Beyond the exponential family

Eric Pedersen, Gavin Simpson, David Miller
August 6th, 2016

Away from the exponential family

Most glm families (Poisson, Gamma, Gaussian, Binomial) are *exponential families*

Away from the exponential family

Most glm families (Poisson, Gamma, Gaussian, Binomial) are *exponential families*

$$f(x|\theta) \sim \exp(\sum_i \eta_i(\theta) T_i(x) - A(\theta))$$

Away from the exponential family

Most glm families (Poisson, Gamma, Gaussian, Binomial) are *exponential families*

$$f(x|\theta) \sim \exp\left(\sum_i \eta_i(\theta) T_i(x) - A(\theta)\right)$$

- Computationally easy

Away from the exponential family

Most glm families (Poisson, Gamma, Gaussian, Binomial) are *exponential families*

$$f(x|\theta) \sim \exp(\sum_i \eta_i(\theta) T_i(x) - A(\theta))$$

- Computationally easy
- Has sufficient statistics: easier to estimate parameter variance

Away from the exponential family

Most glm families (Poisson, Gamma, Gaussian, Binomial) are *exponential families*

$$f(x|\theta) \sim \exp\left(\sum_i \eta_i(\theta)T_i(x) - A(\theta)\right)$$

- Computationally easy
- Has sufficient statistics: easier to estimate parameter variance
- … but it doesn't describe everything

Away from the exponential family

Most glm families (Poisson, Gamma, Gaussian, Binomial) are *exponential families*

$$f(x|\theta) \sim \exp\left(\sum_i \eta_i(\theta) T_i(x) - A(\theta)\right)$$

- Computationally easy
- Has sufficient statistics: easier to estimate parameter variance
- ... but it doesn't describe everything
- `mgcv` has expanded to cover many new families

Away from the exponential family

Most glm families (Poisson, Gamma, Gaussian, Binomial) are *exponential families*

$$f(x|\theta) \sim \exp\left(\sum_i \eta_i(\theta)T_i(x) - A(\theta)\right)$$

- Computationally easy
- Has sufficient statistics: easier to estimate parameter variance
- ... but it doesn't describe everything
- `mgcv` has expanded to cover many new families
- Lets you model a much wider range of scenarios with smooths

What we’ll cover

- “Counts”: Negative binomial and Tweedie distributions
- Modelling proportions with the Beta distribution
- Robust regression with the Student's t distribution
- Ordered and unorderd categorical data
- Multivariate normal data
- Modelling exta zeros with zero-inflated and adjusted families

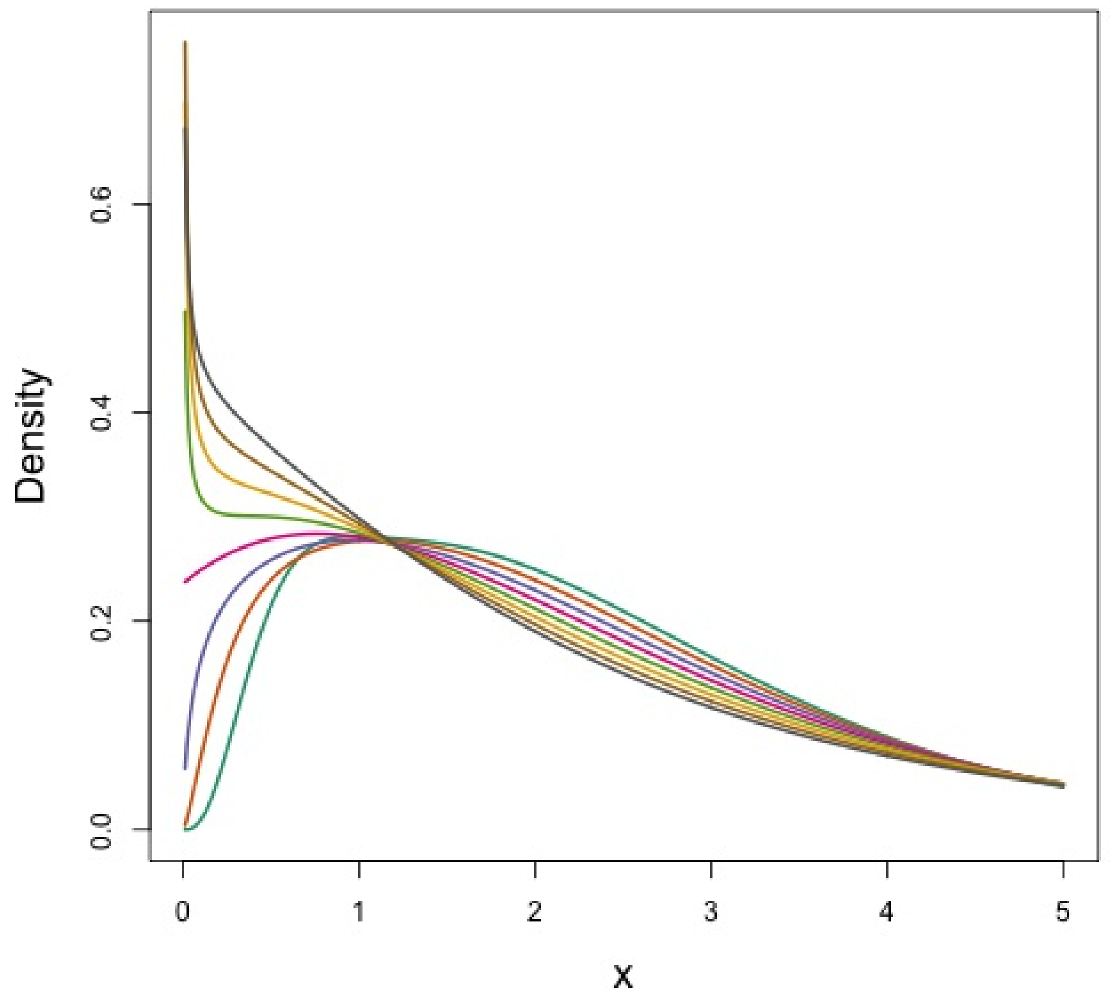
- *NOTE*: All the distributions we're covering here have their own quirks. Read the help files carefully before using them!

