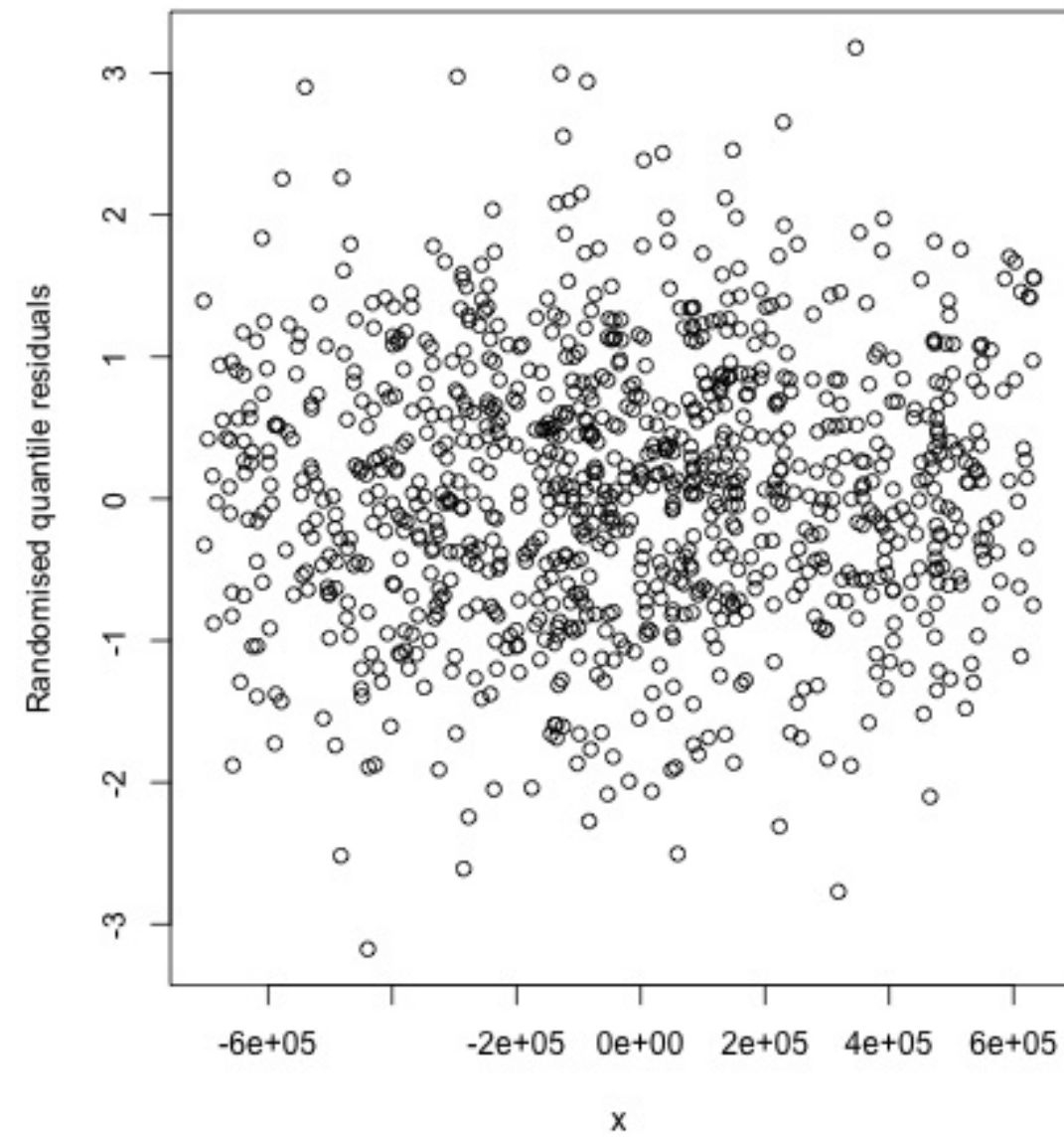
