Model checking

Eric J Pedersen
August 6th, 2016

Outline

So you have a GAM:

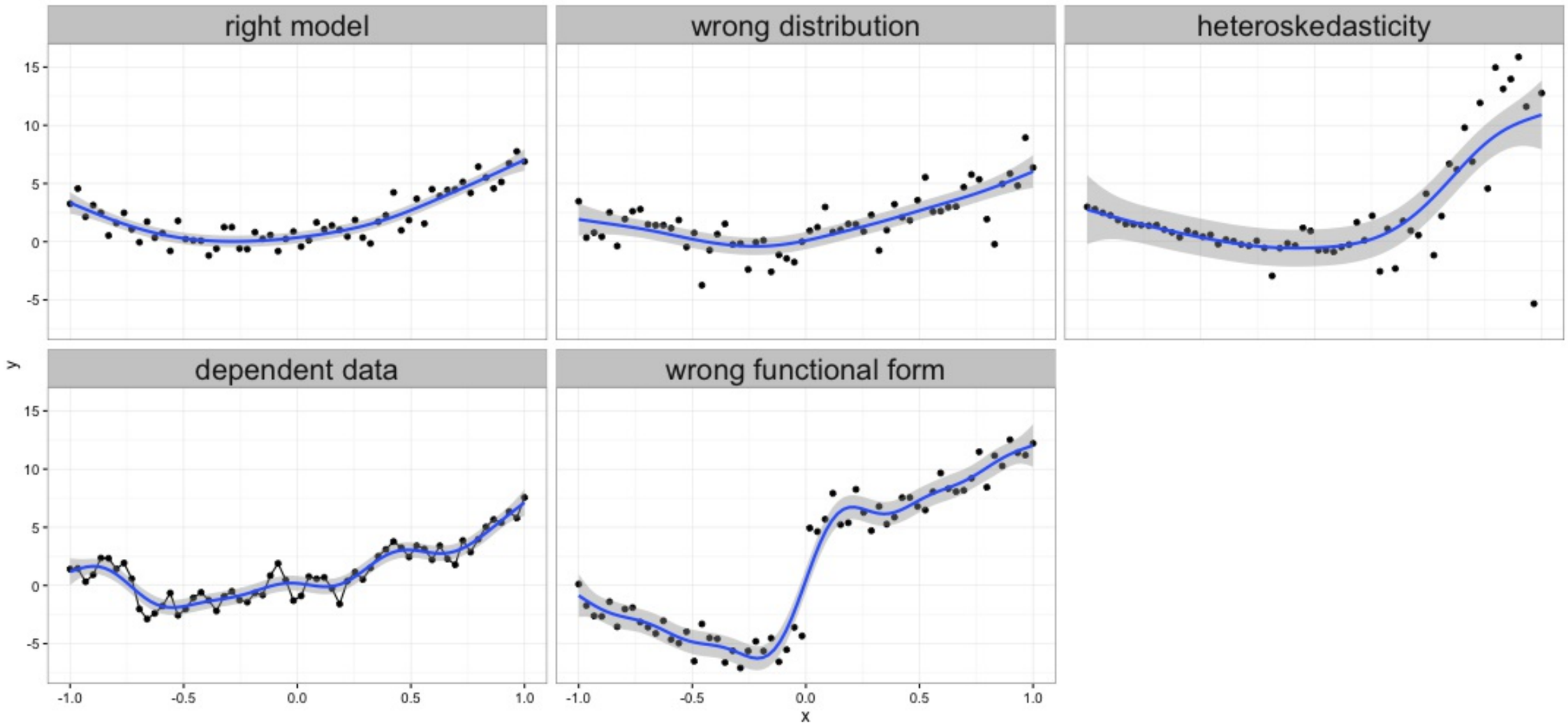
- How do you know you have the right degrees of freedom? `gam.check()`
- Diagnosing model issues: `gam.check()` part 2
- Do you have the right error model? Randomized quantile residuals
- When covariates aren't independent: estimating concurvity

Packages you'll need to follow along:

```
library(mgcv)
library(magrittr)
library(ggplot2)
library(dplyr)
library(statmod)
source("quantile_resid.R")
```

GAMs are models too

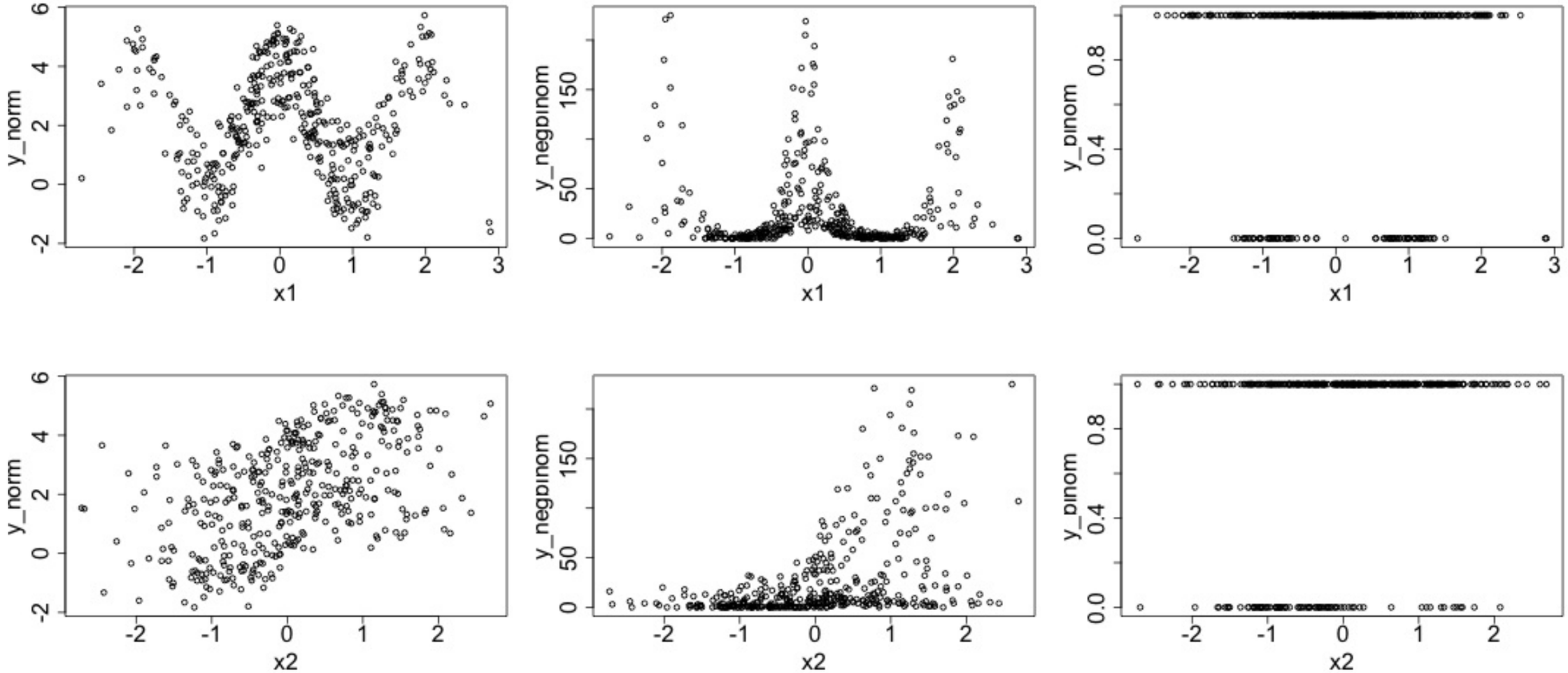
With all models, how accurate your predictions will be depends on how good the model is



So how do we test how well our model fits?

Examples:

```
set.seed(2)
n = 400
x1 = rnorm(n)
x2 = rnorm(n)
y_val =1 + 2*cos(pi*x1) + 2/(1+exp(-5*(x2)))
y_norm = y_val + rnorm(n, 0, 0.5)
y_negbinom = rnbinom(n, mu = exp(y_val),size=10)
y_binom = rbinom(n,1,prob = exp(y_val)/(1+exp(y_val)))
```



gam.check() part 1: do you have the right functional form?

How well does the model fit?

- Many choices: k, family, type of smoother, …
- How do we assess how well our model fits?