Modelling ”counts”

Counts and count-like things

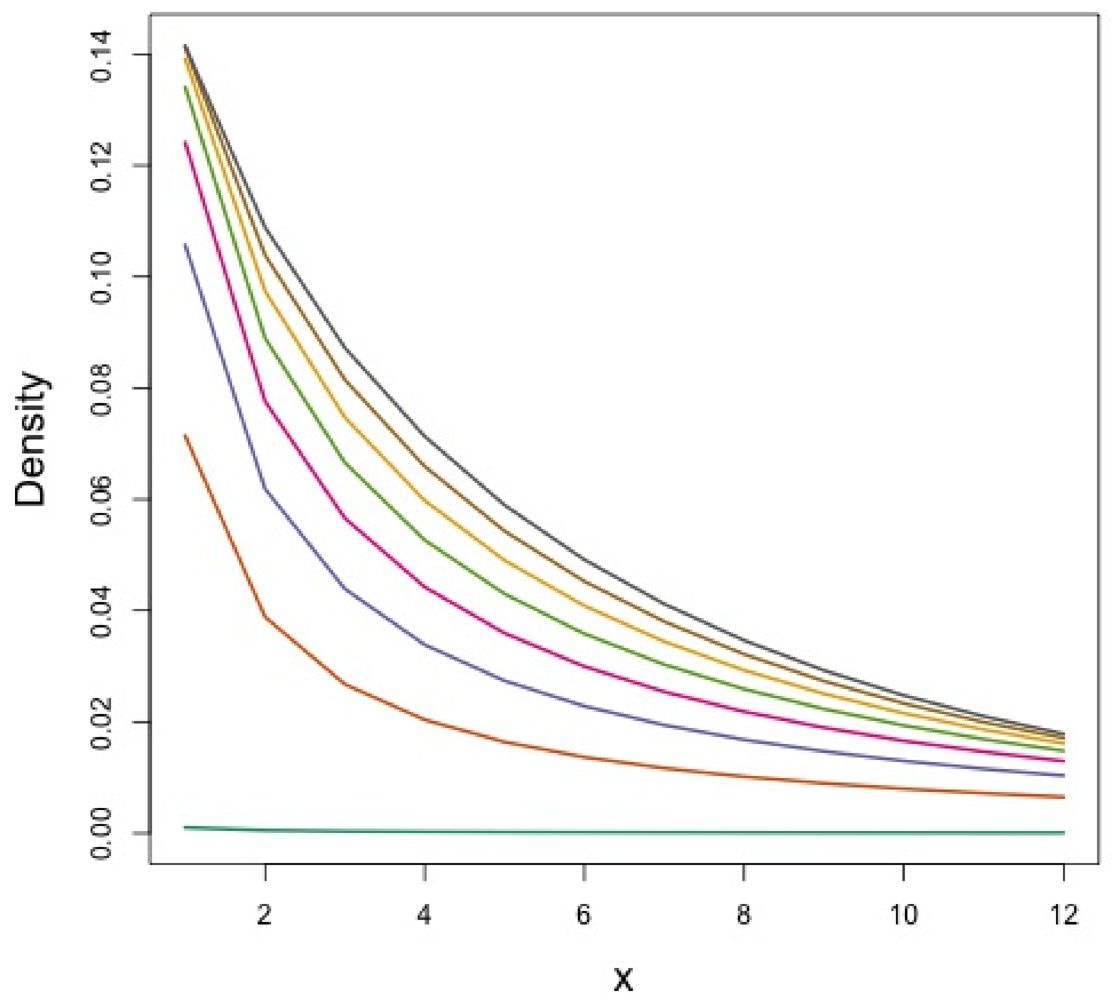
- Response is a count (not always integer)
- Often, it's mostly zero (that's complicated)
- Could also be catch per unit effort, biomass etc
- Flexible mean-variance relationship