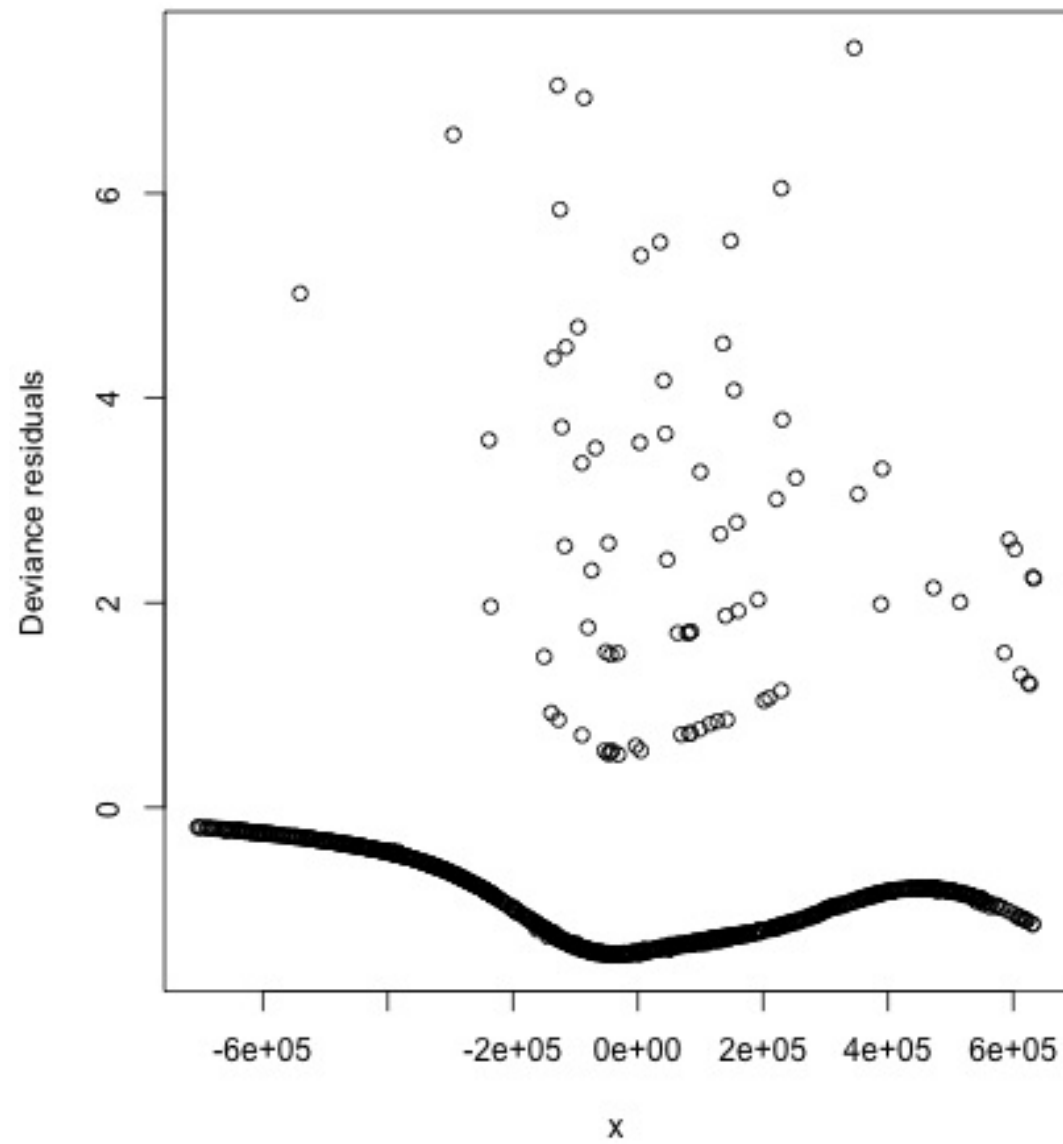
# Shortcomings