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

```
norm_model_1 = gam(y_norm~s(x1,k=4)+s(x2,k=4),method= "REML")
gam.check(norm_model_1)
```

```
Method: REML   Optimizer: outer newton
full convergence after 8 iterations.
Gradient range [-0.0003467788,0.0005154578]
(score 736.9402 & scale 2.252304).
Hessian positive definite, eigenvalue range
[0.000346021,198.5041].
Model rank =  7 / 7

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(x1) 3.000 1.002   0.125    0.00
s(x2) 3.000 2.914   1.045    0.79
```

Checking basis size

```
norm_model_2 = gam(y_norm~s(x1,k=12)+s(x2,k=4),method= "REML")
gam.check(norm_model_2)
```

```
Method: REML   Optimizer: outer newton
full convergence after 11 iterations.
Gradient range [-8.009729e-05,7.329675e-05]
(score 345.3111 & scale 0.2706205).
Hessian positive definite, eigenvalue range [0.9677905,198.63].
Model rank =  15 / 15

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(x1) 11.000 10.837   0.989    0.42
s(x2)  3.000  2.984   0.861    0.00
```

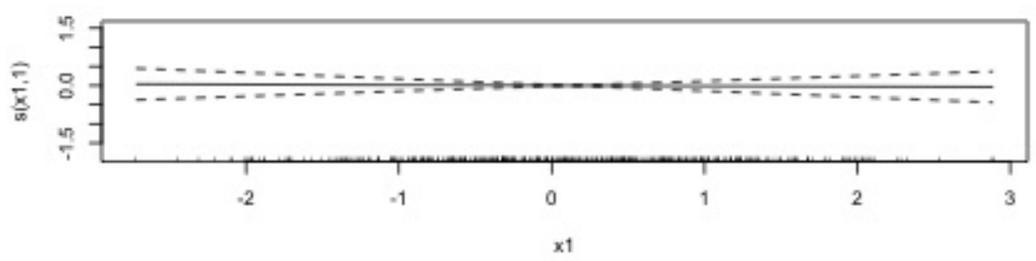
Checking basis size

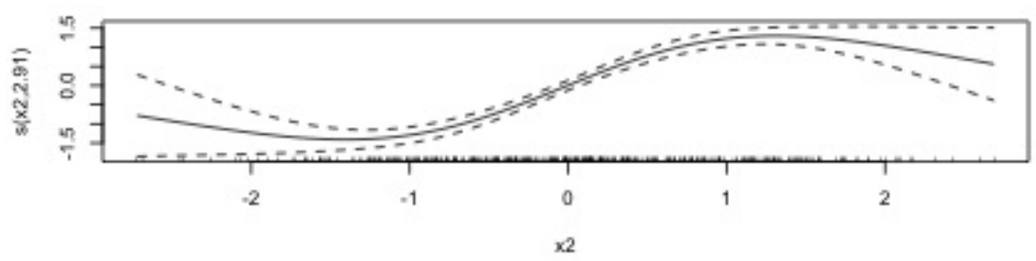
```
norm_model_3 = gam(y_norm~s(x1,k=12)+s(x2,k=12),method= "REML")
gam.check(norm_model_3)
```

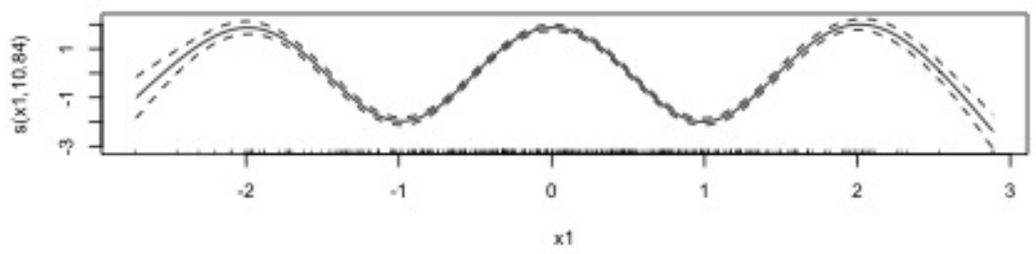
```
Method: REML   Optimizer: outer newton
full convergence after 8 iterations.
Gradient range [-1.136198e-08,6.812328e-13]
(score 334.2084 & scale 0.2485446).
Hessian positive definite, eigenvalue range [2.812271,198.6868].
Model rank =  23 / 23

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(x1) 11.000 10.848   0.976    0.28
s(x2) 11.000  7.948   0.946    0.12
```

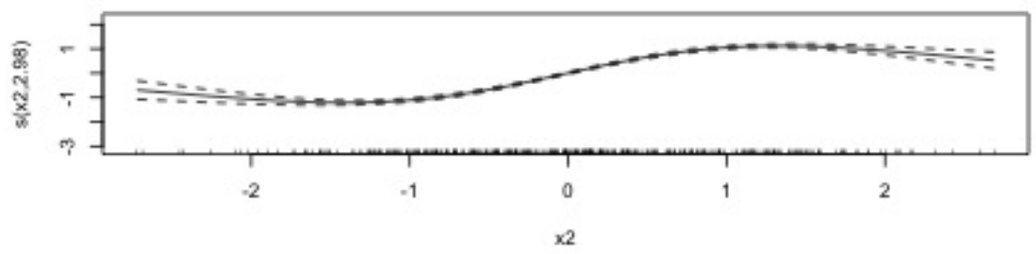
Checking basis size

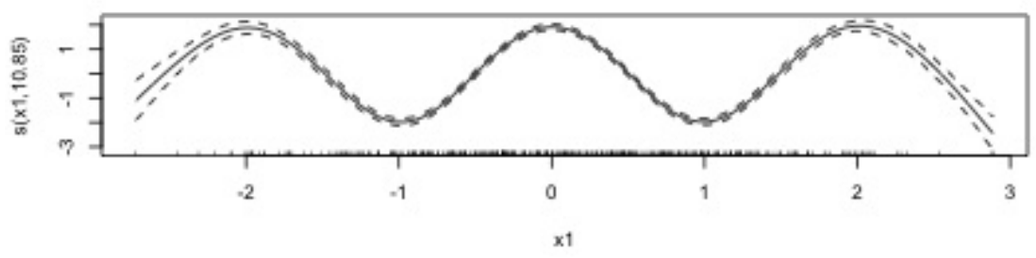
```
layout(matrix(1:6,ncol=2,byrow = T))
plot(norm_model_1);plot(norm_model_2);plot(norm_model_3)
```

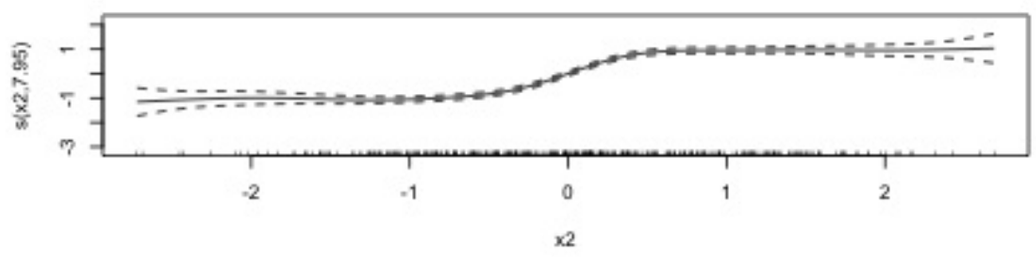













```
layout(1)
```

Using gam.check() part 2: visual checks

gam.check() plots

`gam.check()` creates 4 plots:

