

Fitting GAMs in practice

David L Miller

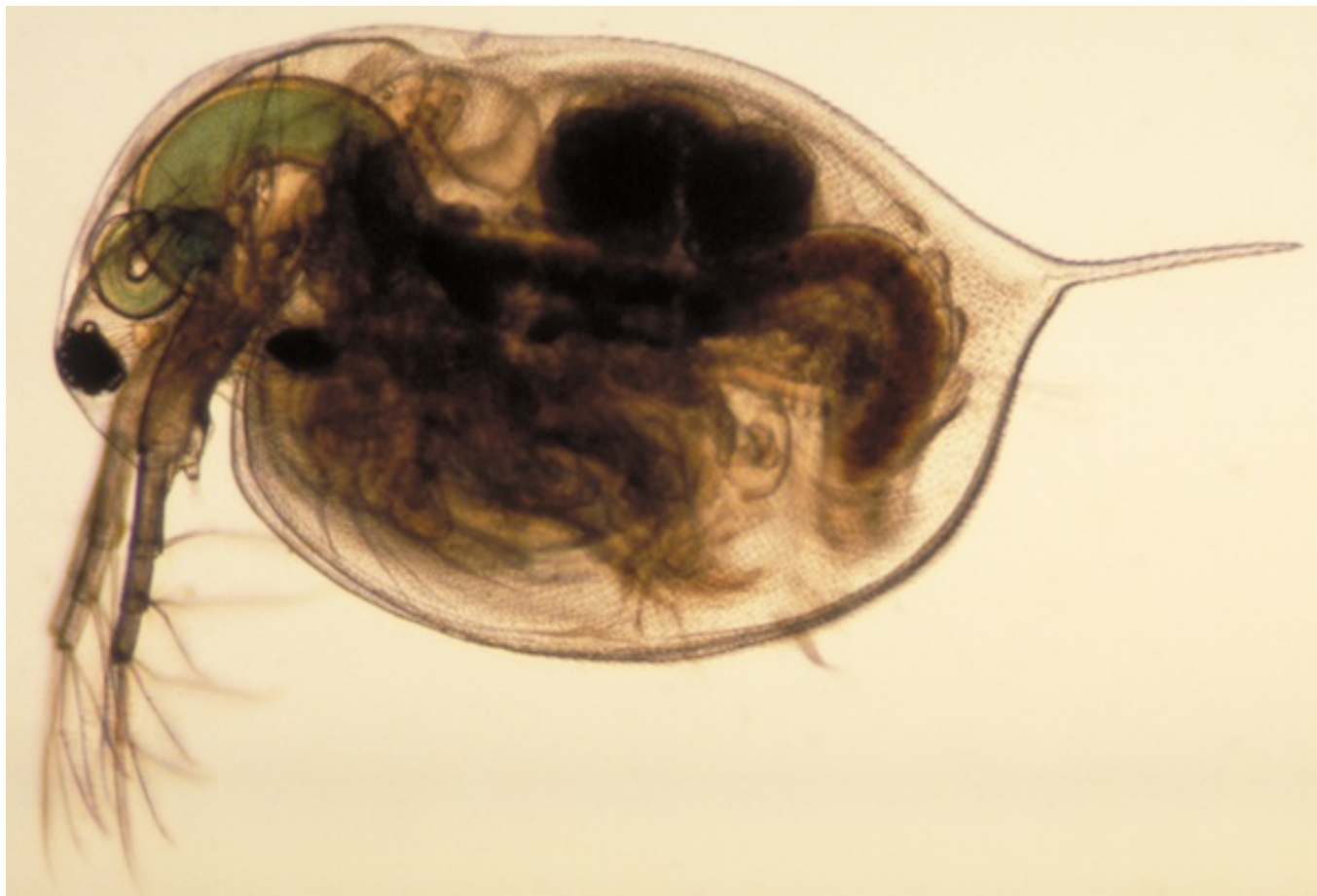
Overview

- Introduction to some data
- Fitting simple models
- Plotting simple models

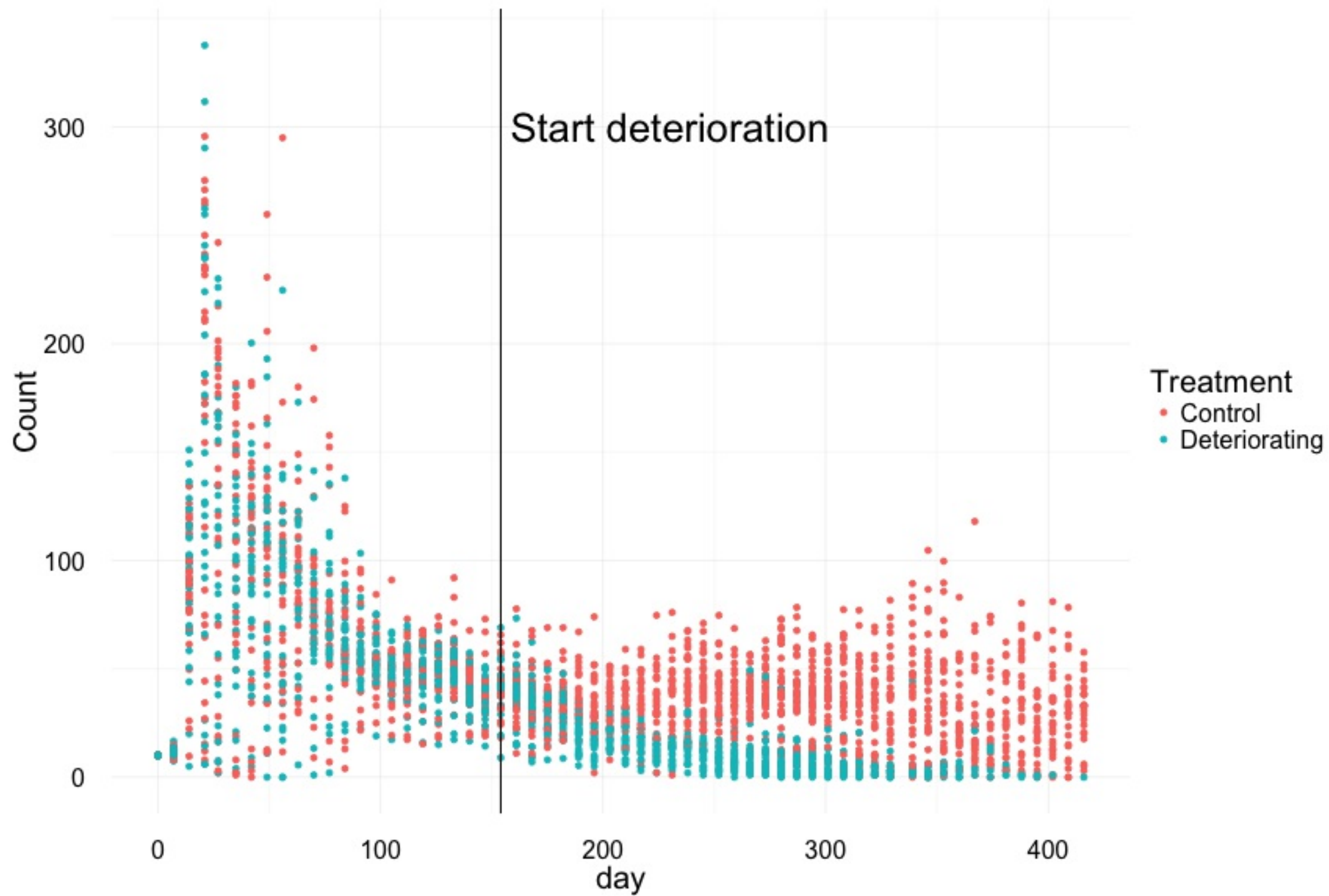
Data

Extinction experiment

- 60 populations of *Daphnia magna* in lab
- ½ in constant conditions, ½ in “deteriorating”
- Data from Drake & Griffen (Nature [doi:10.1038/nature09389](https://doi.org/10.1038/nature09389))



Extinction experiment



Inferential goals

- Make sense of all these dots!
- What are the “average” trends?
 - (What are non-average trends?)
- When do the deteriorating populations go extinct on average?