- `gam.check` left side can be helpful
- Right side is victim of artifacts
- Need an alternative
- “Randomised quantile residuals” (*experimental*)
  - `rqqam.check`
  - Exactly normal residuals (left side useless)

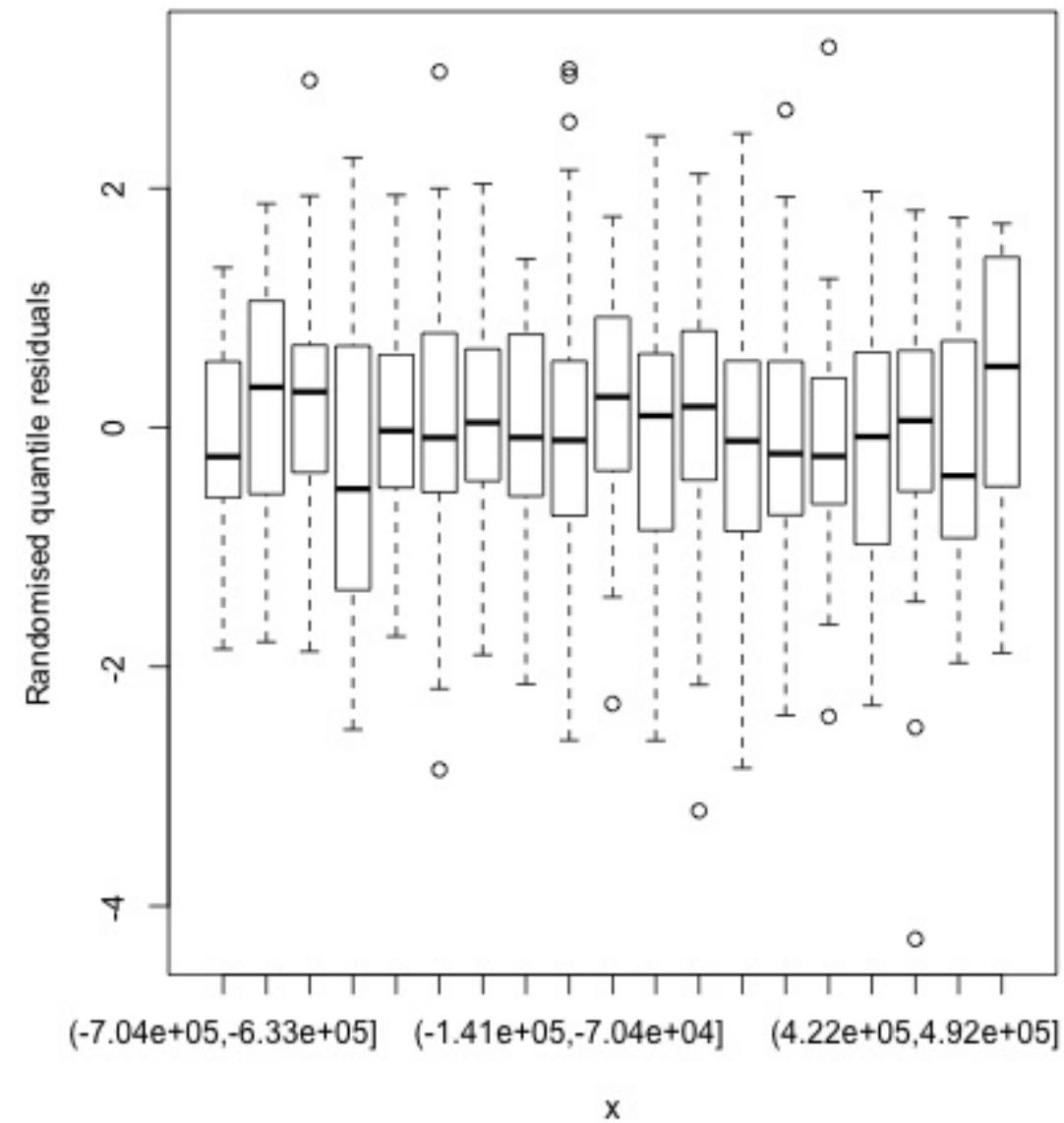
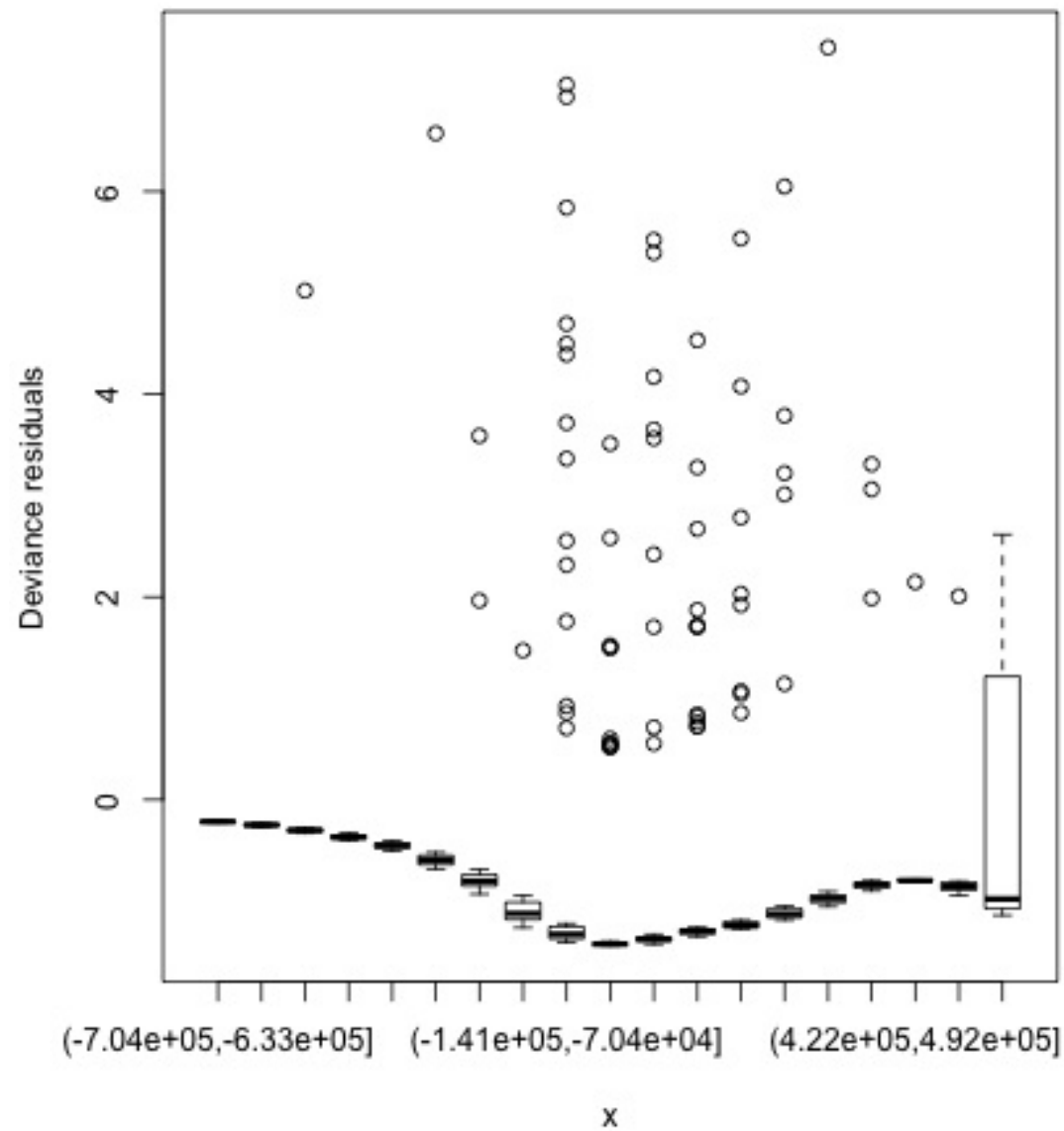
# Randomised quantile residuals