1. Quantile-quantile plots of residuals. If the model is right, should follow 1-1 line

2. Histogram of residuals

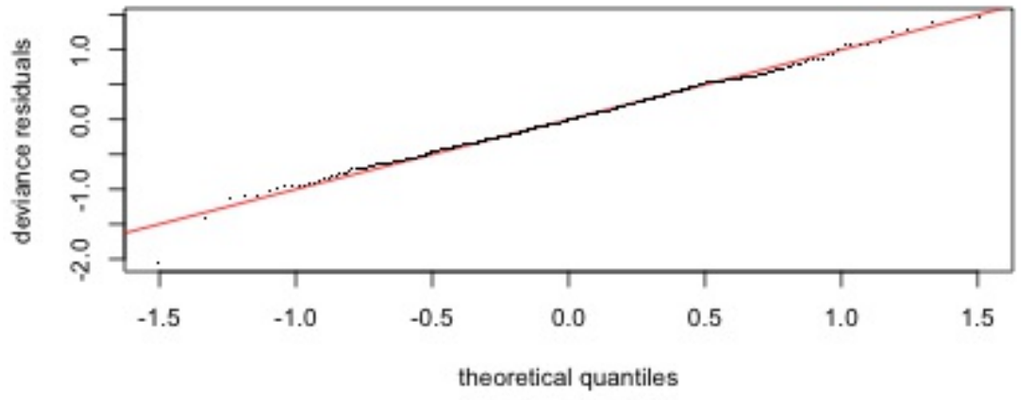
3. Residuals vs. linear predictor

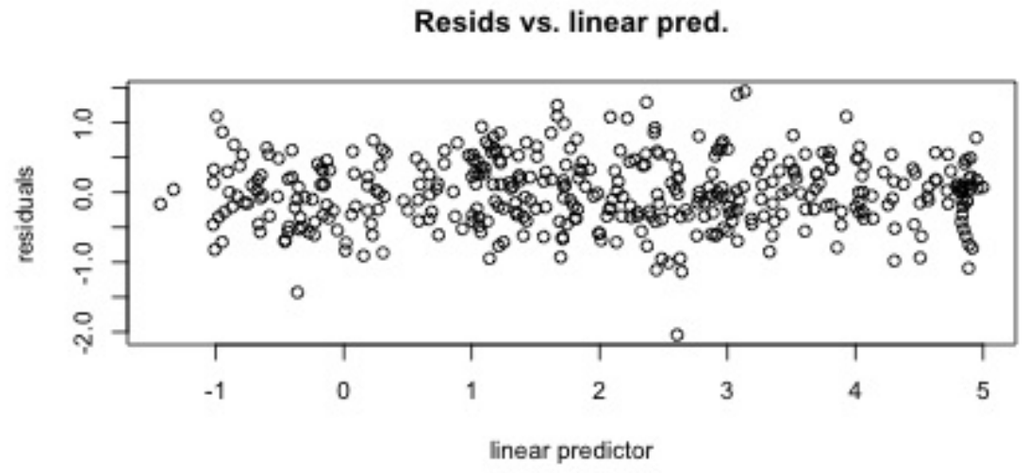
4. Observed vs. fitted values

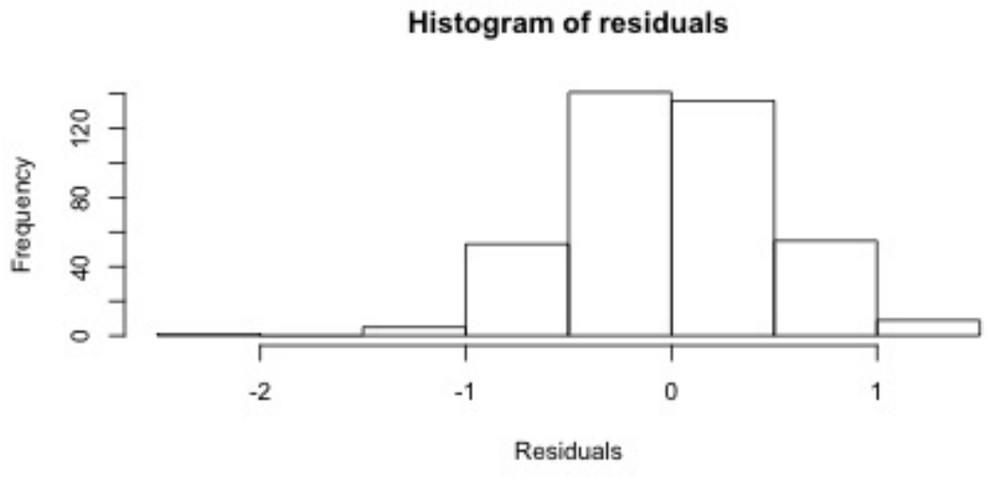
`gam.check()` uses deviance residuals by default

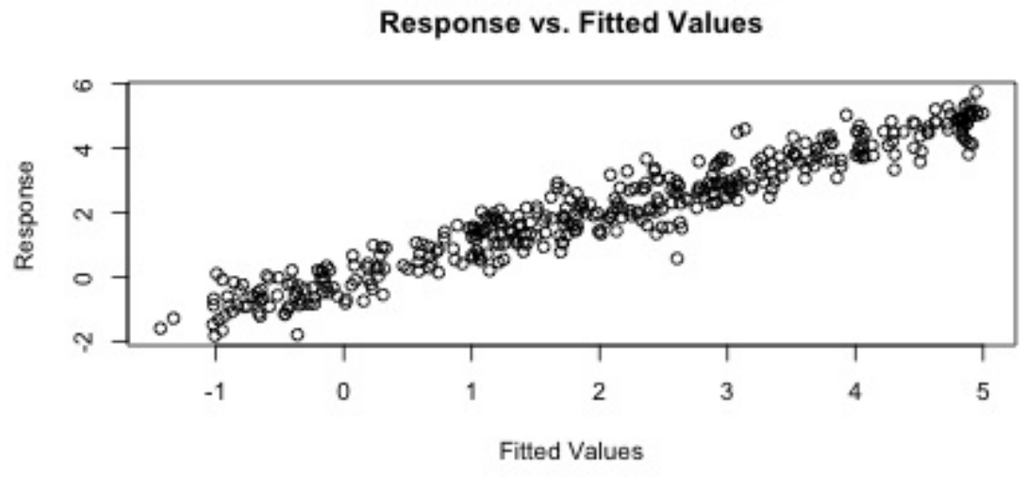
gam.check() plots: Gaussian data, Gaussian model

```
norm_model = gam(y_norm~s(x1,k=12)+s(x2,k=12),method= "REML")
gam.check(norm_model)
```