Tweedie distribution



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

Negative binomial



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

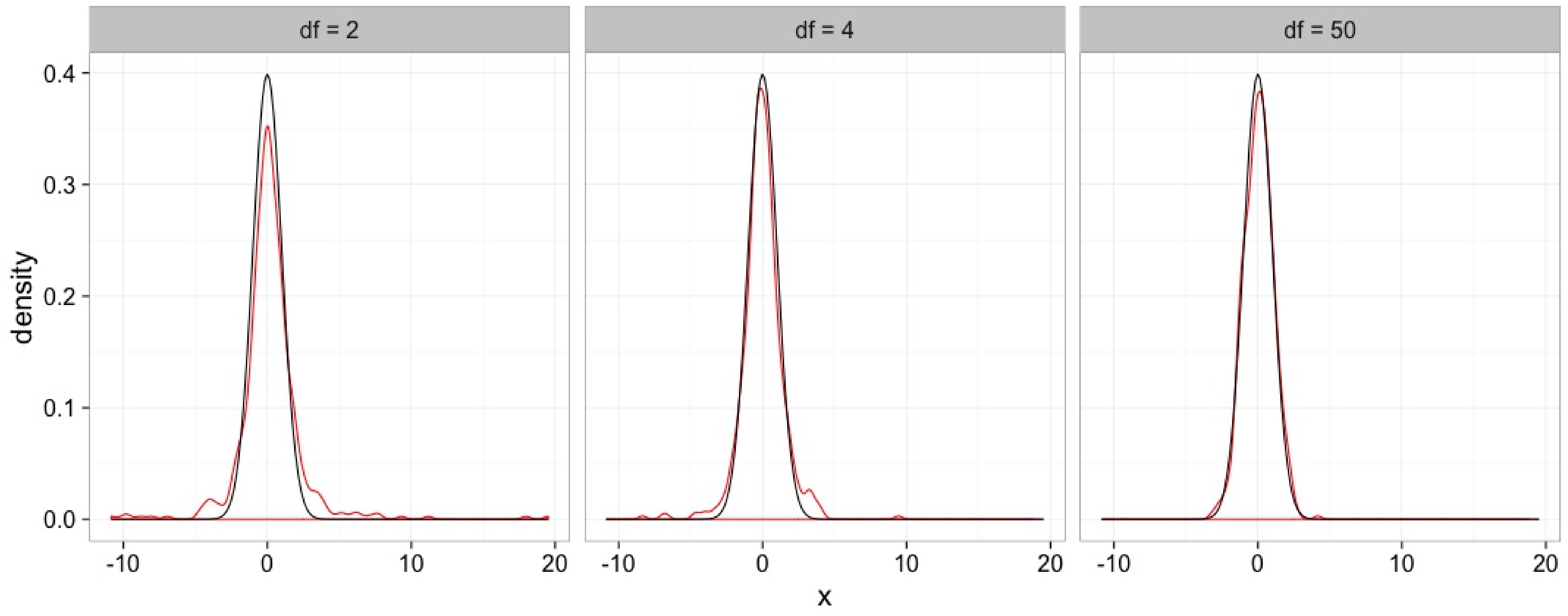
Modelling proportions

The Beta distribution

Modelling outliers

The student-t distribution

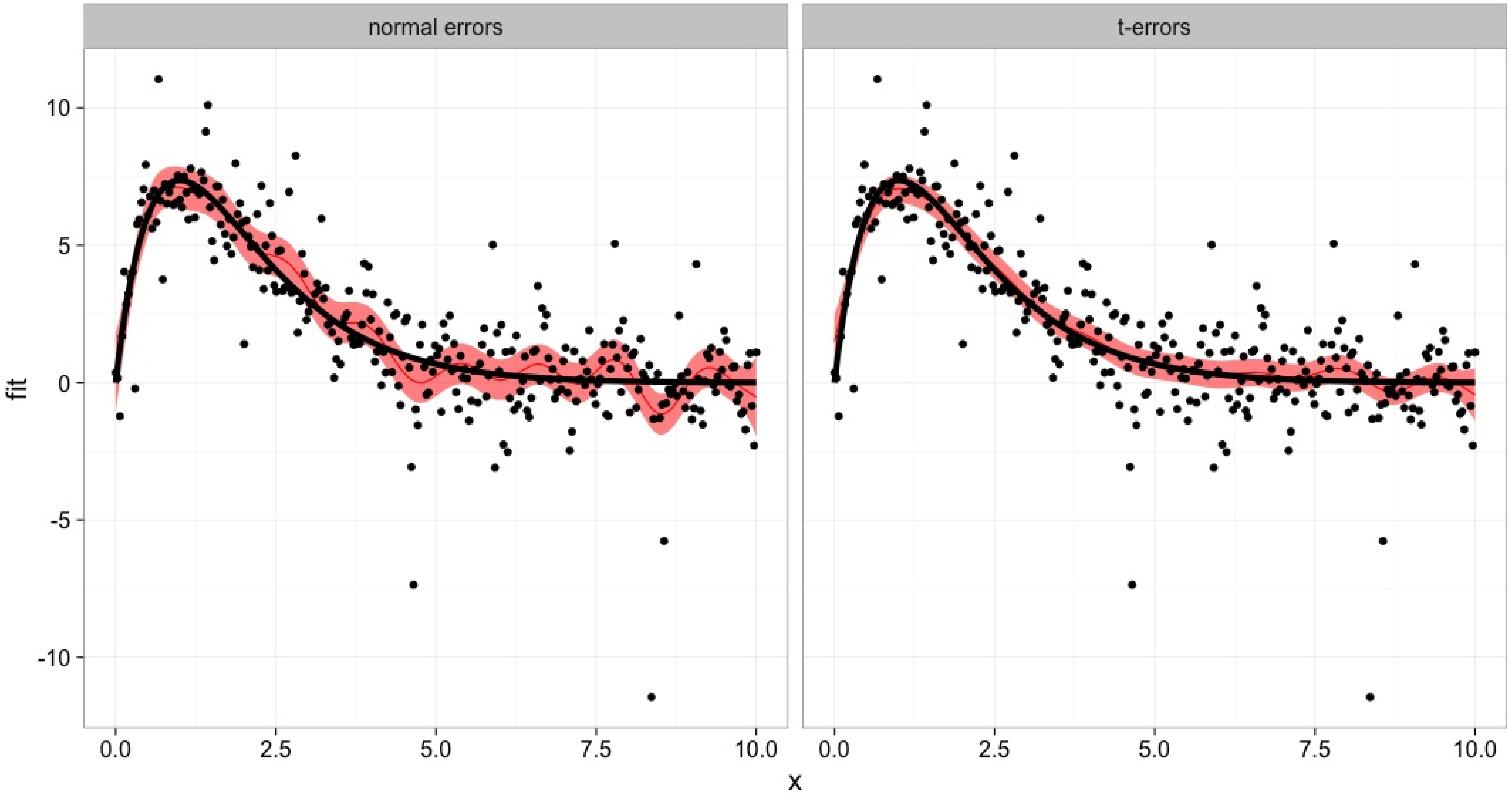
- Models continuous data w/ longer tails than normal
- Far less sensitive to outliers
- Has one extra parameter: df.
- bigger df: t dist approaches normal



The student-t distribution: Usage

```r
set.seed(4)
n=300
dat = data.frame(x=seq(0,10,length=n))
dat$f = 20*exp(-dat$x)*dat$x
dat$y  = 1*rt(n,df = 3) + dat$f
norm_mod =  gam(y~s(x,k=20), data=dat,
family=gaussian(link="identity"))
t_mod = gam(y~s(x,k=20), data=dat, family=scat(link="identity"))
```