# Residuals vs. covariates

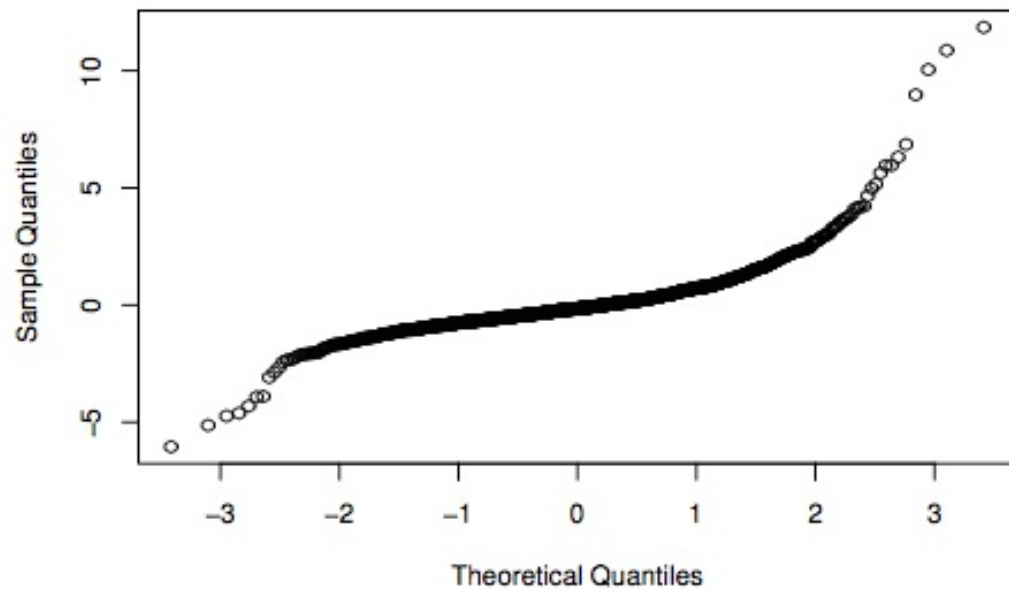


# Residuals vs. covariates (boxplots)

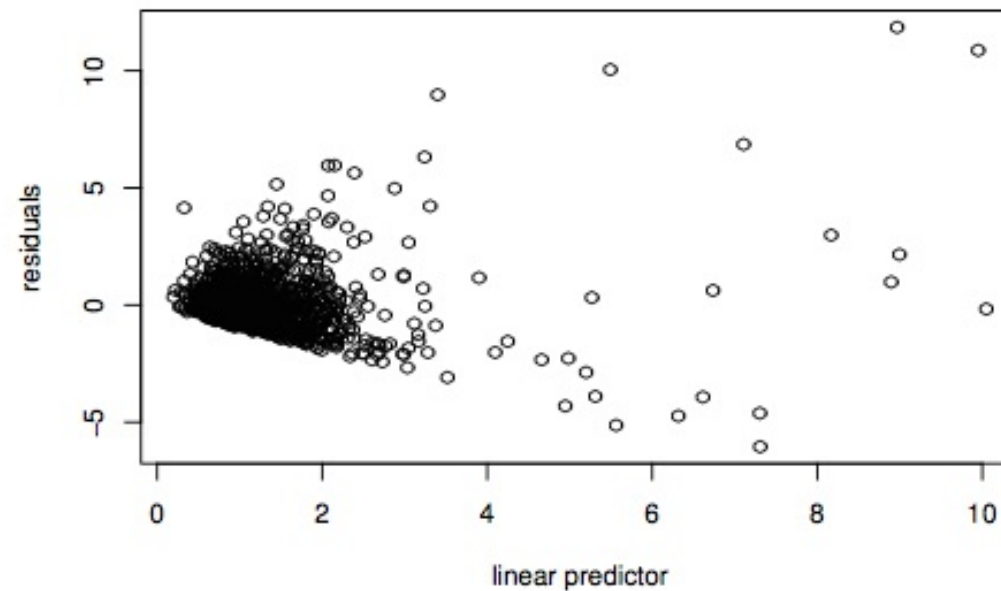