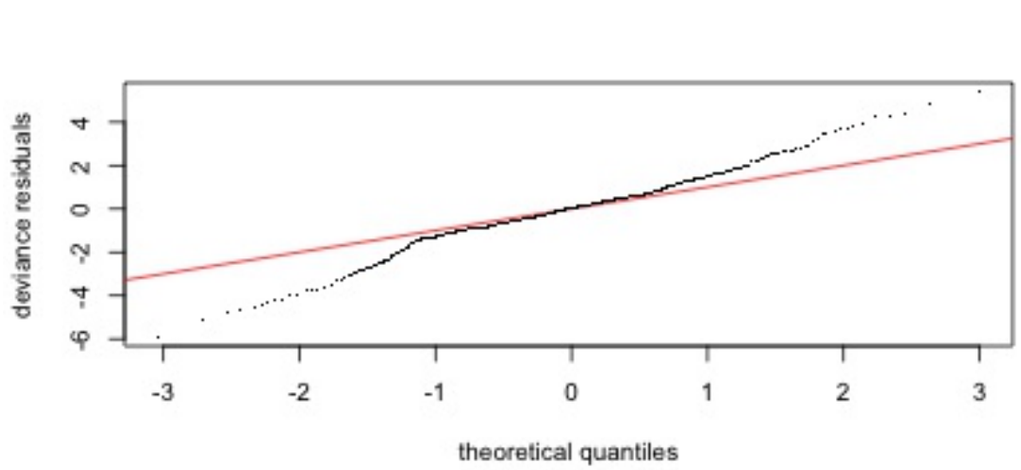


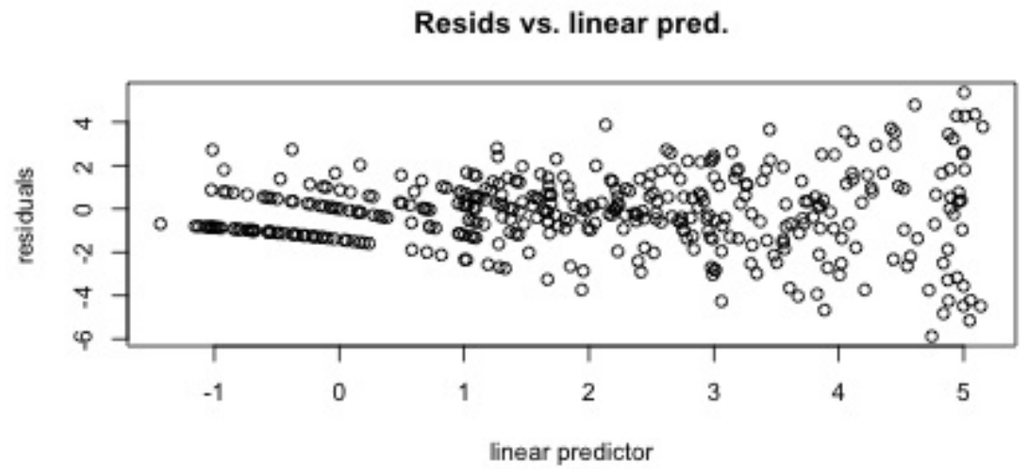


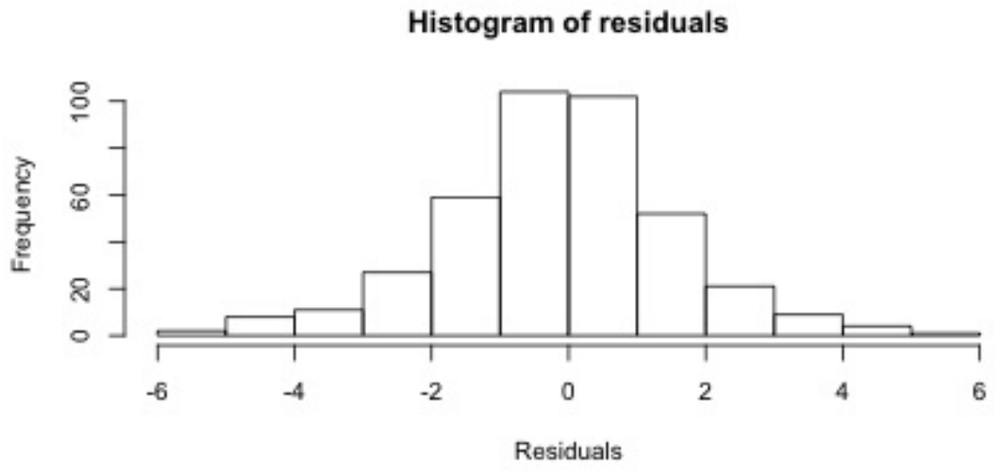


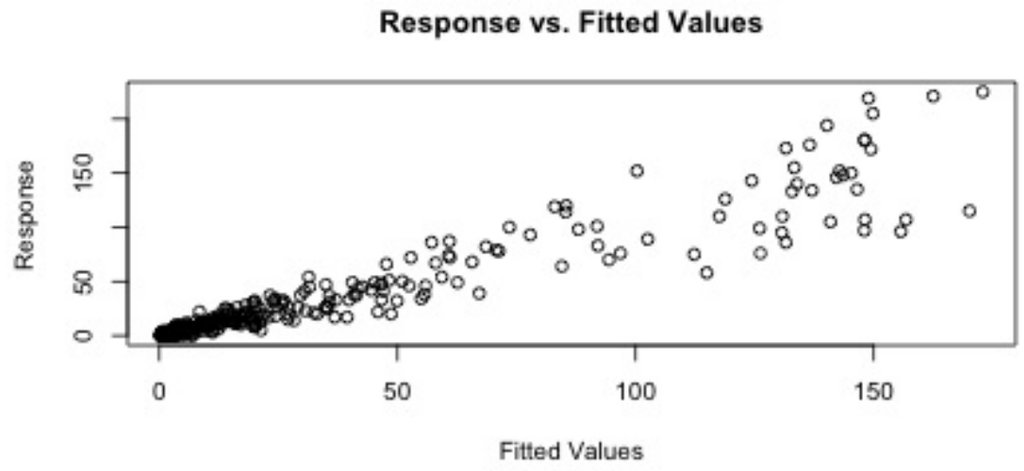
gam.check() plots: negative binomial data, Poisson model

```
pois_model =
gam(y_negbinom~s(x1,k=12)+s(x2,k=12),family=poisson,method=
"REML")
gam.check(pois_model)
```



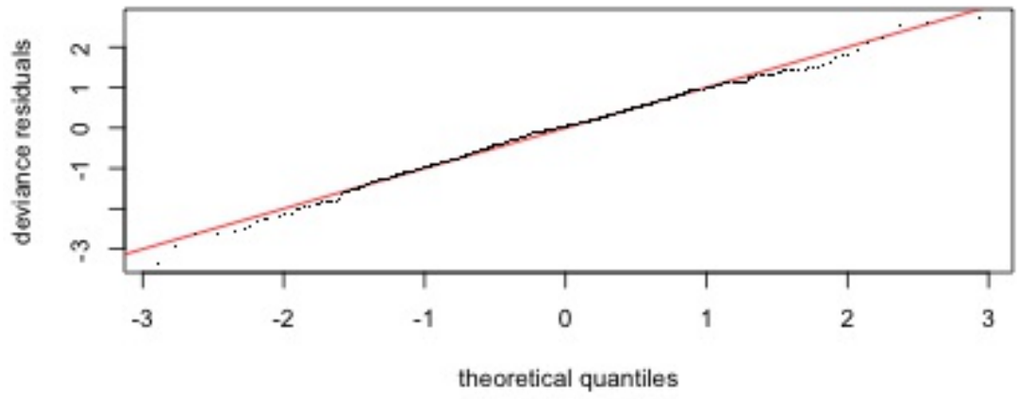


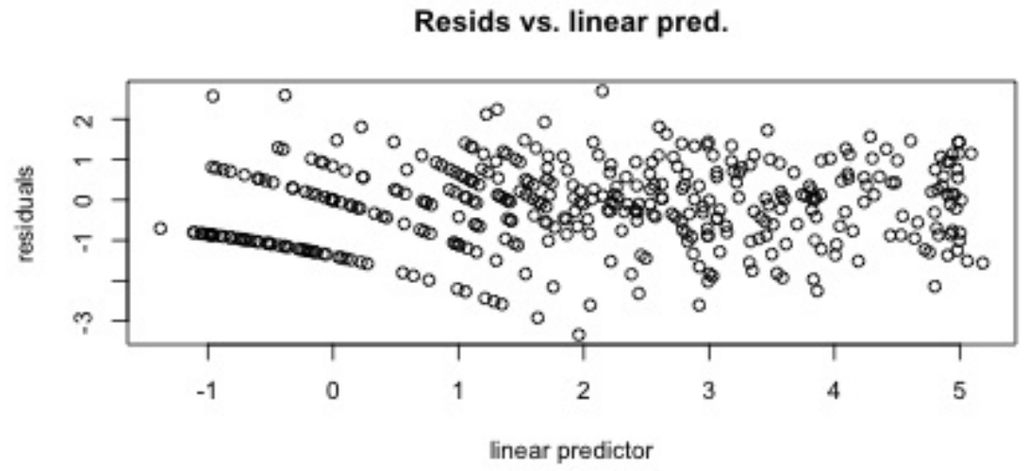


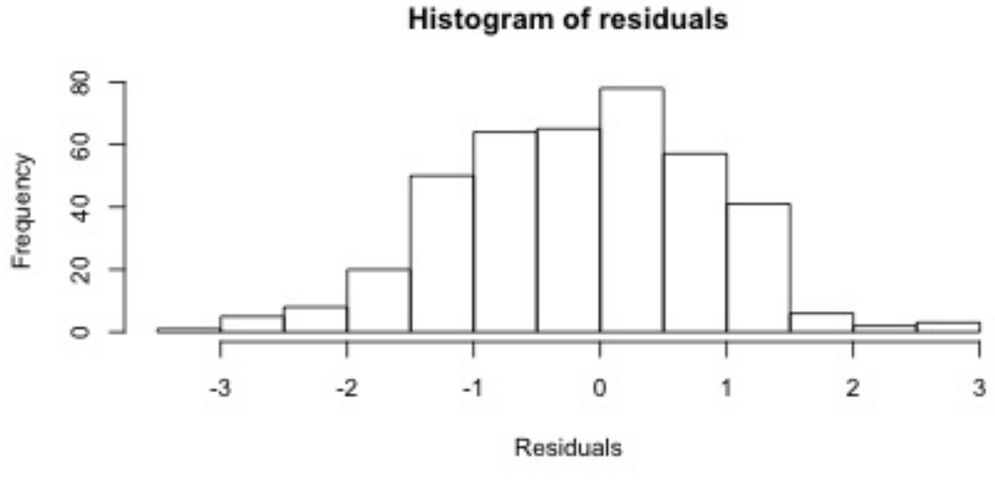


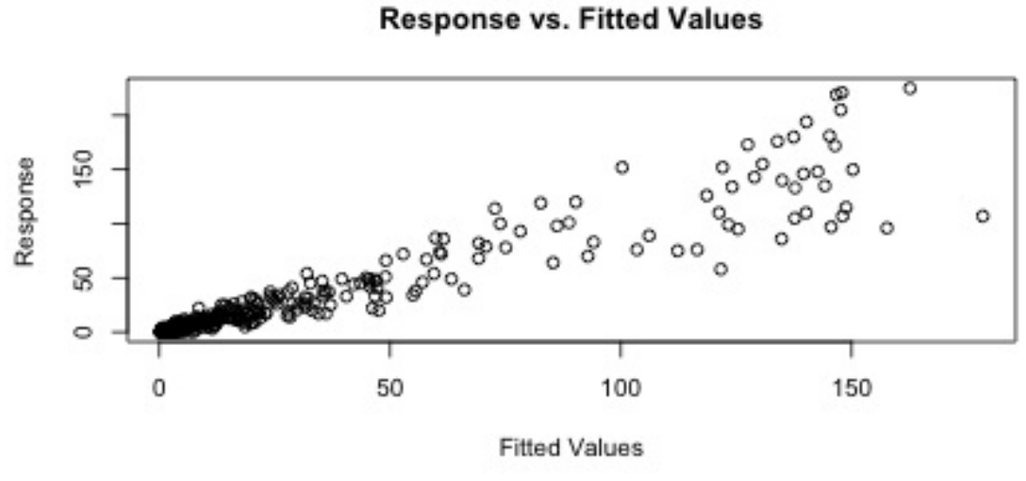
gam.check() plots: negative binomial data, negative binomial model

```
negbin_model =
gam(y_negbinom~s(x1,k=12)+s(x2,k=12),family=nb,method= "REML")
gam.check(negbin_model)
```









Exercises

1. Work with the `y_negbinom` data. Try fitting both Poisson and negative binomial model, with different degrees of freedom. Using model summaries and `gam.check`, determine which model requires more degrees of freedom to fit well. How do the fitted functions vary between the two?

2. Using the `y_binomial` data, fit a smooth model with `family=binomial`, and use `gam.check()` to determine the appropriate degrees of freedom. What do the `gam.check()` plots tell you about model fit?

Fixing Residuals

What are residuals?