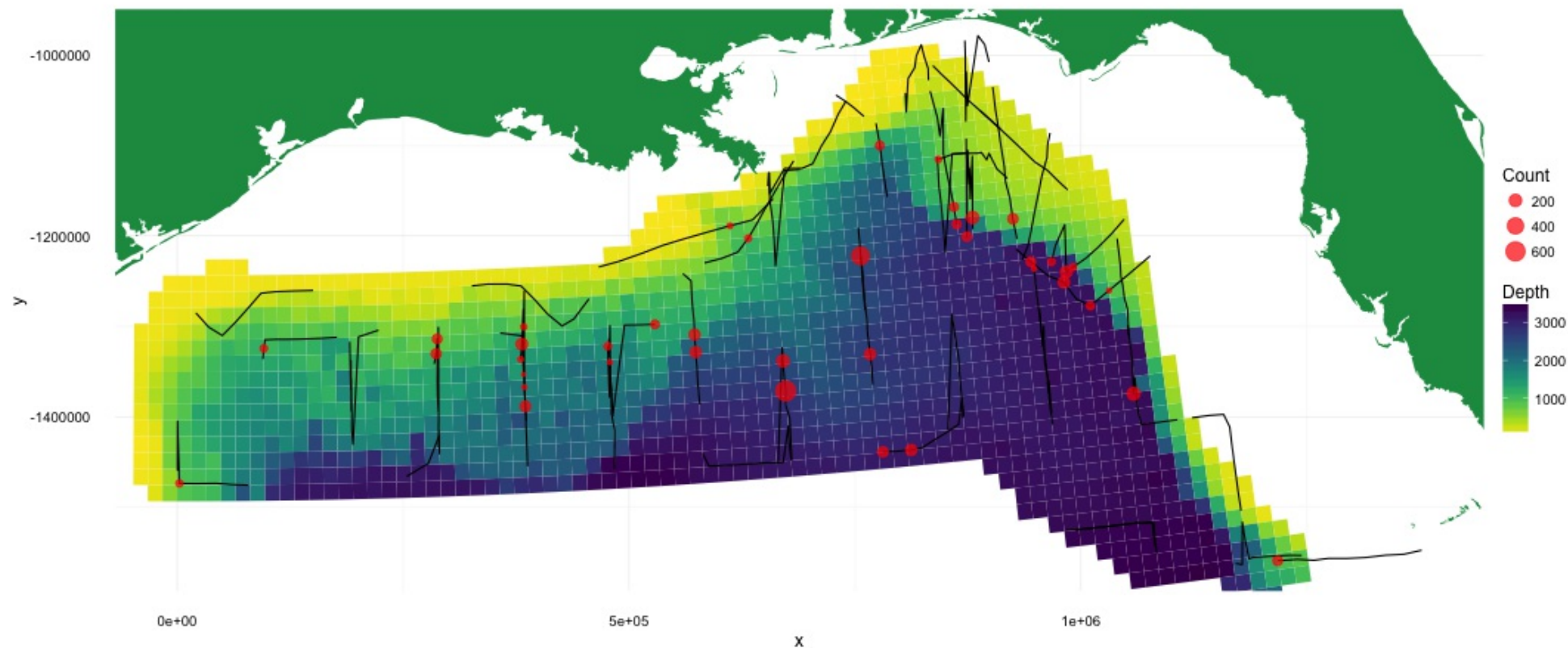
Pantropical spotted dolphins

- Example taken from Miller et al (2013)
- [Paper appendix](#) has a better analysis
- Simple example here, ignoring all kinds of important stuff!



Inferential aims

- How many dolphins are there?
- Where are the dolphins?
- What are they interested in?



Translating maths into R

A simple example:

$$y_i = \beta_0 + s(x) + s(w) + \epsilon_i$$

where $\epsilon_i \sim N(0, \sigma^2)$

Let's pretend that $y_i \sim \text{Normal}$

- linear predictor: `formula = y ~ s(x) + s(w)`
- response distribution: `family=gaussian()`
- data: `data=some_data_frame`

Putting that together

```
my_model <- gam(y ~ s(x) + s(w),  
               family = gaussian(),  
               data = some_data_frame,  
               method = "REML")
```

- method="REML" uses REML for smoothness selection (default is "GCV.Cp")

What about a practical
example?

Simple extinction example

```
dmagna_happy <- gam(Nhat~s(day, k=50),  
data=pop_happy, method="REML")  
dmagna_unhappy <- gam(Nhat~s(day,  
k=50), data=pop_unhappy, method="REML")
```

- Fit 2 models, one to control group, one to experimental group
- Nhat is the count of *D. magna* (estimated from 3 censuses)
- Model as a function of day

What did that do?

```
summary(dmagna_unhappy)
```

```
Family: gaussian  
Link function: identity
```

```
Formula:  
Nhat ~ s(day, k = 50)
```

```
Parametric coefficients:
```

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	38.9173	0.6959	55.93	<2e-16	***