The student-t distribution: Usage



The student-t distribution: Usage

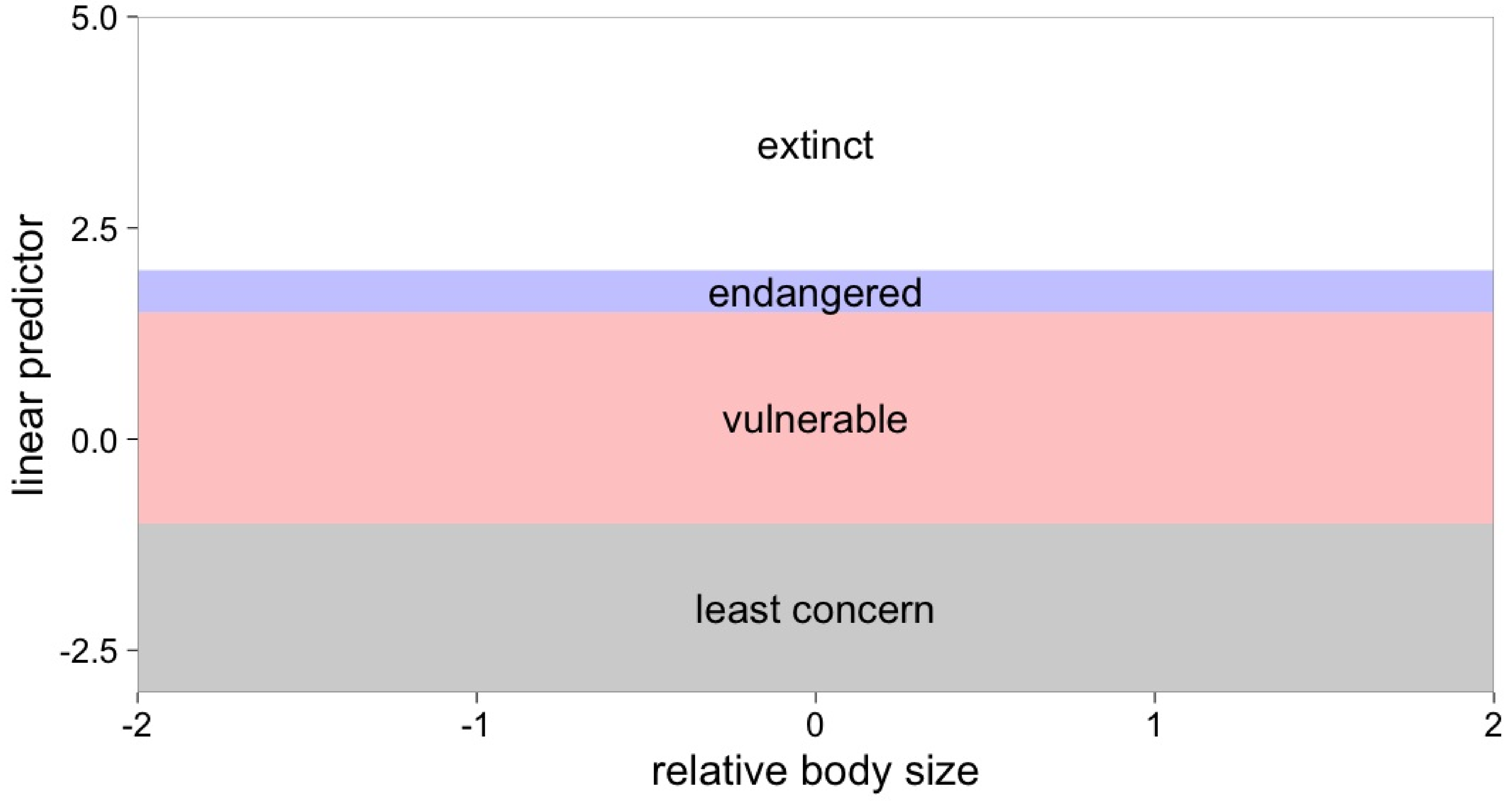
```
Family: Scaled t(2.976,0.968)
Link function: identity

Formula:
y ~ s(x, k = 20)

Parametric coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  2.02664    0.06853   29.57   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
       edf Ref.df Chi.sq p-value
s(x) 13.27  15.71   1221  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.695   Deviance explained = 63.1%
-REML = 546.75  Scale est. = 1         n = 300
```

Modelling multi-dimensional data

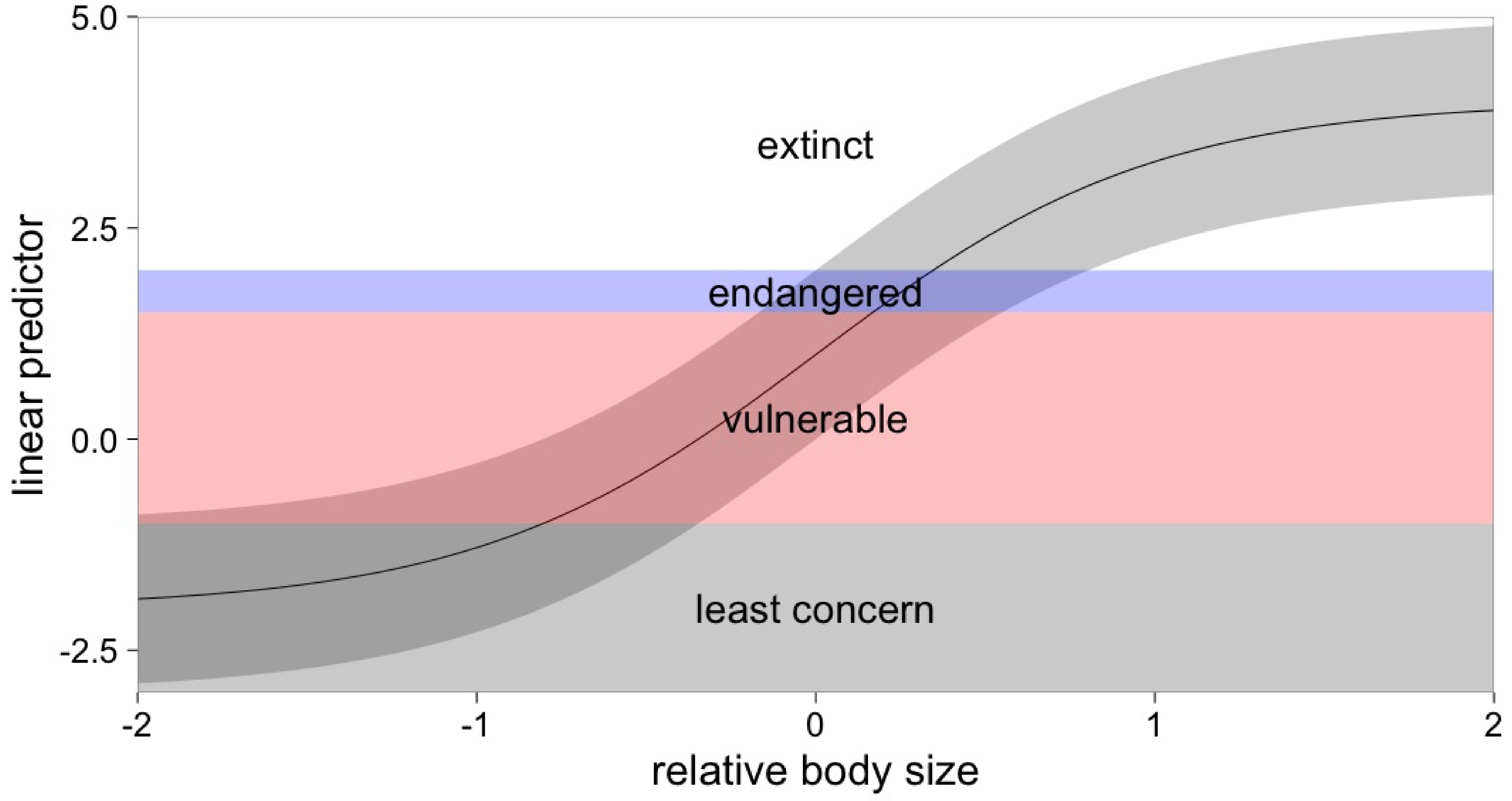
Ordered categorical data

- Assumes data are in discrete categories, and categories fall in order
- e.g.: conservation status: “least concern”, “vulnerable”, “endangered”, “extinct”
- fits a linear latent model using covariates, w/ threshold for each level
- First cut-off always occurs at -1