# Example of "bad" plots

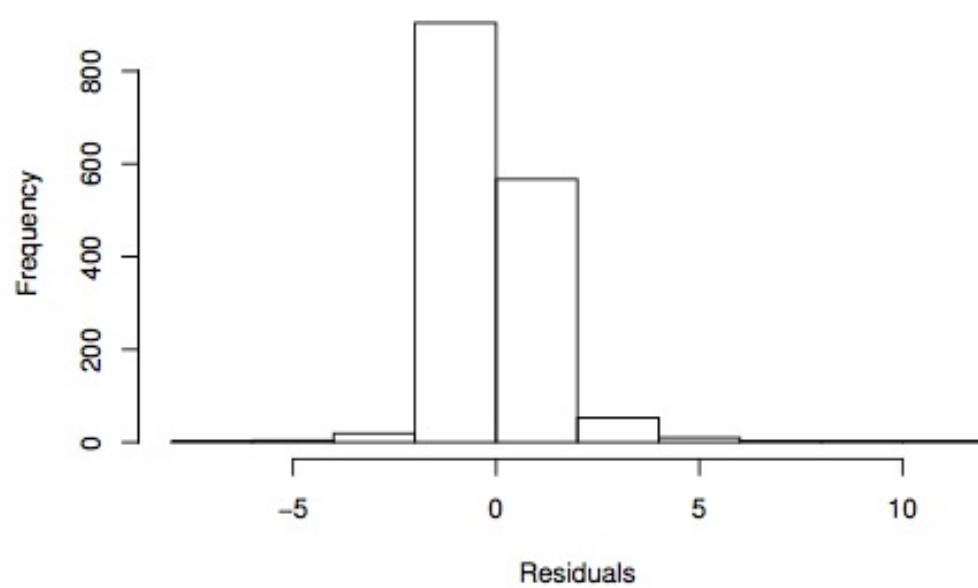
Normal Q-Q Plot



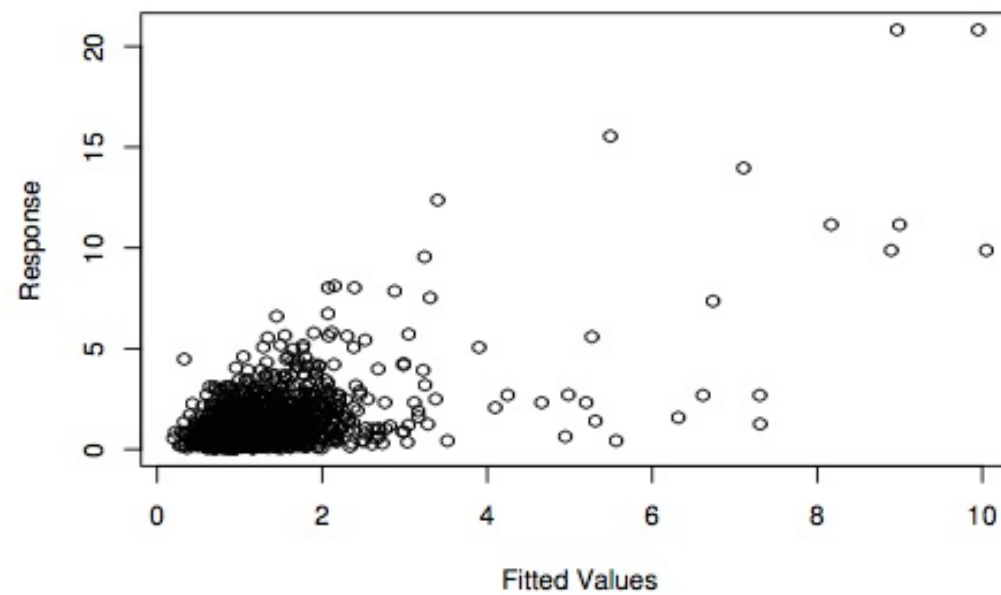
Resids vs. linear pred.