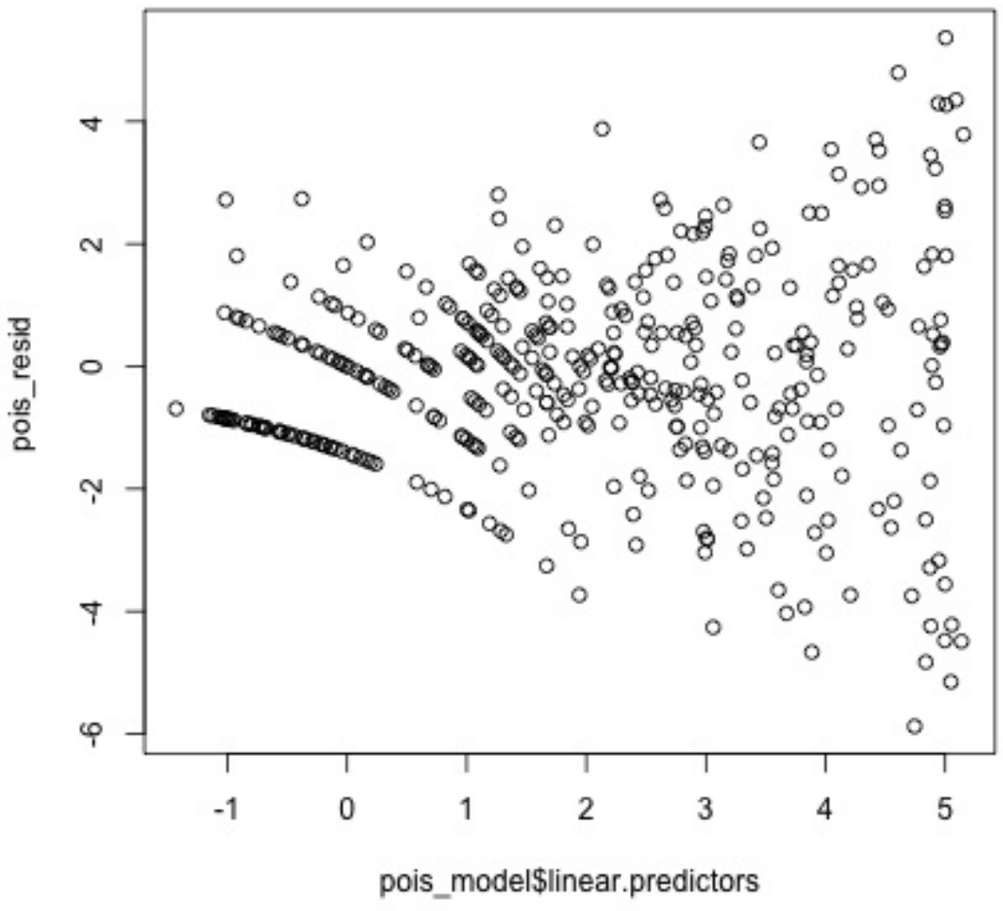
- Generally residuals = observed value - fitted value
- BUT hard to see patterns in these “raw” residuals
- Need to standardise – **deviance residuals**
- Residual sum of squares ⟹ linear model
  - deviance ⟹ GAM
- Expect these residuals ~ $\mathrm{N}(0, 1)$

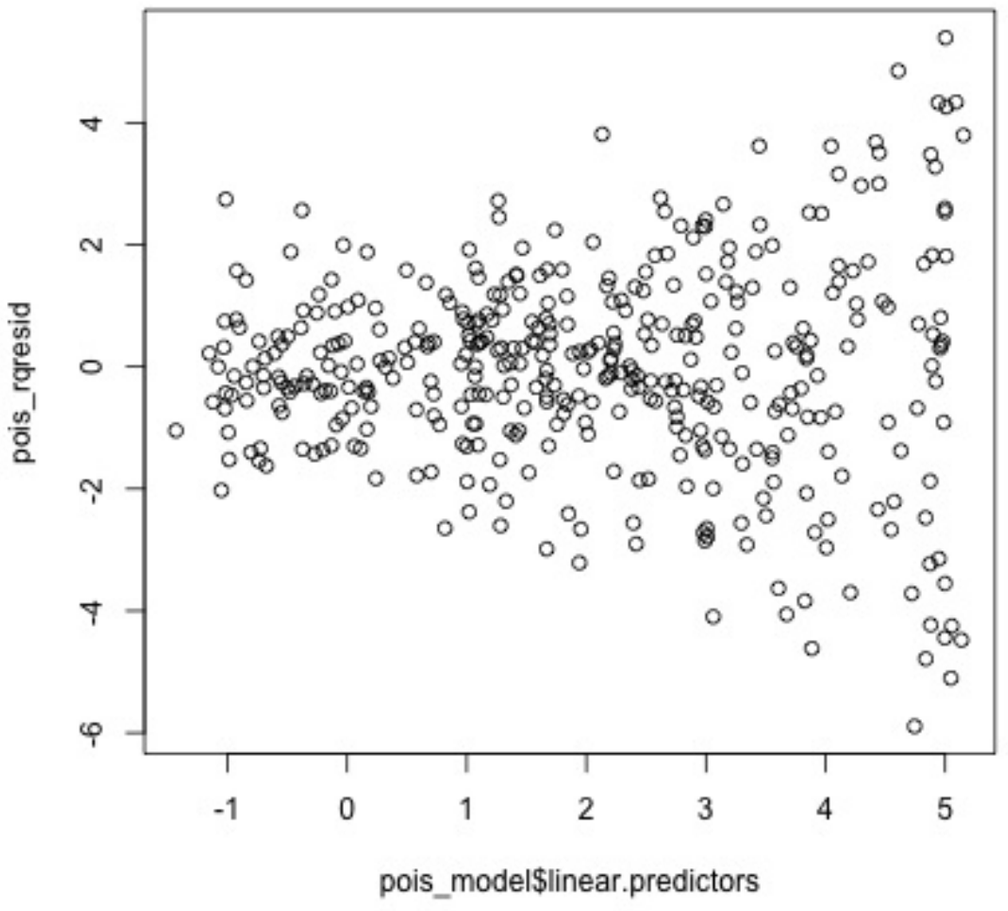
Shortcomings

- `gam.check` left side can be helpful
- Right side is victim of artifacts
- Need an alternative
- “Randomised quantile residuals” (*experimental*)
  - `rqresiduals`
  - Exactly normal residuals … if the model is right!

Using randomized quantile residuals

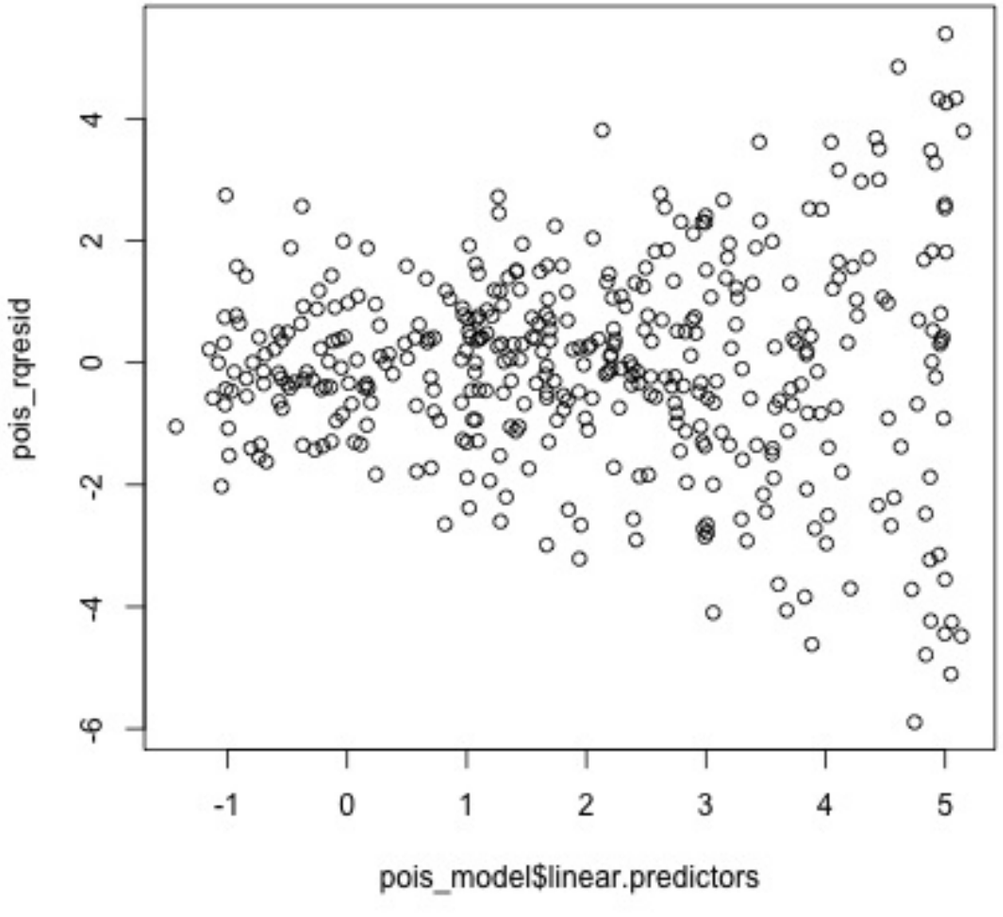
```
pois_model =
gam(y_negbinom~s(x1,k=12)+s(x2,k=12),family=poisson,method=
"REML")
pois_resid = residuals(pois_model, type="deviance")
pois_rqresid = rqresiduals(pois_model)
layout(matrix(1:2, nrow=1))
plot(pois_model$linear.predictors,pois_resid)
plot(pois_model$linear.predictors,pois_rqresid)
```