Ordered categorical data

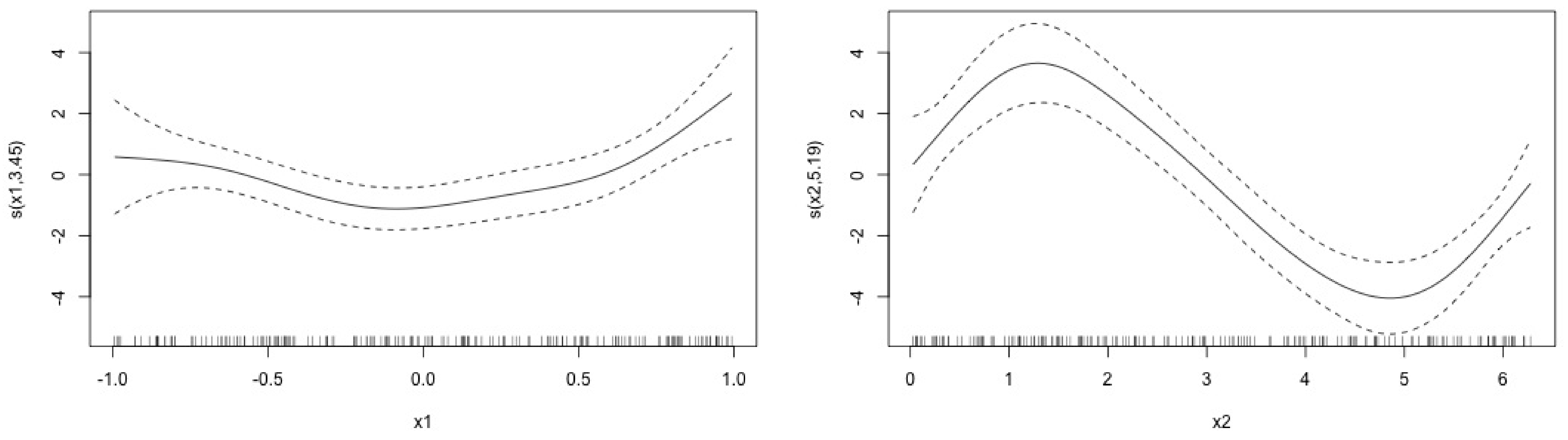


Ordered categorical data



Using ocat

```
n= 200
dat = data.frame(x1 = runif(n,-1,1),x2=2*pi*runif(n))
dat$f = dat$x1^2 + sin(dat$x2)
dat$y_latent = dat$f + rnorm(n,dat$f)
dat$y = ifelse(dat$y_latent<0,1, ifelse(dat$y_latent<0.5,2,3))
ocat_model = gam(y~s(x1)+s(x2), family=ocat(R=3),data=dat)
plot(ocat_model,page=1)
```

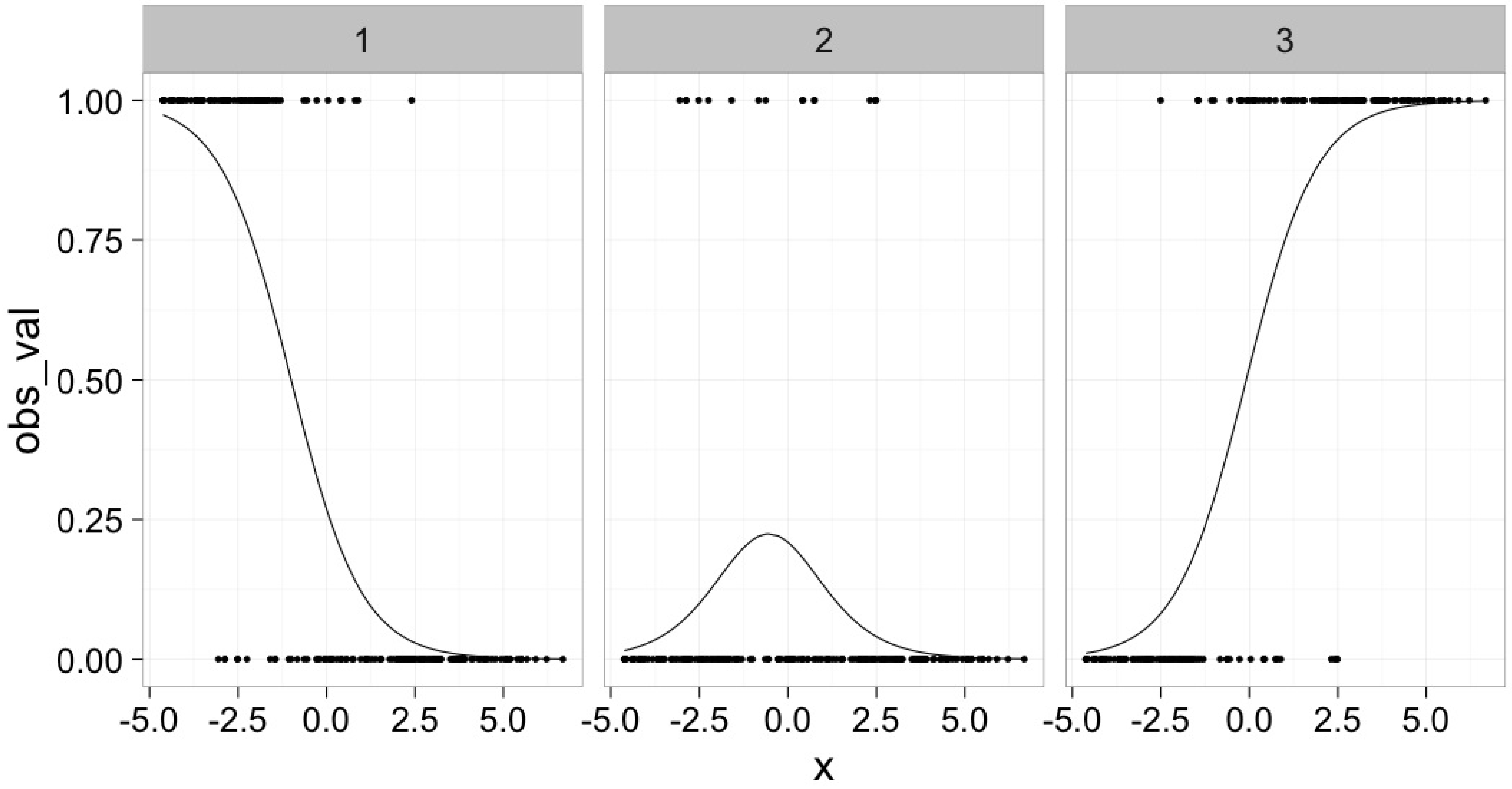


Using ocat

```
summary(ocat_model)
```

```

Family: Ordered Categorical(-1,-0.09)
Link function: identity

Formula:
y ~ s(x1) + s(x2)

Parametric coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.5010     0.2792   1.794   0.0727 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
        edf Ref.df Chi.sq  p-value
s(x1) 3.452  4.282  18.67  0.00133 **
s(x2) 5.195  6.270  84.34 1.09e-15 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Deviance explained = 57.7%
-REML =  97.38  Scale est. = 1         n = 200
```