Histogram of residuals



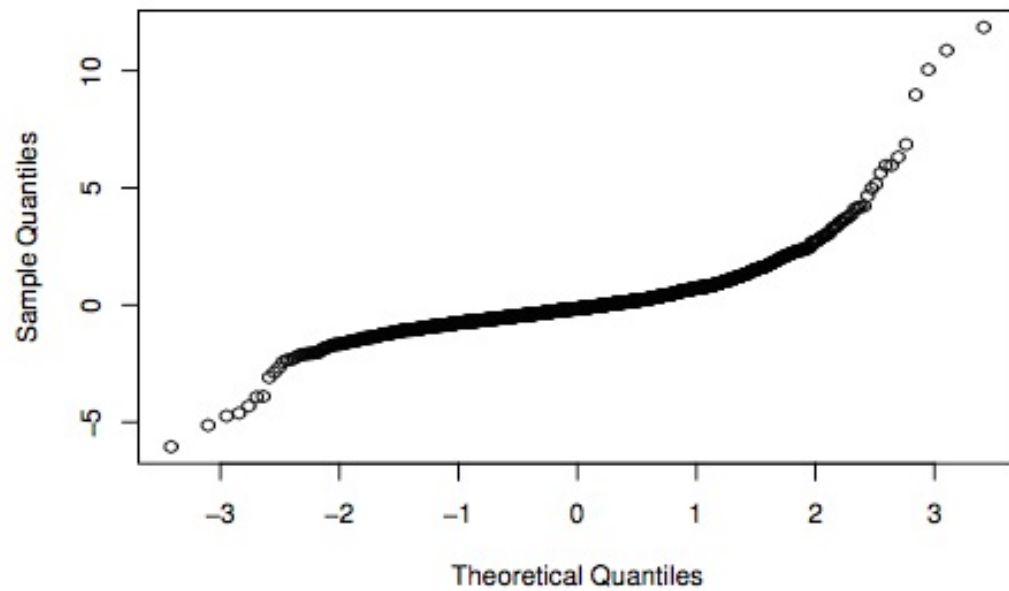
Response vs. Fitted Values



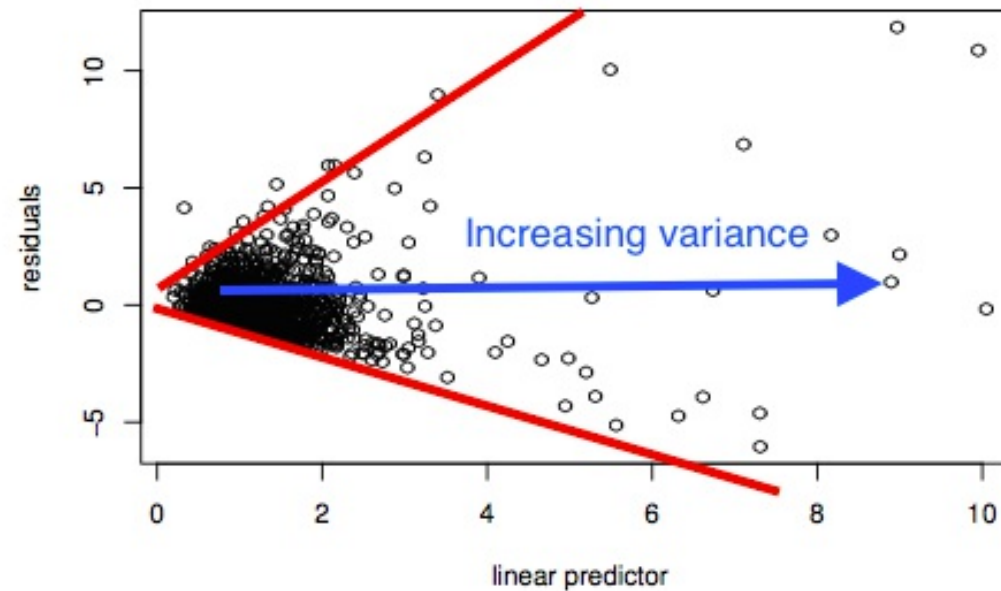


# Example of "bad" plots

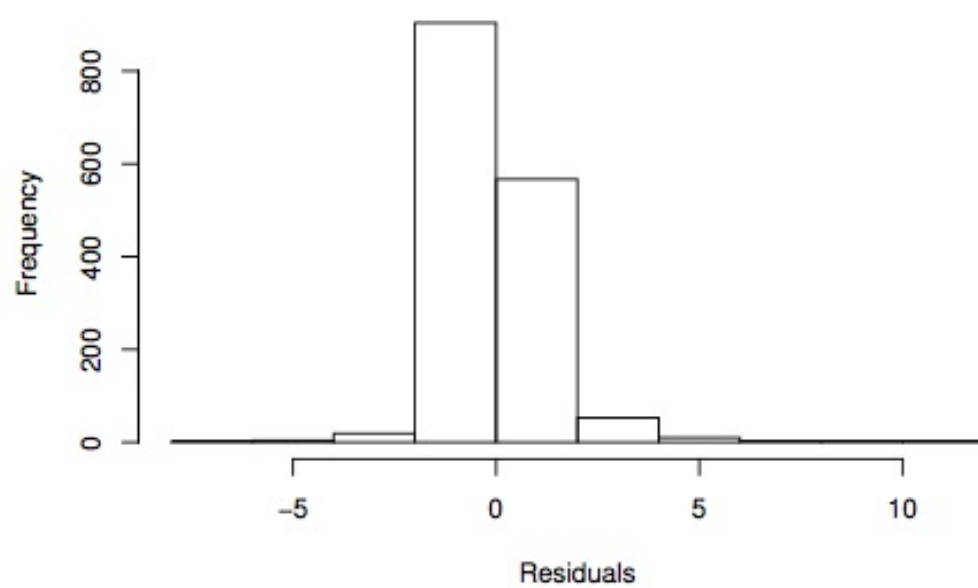
Normal Q-Q Plot