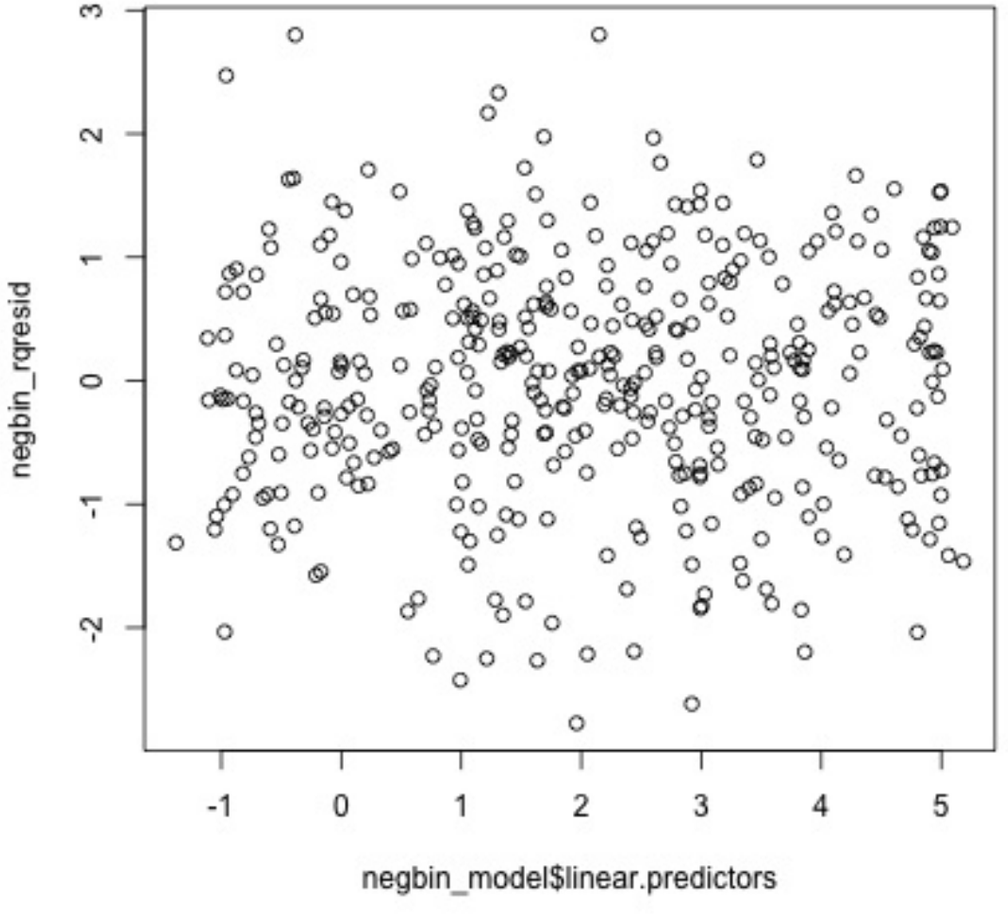




Using randomized quantile residuals

```
negbin_model =
gam(y_negbinom~s(x1,k=12)+s(x2,k=12),family=nb,method= "REML")
negbin_rqresid = rqresiduals(negbin_model)
layout(matrix(1:2, nrow=1))
plot(pois_model$linear.predictors,pois_rqresid)
plot(negbin_model$linear.predictors,negbin_rqresid)
```





Exercise

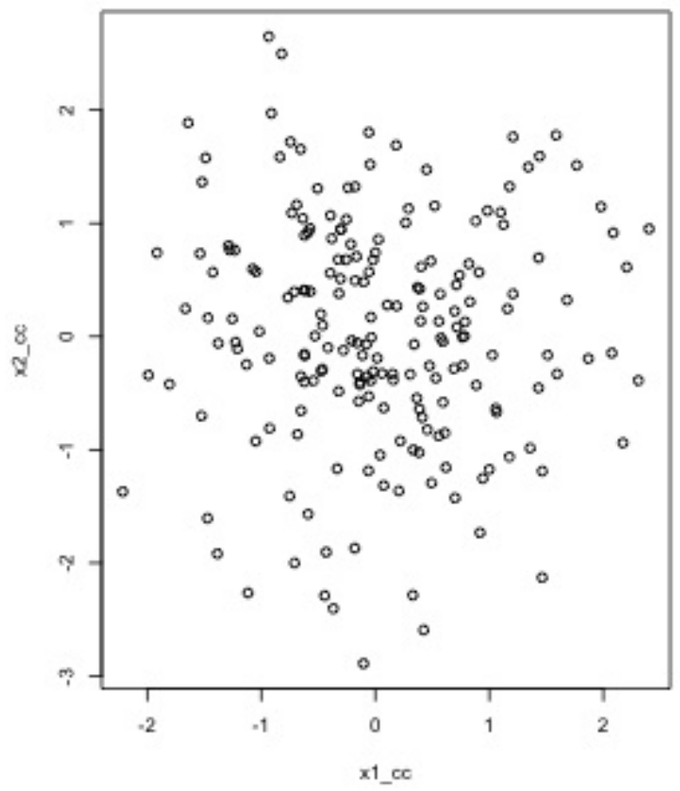
Use the `rqresiduals` function to test the model residuals for patterns from your binomial model from the previous exercise

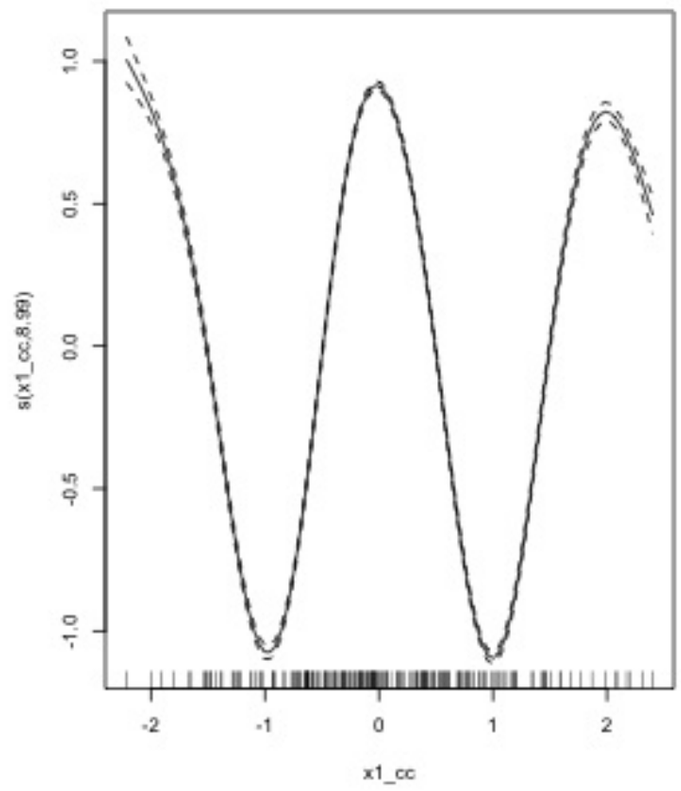
Concurvity

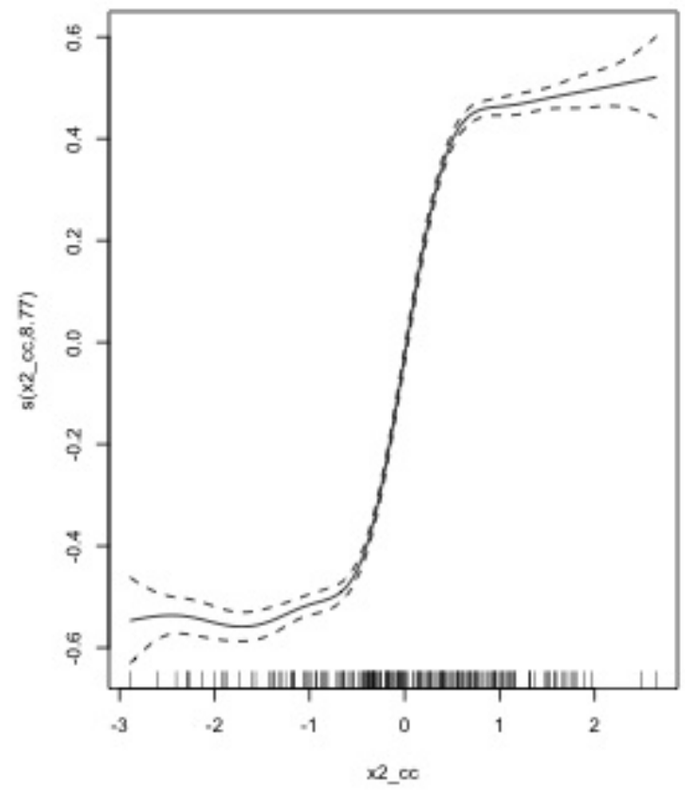
What is concurvity?