Using ocat

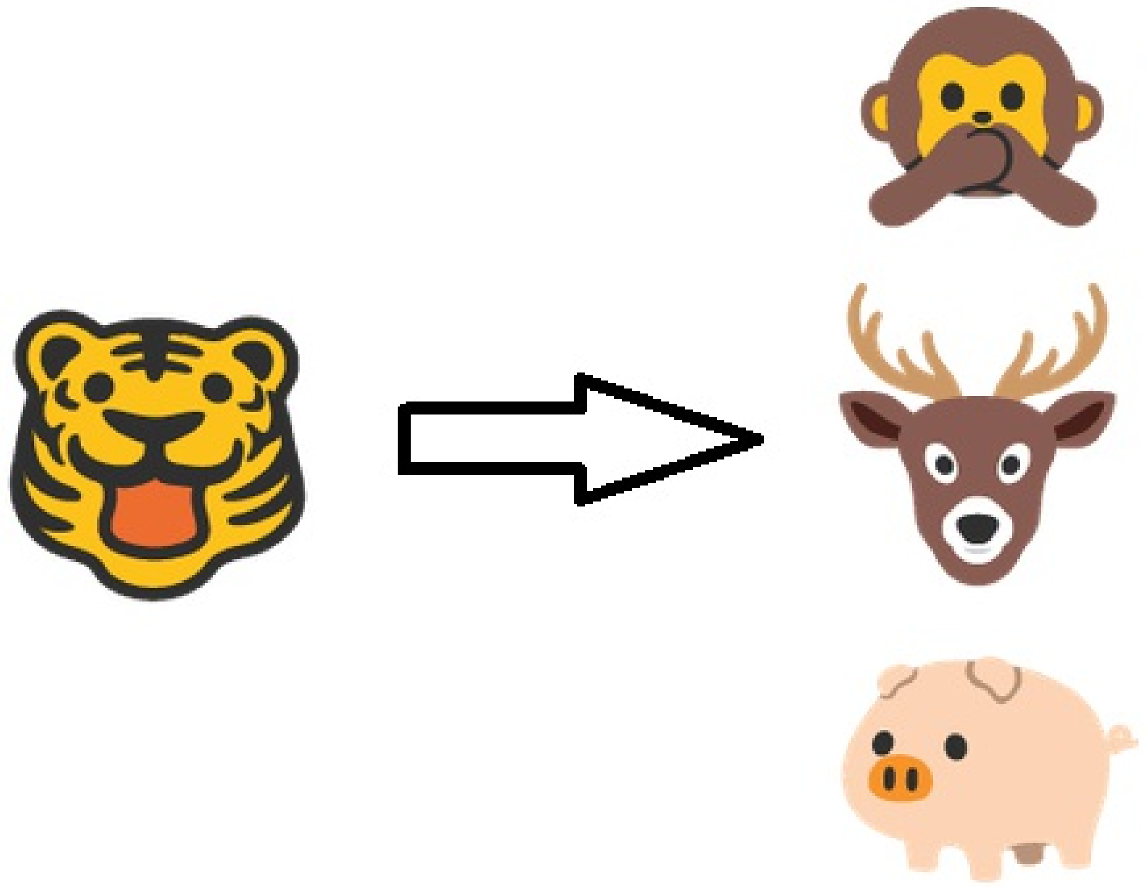


Unordered categorical data

- What do you do if categorical data doesn't fall in a nice order?

Unordered categorical data

- What do you do if categorical data doesn't fall in a nice order?



Unordered categorical data

Unordered categorical data

- Model probability of a category occuring relative to an (arbitrary) reference level

Unordered categorical data

- Model probability of a category occuring relative to an (arbitrary) reference level
- one linear equation for each category except the reference class

Unordered categorical data

- Model probability of a category occuring relative to an (arbitrary) reference level
- one linear equation for each category except the reference class
- $p(y = i|\mathbf{x}) = \exp(\mu_i(\mathbf{x}))/(1 + \sum_j \exp(\mu_j(\mathbf{x}))$

Unordered categorical data

- Model probability of a category occuring relative to an (arbitrary) reference level
- one linear equation for each category except the reference class
- $p(y = i|\mathbf{x}) = \exp(\mu_i(\mathbf{x}))/(1 + \sum_j \exp(\mu_j(\mathbf{x}))$
- $\mu_i(\mathbf{x}) = s_{1,j}(x_1) + s_{2,j}(x_2)$

Unordered categorical data

- Model probability of a category occuring relative to an (arbitrary) reference level
- one linear equation for each category except the reference class
- $p(y = i|\mathbf{x}) = \exp(\mu_i(\mathbf{x}))/(1 + \sum_j \exp(\mu_j(\mathbf{x}))$
- $\mu_i(\mathbf{x}) = s_{1,j}(x_1) + s_{2,j}(x_2)$
- $p(y = 0|\mathbf{x}) = 1/(1 + \sum_j \exp(\mu_j(\mathbf{x}))$

Using the multinom function

🙊 = 0

🦌 = 1

🐖 = 2

🦌 = f(tree cover, road distance)