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

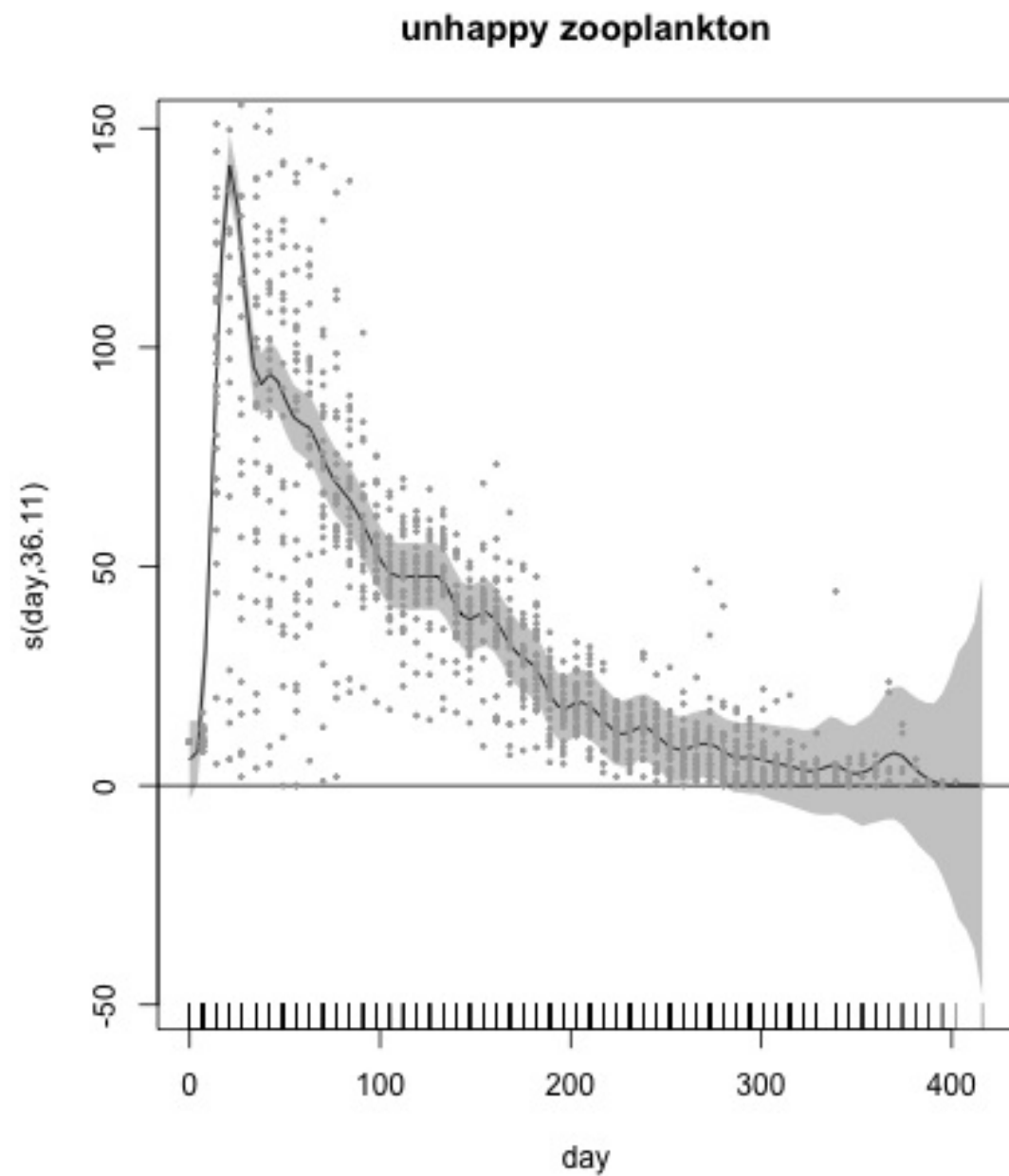
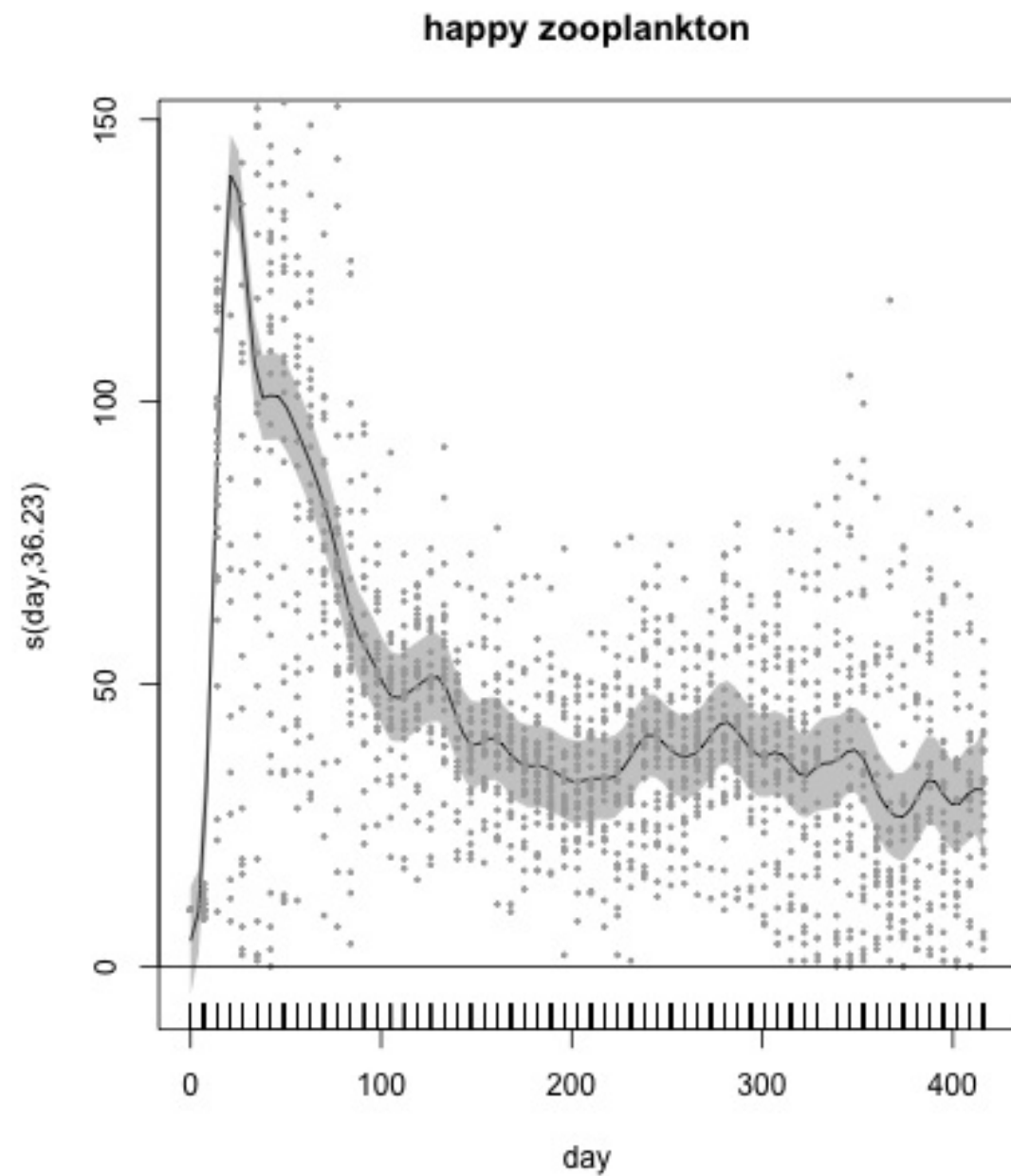
```
Approximate significance of smooth terms:
```