- Nonlinear measure, similar to co-linearity
- Measures, for each smooth term, how well this term could be approximated by
  - `concurvity(model, full=TRUE)`: some combination of all other smooth terms
  - `concurvity(model, full=FALSE)`: Each of the other smooth terms in the model (useful for identifying which terms are causing issues)

A demonstration



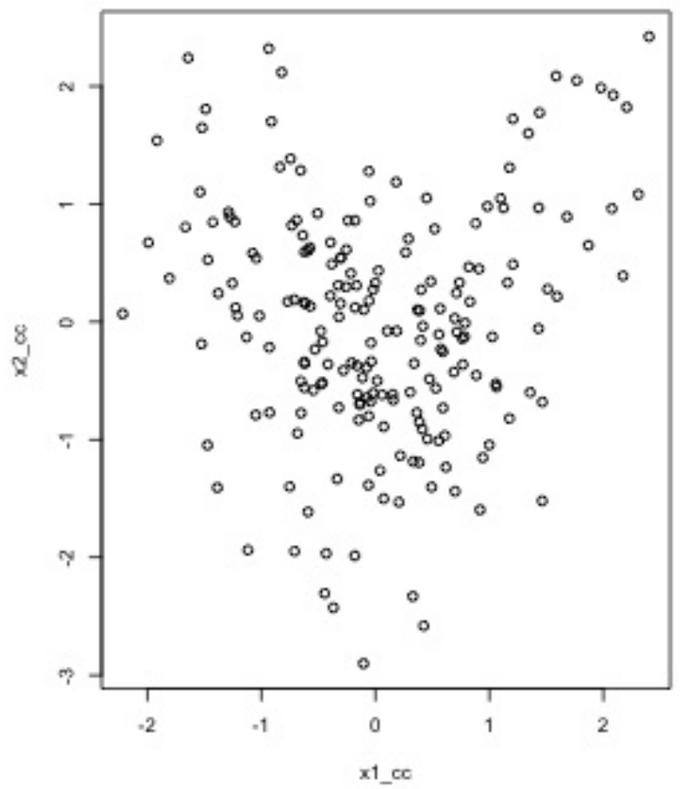


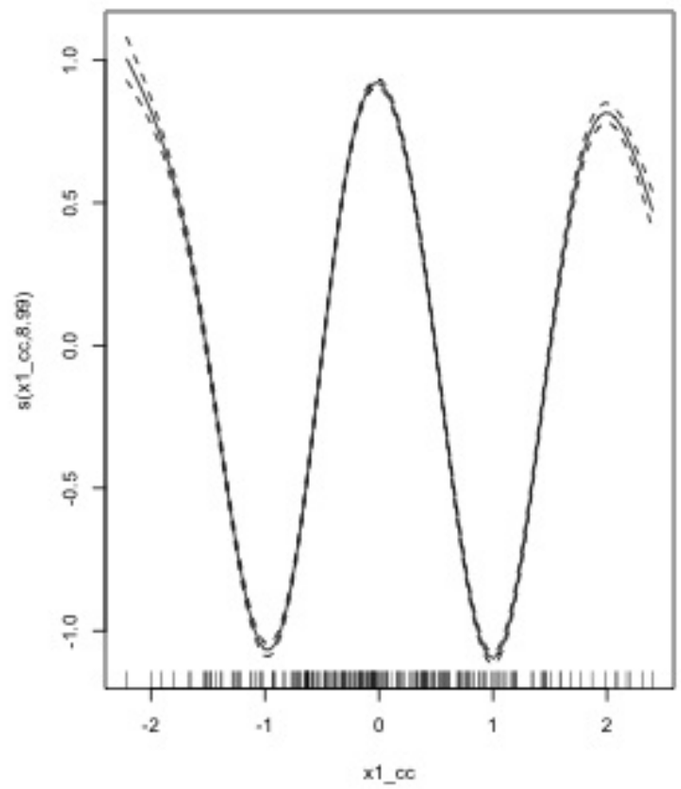


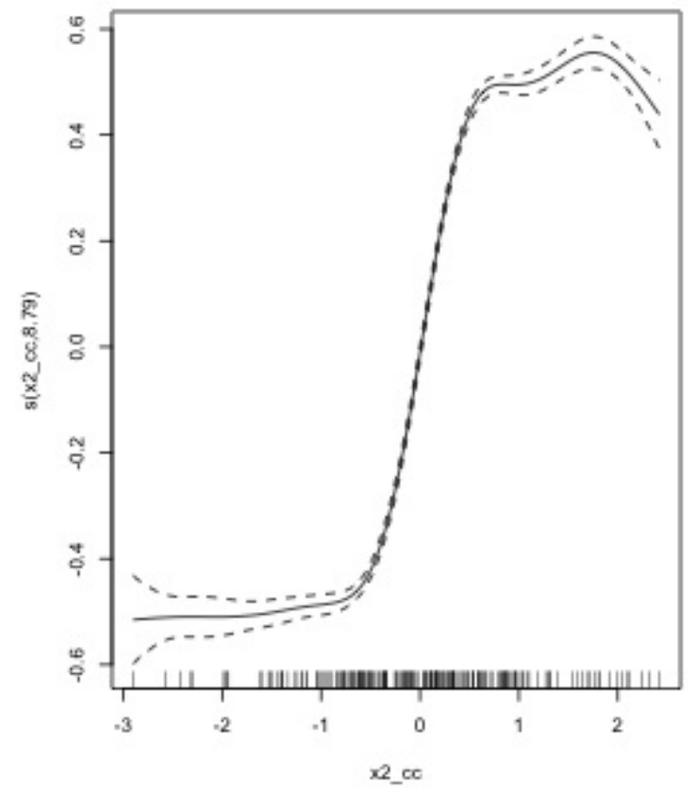

```
[1] "concurvity(m1)"
```

```
          para s(x1_cc) s(x2_cc)
worst        0     0.12     0.12
observed     0     0.07     0.04
estimate     0     0.04     0.04
```