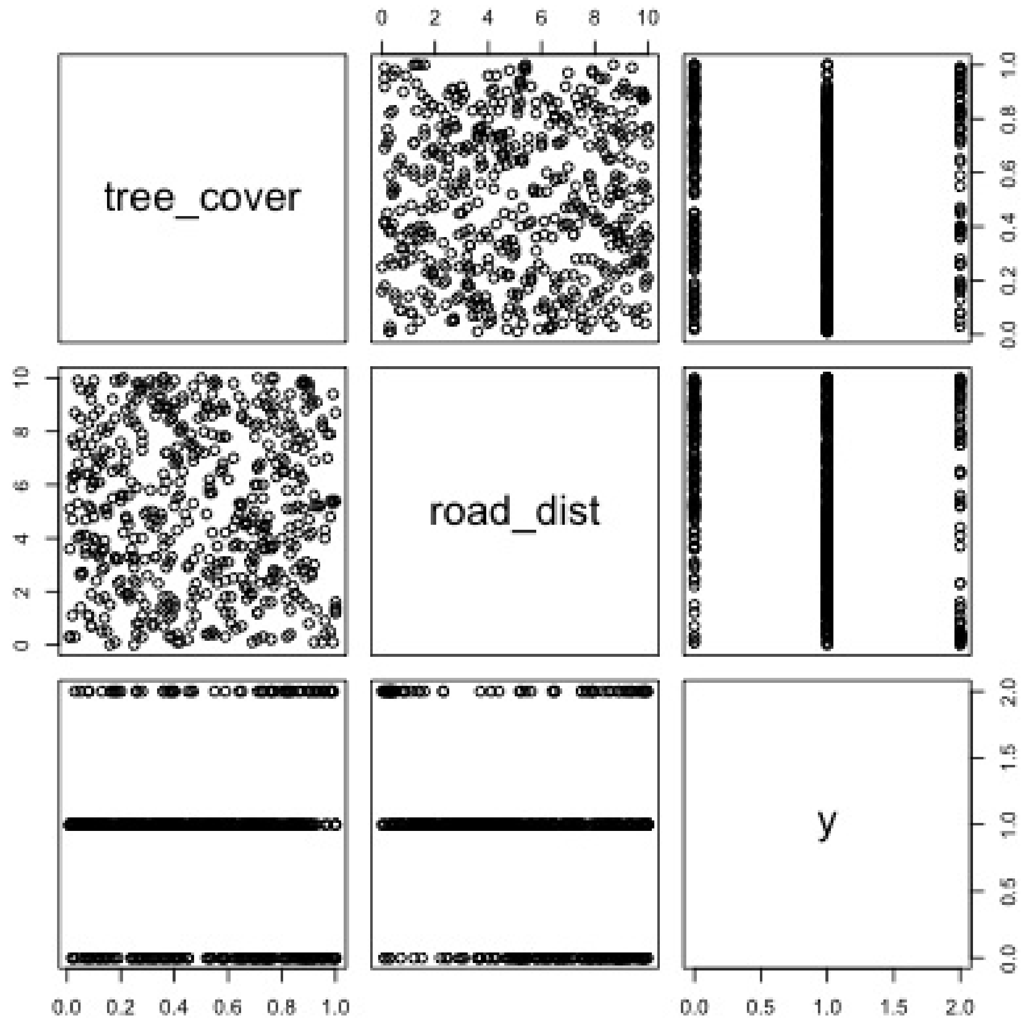
🐖 = g(tree cover, road distance)

Using the multinom function

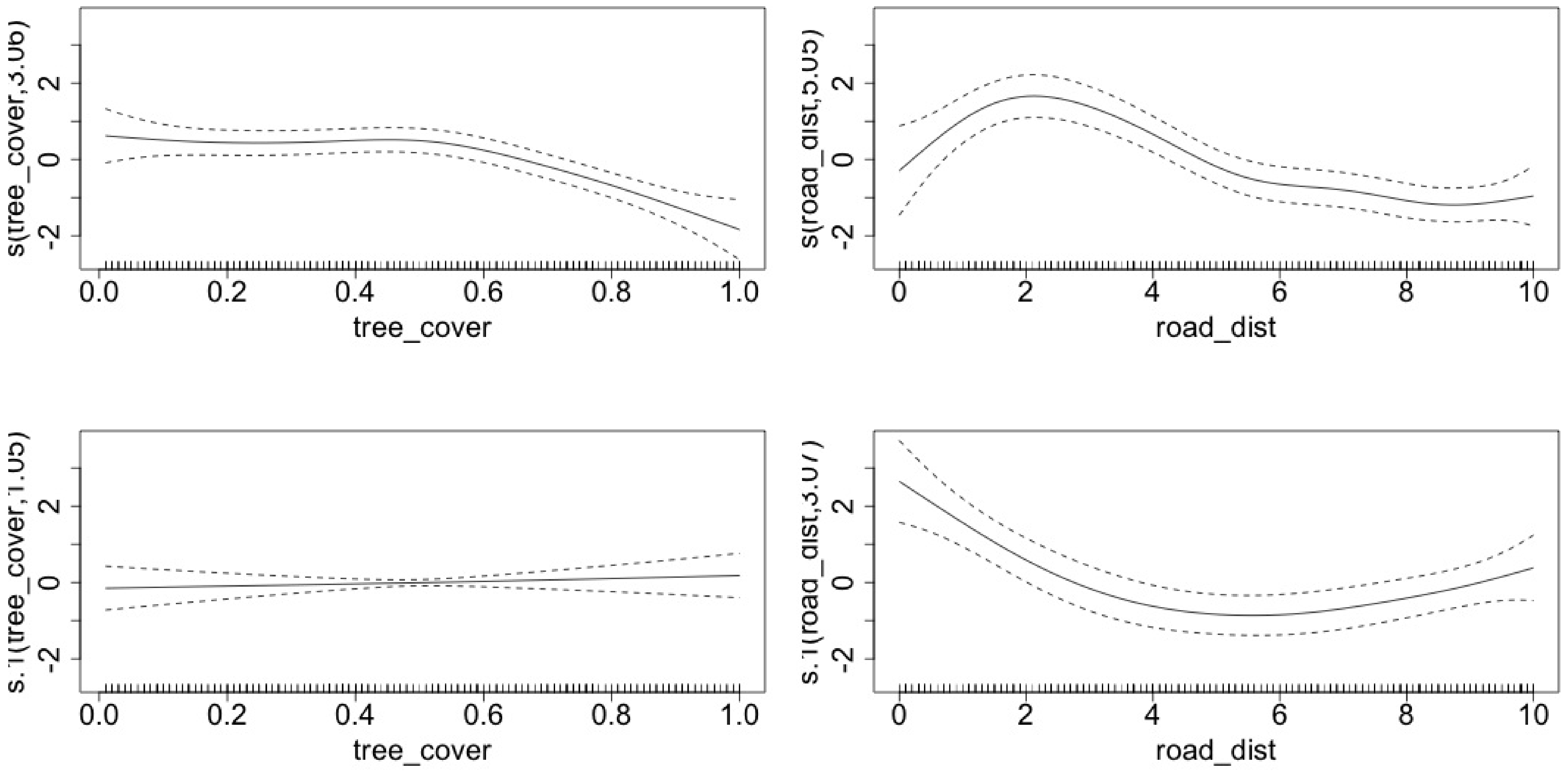
```
head(model_dat)
```

```
  tree_cover road_dist y
1       0.51       8.6 1
2       0.31       9.9 0
3       0.43       8.2 1
4       0.69       2.9 1
5       0.09       0.7 1
6       0.23       5.6 1
```

```
pairs(model_dat)
```