	edf	Ref.df	F	p-value	
s(day)	36.11	42.04	60.94	<2e-16	***

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

```
R-sq.(adj) = 0.657    Deviance explained = 66.6%  
-REML = 6296.6    Scale est. = 647.88    n = 1338
```

Cool, but what about a plot?



How did we do that?

```
plot(dmagna_unhappy, main="unhappy  
zooplankton",  
shift=mean(pop_unhappy$Nhat), scale=0,  
shade=TRUE)  
abline(h=0)
```

- `scale=0` puts each term on its own y axis
- Terms are centred, need `shift=` to re-centre (don't in models w/ > 1 term!)
- `shade=TRUE` makes ± 2 se into shaded area
- `main=` sets title, as usual

A simple dolphin model

```
library(mgcv)
dolphins_depth <- gam(count ~ s(depth)
+ offset(off.set),
                      data =
mexdolphins,
                      family =
quasipoisson(),
                      method = "REML")
```

- count is a function of depth
- off.set is the effort expended
- we have count data, try quasi-Poisson distribution

What did that do?

```
summary(dolphins_depth)
```

```
Family: quasipoisson  
Link function: log
```

```
Formula:  
count ~ s(depth) + offset(off.set)
```

```
Parametric coefficients:
```

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	-18.2344	0.8949	-20.38	<2e-16	***