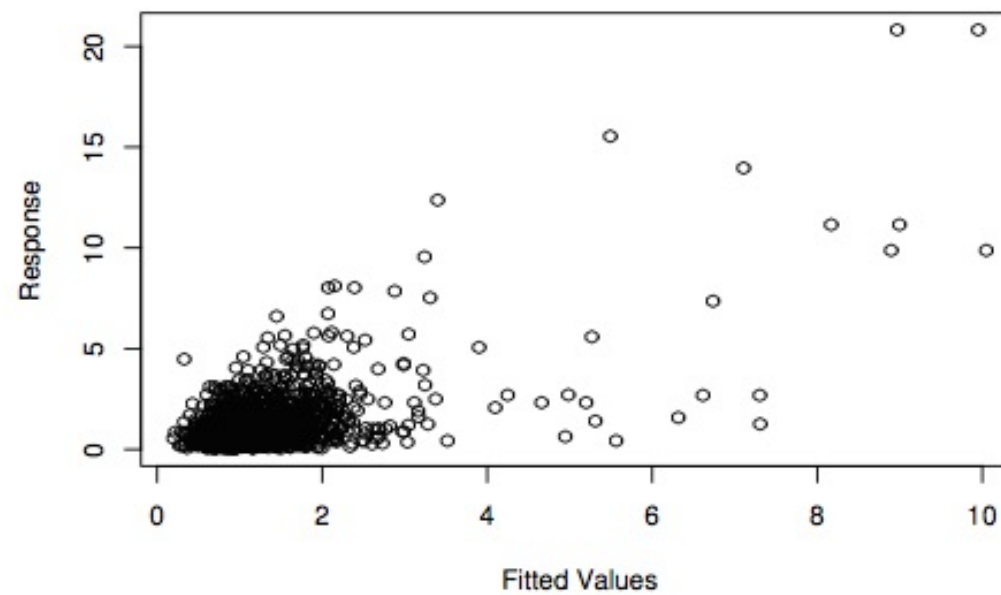
Resids vs. linear pred.



Histogram of residuals



Response vs. Fitted Values



# Residual checks

- Looking for patterns (not artifacts)
- This can be tricky
- Need to use a mixture of techniques

Let's have a go...