A demonstration



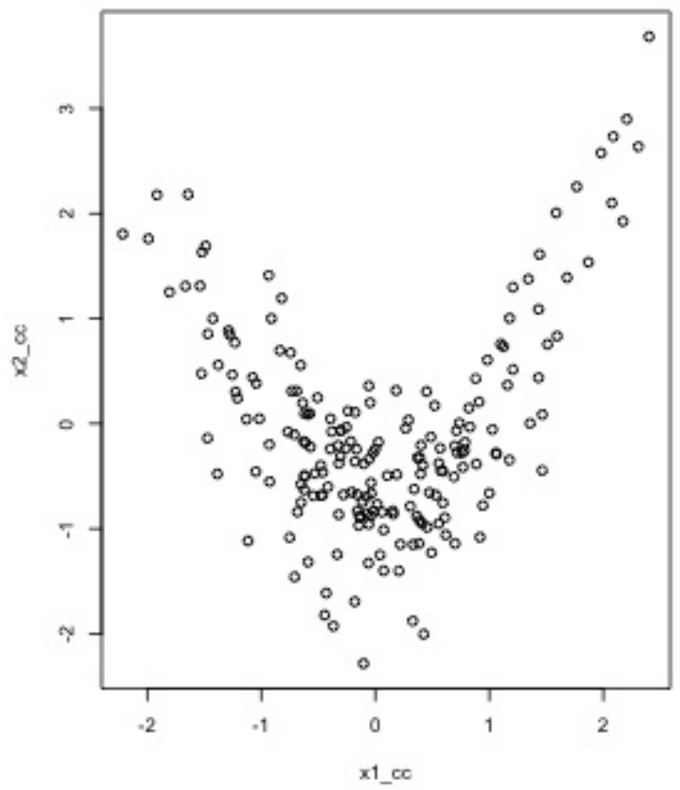


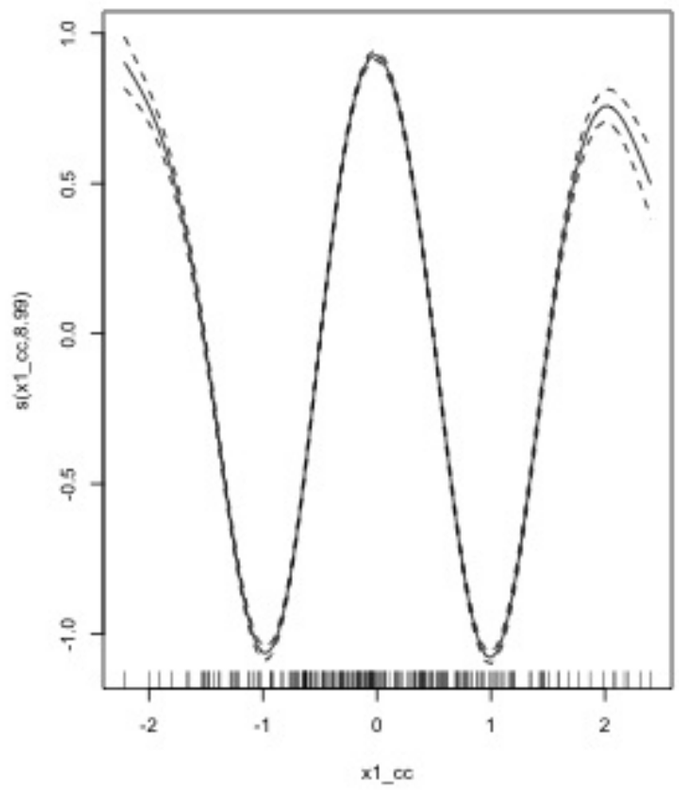


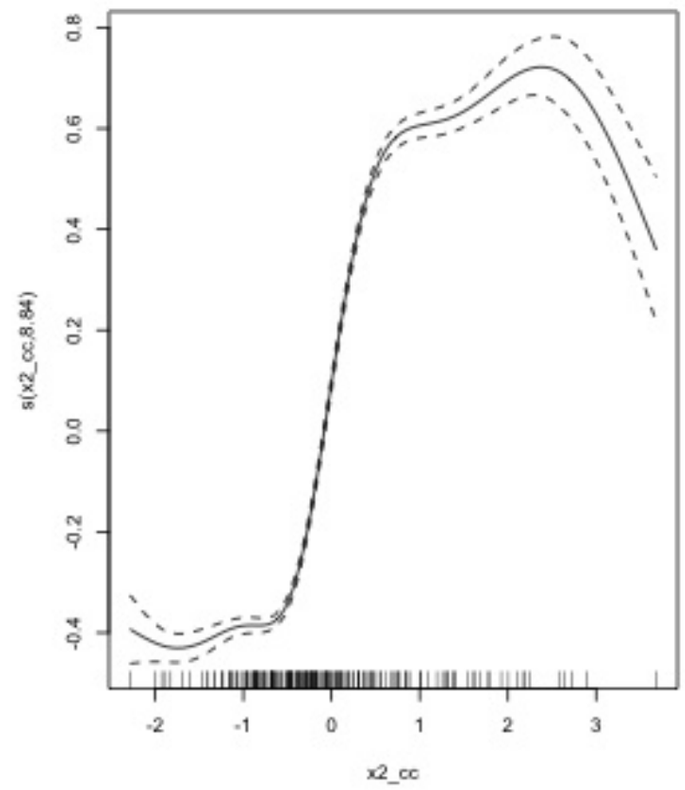

```
[1] "concurvity(m1, full=TRUE)"
```

```
          para s(x1_cc) s(x2_cc)
worst        0     0.32     0.32
observed     0     0.03     0.19
estimate     0     0.08     0.19
```

A demonstration



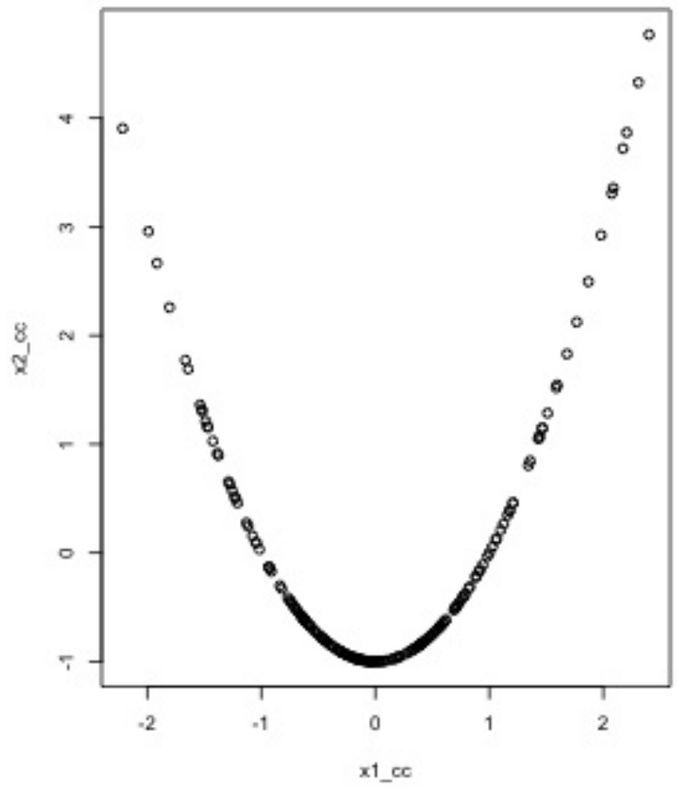


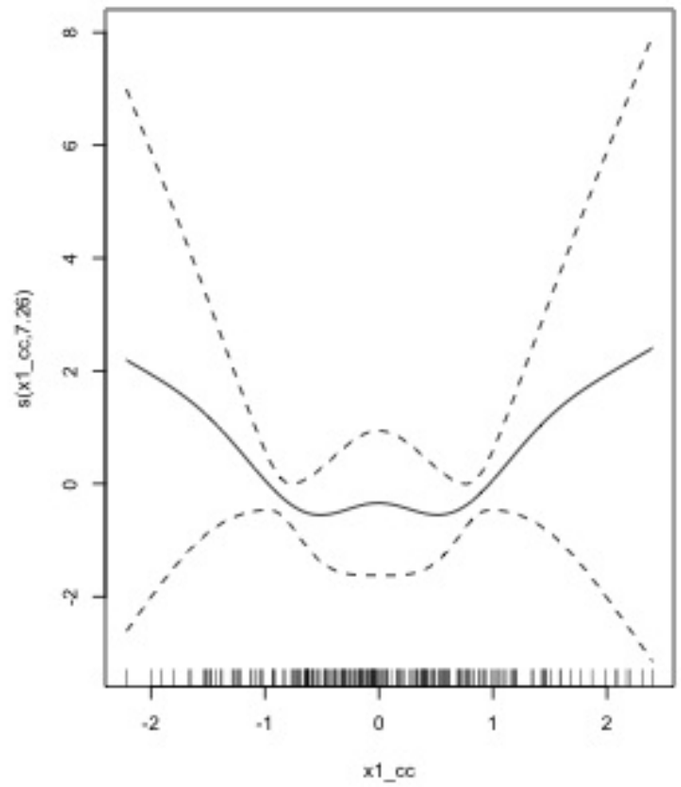


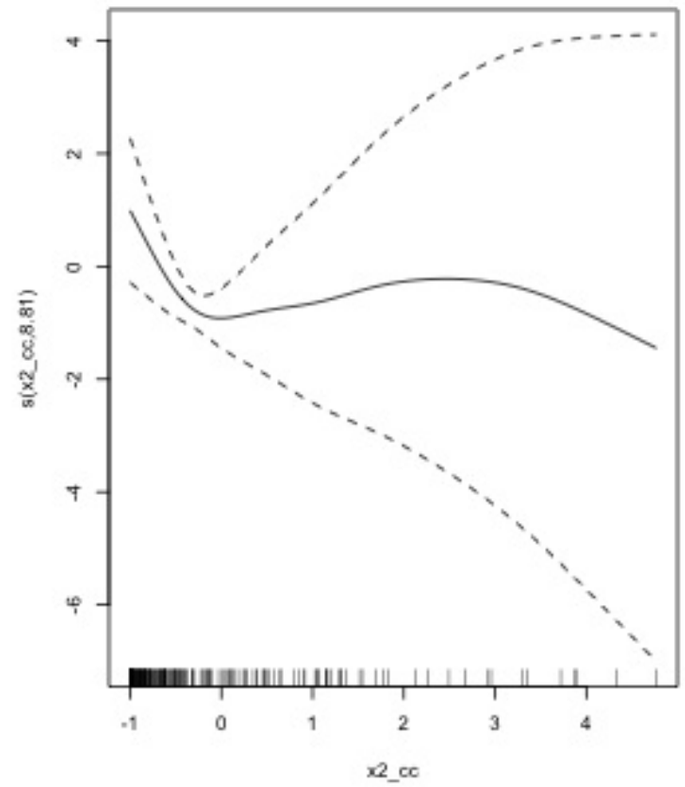

```
[1] "concurvity(m1, full=TRUE)"
```

```
          para s(x1_cc) s(x2_cc)
worst        0     0.84     0.84
observed     0     0.22     0.57
estimate     0     0.28     0.60
```

A demonstration








```
[1] "concurvity(m1, full=TRUE)"
```

```
          para s(x1_cc) s(x2_cc)
worst        0      1.0     1.00
observed     0      1.0     1.00
estimate     0      0.3     0.97
```

Concurvity: things to remember

- Can make your model unstable to small changes
- `cor(data)` not sufficient: use the `concurvity(model)` function
- Not always obvious from plots of smooths!!

Overall

Make sure to test your model! GAMs are powerful, but with great power…

You should at least:

1. Check if your smooths are sufficiently smooth
2. Test if you have the right distribution
3. Make sure there's no patterns left in your data
4. If you have time series, grouped, spatial, etc. data, check for dependencies

We'll get into testing for dependence later on in the extended examples.