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

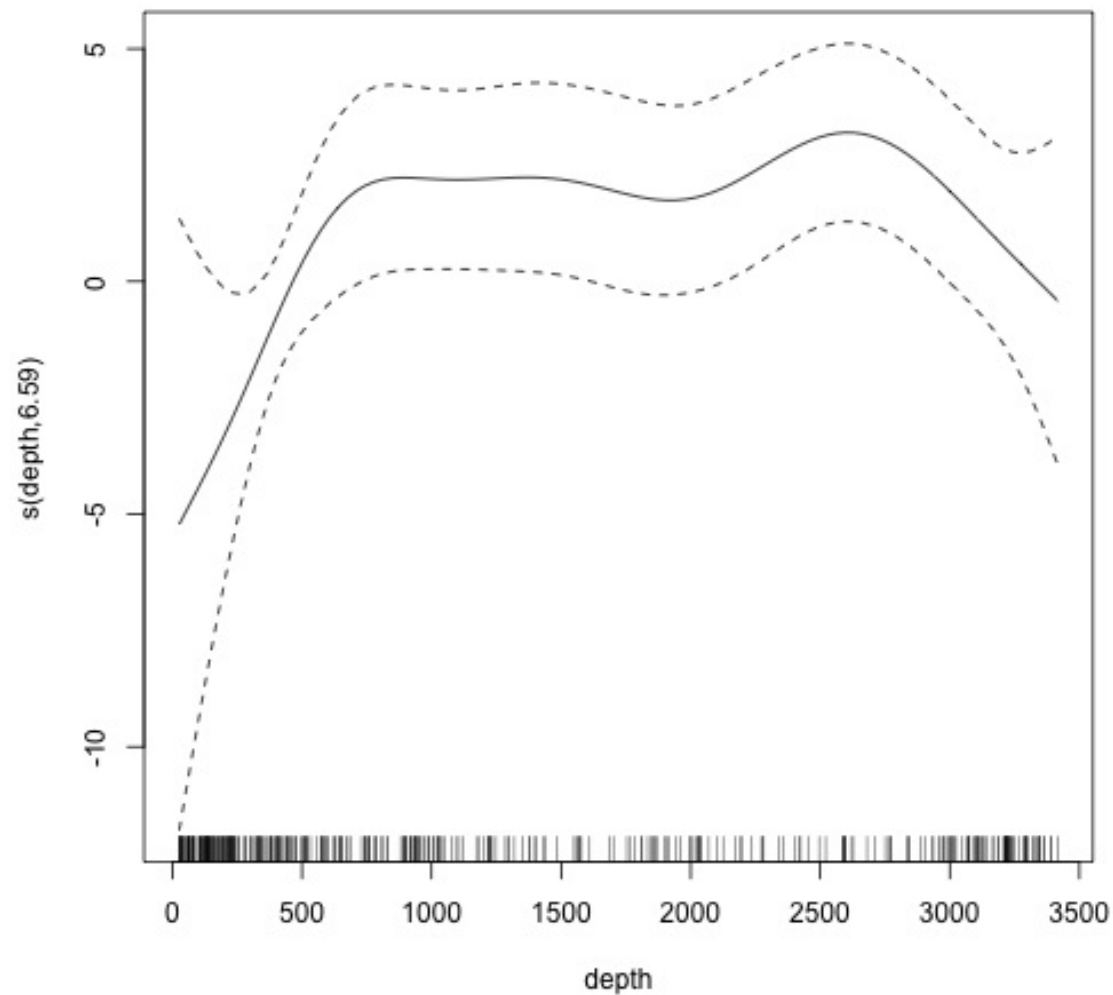
```
Approximate significance of smooth terms:
```

	edf	Ref.df	F	p-value	
s(depth)	6.592	7.534	2.329	0.0224	*

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

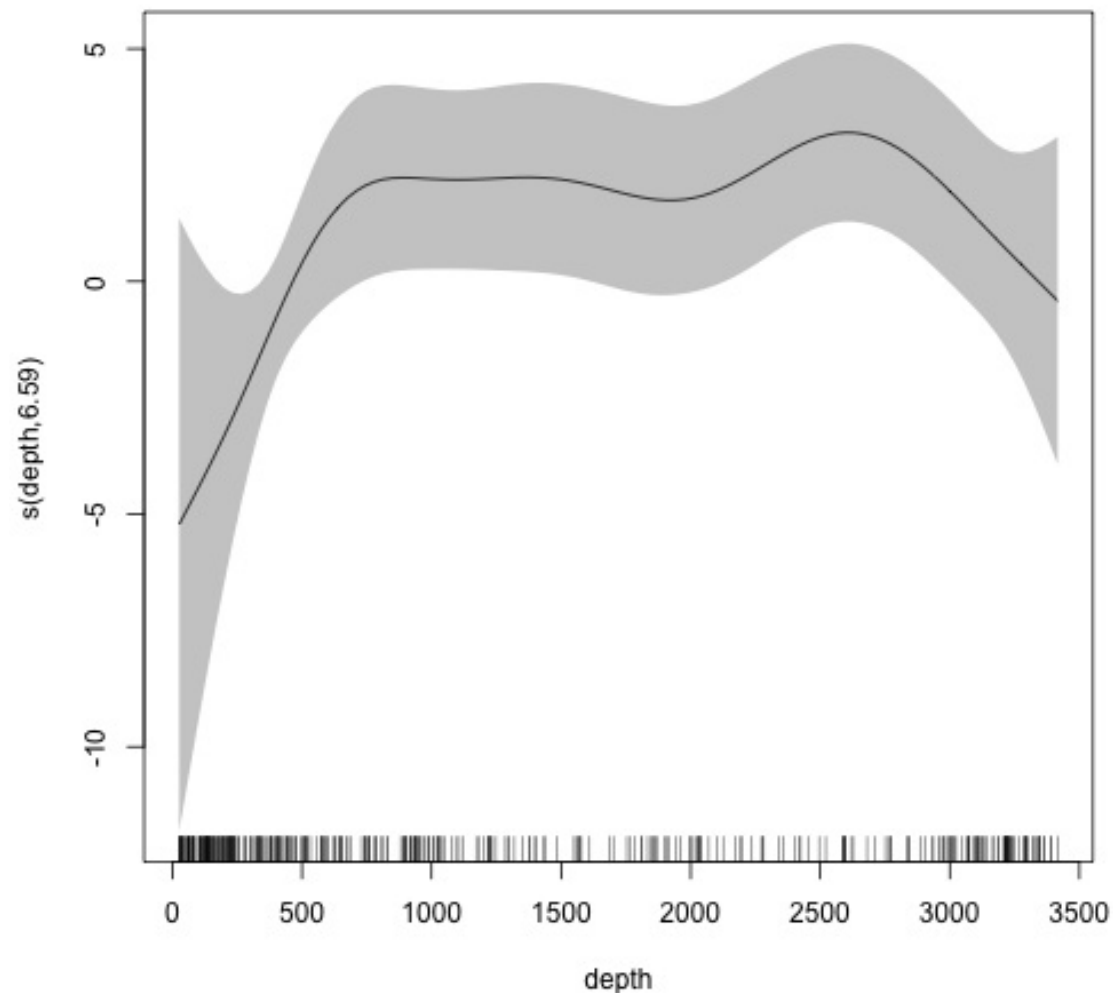
```
R-sq.(adj) = 0.0545    Deviance explained = 26.4%  
-REML = 948.28    Scale est. = 145.34    n = 387
```

Plotting



- `plot(dolphins_depth)`
- Dashed lines indicate ± 2 standard errors
- Rug plot
- On the link scale
- EDF on y axis

Plotting (shaded se)



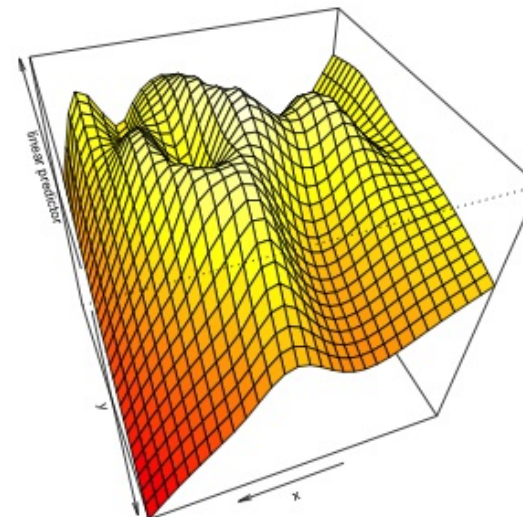
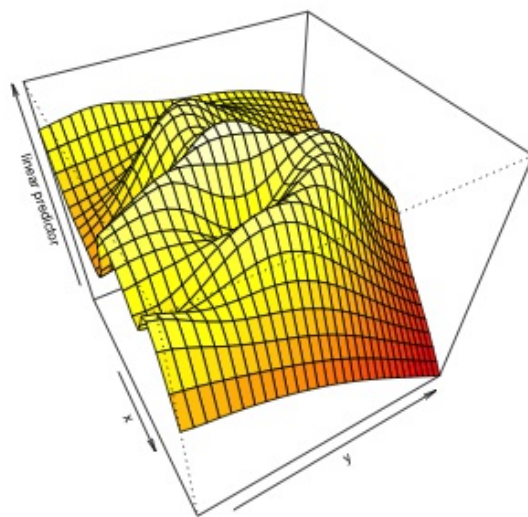
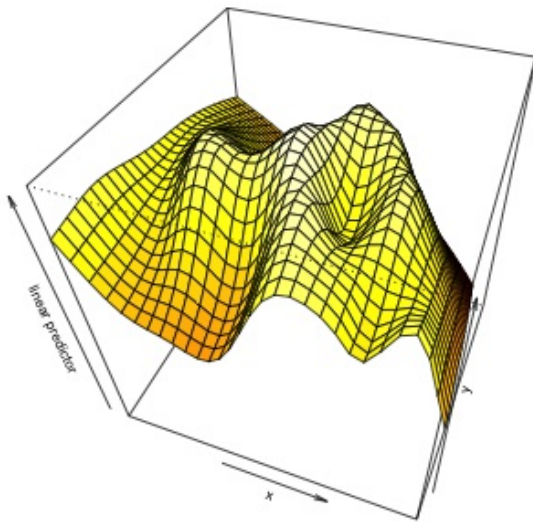
- `plot(dolphins_depth, shade=TRUE)`
- `?plot` has a **lot** of options
- `plotdat <- plot(model)` gives you the data that generates the plot

Thin plate regression splines

- Default basis
- One basis function per data point
- Reduce # basis functions (eigendecomposition)
- Fitting on reduced problem
- Multidimensional
- Wood (2003)

Bivariate terms

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



Adding a term

- Add a **surface** for location (x and y)
- Just use + for an extra term

```
dolphins_depth_xy <- gam(count ~  
s(depth) + s(x, y) + offset(off.set),  
                           data = mexdolphins,  
                           family=quasipoisson(),  
                           method="REML")
```

Summary

```
summary(dolphins_depth_xy)
```

```
Family: quasipoisson  
Link function: log
```

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

```
Parametric coefficients:
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-19.1933	0.9468	-20.27	<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(depth)	6.804	7.669	1.461	0.191
s(x,y)	23.639	26.544	1.358	0.114

```
R-sq.(adj) = 0.22    Deviance explained = 49.9%  
-REML = 923.9    Scale est. = 79.474    n = 387
```

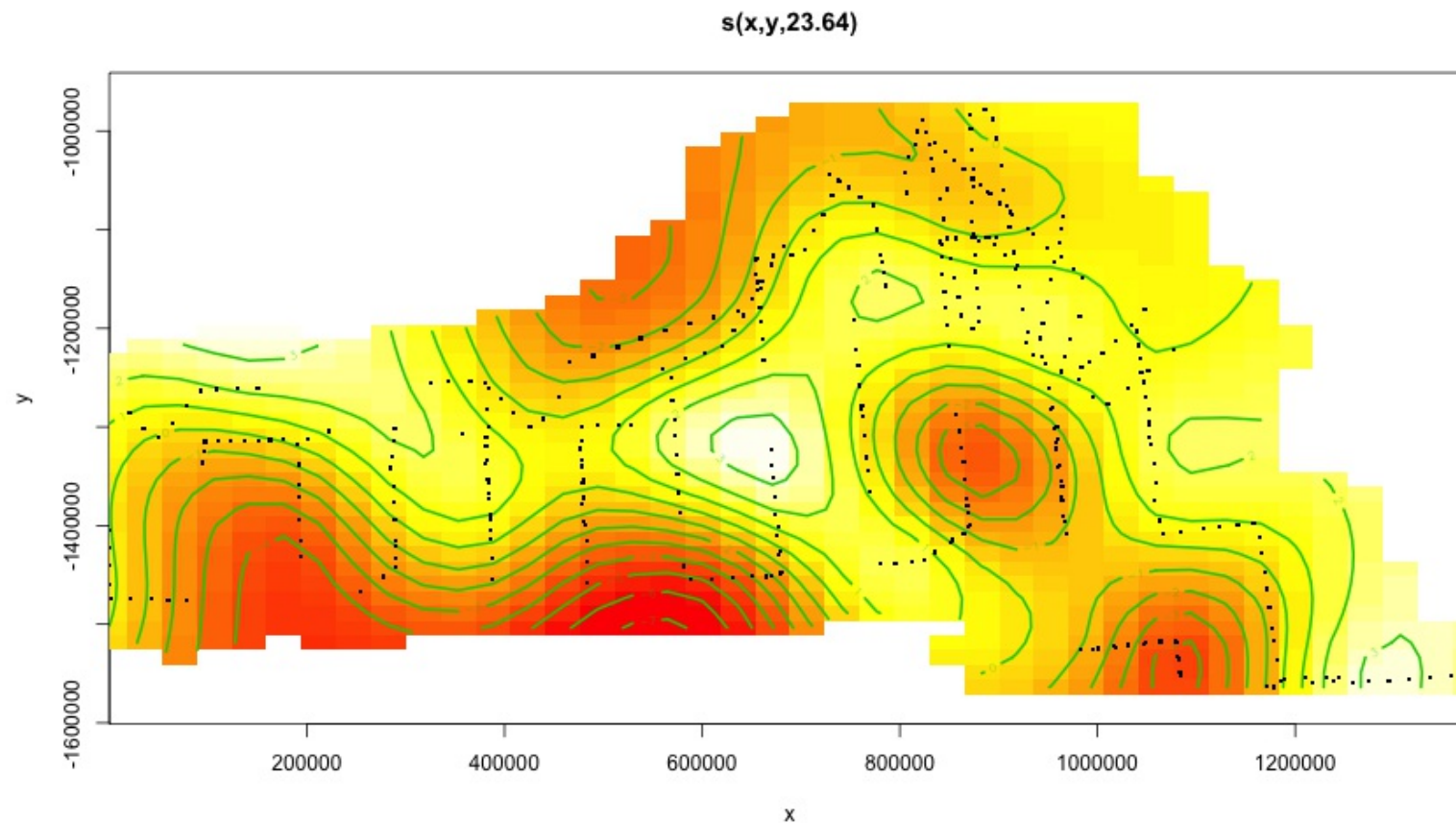
Plot x,y term

```
plot(dolphins_depth_xy, select=2,  
cex=3, asp=1, lwd=2, scheme=2)
```

- `scale=0`: each plot on different scale
- `select`= picks which smooth to plot
- `scheme=2` much better for bivariate terms
- `vis.gam()` also useful

Plot x,y term

```
plot(dolphins_depth_xy, select=2,  
cex=3, asp=1, lwd=2, scheme=2)
```



Fitting/plotting GAMs summary

- `gam` does all the work
- very similar to `glm`
- `s` indicates a smooth term
- `summary`, `plot` methods