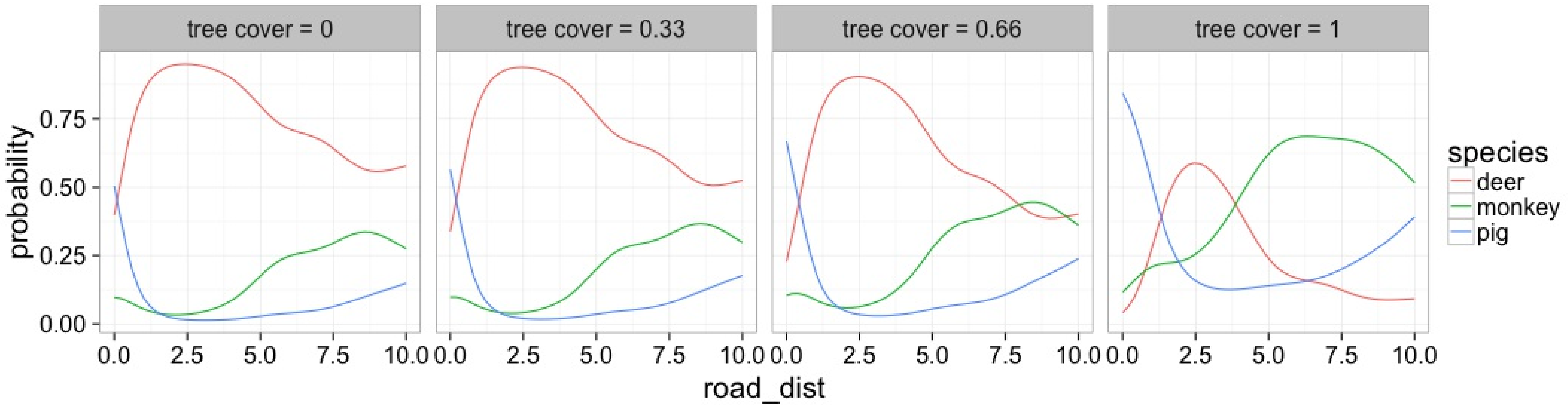


Using the multinom function



Understanding the results

```r
multinom_pred_data = as.data.frame(expand.grid(road_dist
=seq(0,10,length=50),
                                               tree_cover
=c(0,0.33,0.66,1)))
multinom_pred = predict(multinom_model, multinom_pred_data,type =
"response")
colnames(multinom_pred) = c("monkey","deer","pig")
multinom_pred_data = cbind(multinom_pred_data,multinom_pred)
multinom_pred_data_long = multinom_pred_data %>%
  gather(species, probability, monkey, deer,pig)%>%
  mutate(tree_cover =paste("tree cover = ", tree_cover,sep=""))
ggplot(aes(road_dist,
probability,color=species),data=multinom_pred_data_long)+
  geom_line()+
  facet_grid(.~tree_cover)+
  theme_bw(20)
```



Other multivariate distributions to check out

Multivariate normal (family = mvn)

Multivariate normal (family = mvn)

- Fit a different smooth model for multiple y-variables, but allowing correlation between y's

Multivariate normal (family = mvn)

- Fit a different smooth model for multiple y-variables, but allowing correlation between y's
- Example uses: multi-species distribution models, measuring latent correlations between environmental predictors

Multivariate normal (family = mvn)

- Fit a different smooth model for multiple y-variables, but allowing correlation between y's
- Example uses: multi-species distribution models, measuring latent correlations between environmental predictors
- mgcv code: formula=list(y1~s(x1)+s(x2), y2 = s(x1)+s(x3)), family = mvn(d=2)

Cox Proportional hazards (family = cox.ph)

Cox Proportional hazards (family = cox.ph)

- Censored data: y measures time until an event occurs, or the study was stopped (censoring)

Cox Proportional hazards (family = cox.ph)

- Censored data: y measures time until an event occurs, or the study was stopped (censoring)
- Measures relative rates, rather than absolute rates (no intercepts)

Cox Proportional hazards (family = cox.ph)

- Censored data: y measures time until an event occurs, or the study was stopped (censoring)
- Measures relative rates, rather than absolute rates (no intercepts)
- Example uses: time until an individual is infected, time until a subpopulation goes extinct, time until lake is invaded

Cox Proportional hazards (family = cox.ph)

- Censored data: y measures time until an event occurs, or the study was stopped (censoring)
- Measures relative rates, rather than absolute rates (no intercepts)
- Example uses: time until an individual is infected, time until a subpopulation goes extinct, time until lake is invaded
- mgcv code: `formula = y~s(x1)+s(x2), weights= censor.var,family=cox.ph`

Cox Proportional hazards (family = cox.ph)

- Censored data: y measures time until an event occurs, or the study was stopped (censoring)
- Measures relative rates, rather than absolute rates (no intercepts)
- Example uses: time until an individual is infected, time until a subpopulation goes extinct, time until lake is invaded
- mgcv code: `formula = y~s(x1)+s(x2), weights= censor.var,family=cox.ph`
- censor.var = 0 if censored, 1 if not

Gaussian location-scale models (family = gaulss)

Gaussian location-scale models (family = gaulss)

- Model both the mean (“location”) and variance (“scale”) as smooth functions of predictors

Gaussian location-scale models (family = gaulss)

- Model both the mean (“location”) and variance (“scale”) as smooth functions of predictors
- Example uses: detecting early warning signs in time series, finding factors driving population variability

Gaussian location-scale models (family = gaulss)

- Model both the mean ("location") and variance ("scale") as smooth functions of predictors
- Example uses: detecting early warning signs in time series, finding factors driving population variability
- mgcv code: `formula = list(y~s(x1)+s(x2), ~s(x2)+s(x3)), family=gaulss`

Gaussian location-scale models (family = gaulss)

- Model both the mean ("location") and variance ("scale") as smooth functions of predictors
- Example uses: detecting early warning signs in time series, finding factors driving population variability
- mgcv code: `formula = list(y~s(x1)+s(x2), ~s(x2)+s(x3)), family=gaulss`
- censor.var = 0 if censored, 1 if not

Zero-inflated Poisson location-scale models (family = ziplss)

Zero-inflated Poisson location-scale models (family = ziplss)

- Models the probability of zeros seperately from mean counts given that you've observed more than zero at a location.

Zero-inflated Poisson location-scale models (family = ziplss)

- Models the probability of zeros seperately from mean counts given that you've observed more than zero at a location.
- Example uses: Counts of prey caught when a predator might switch between not hunting at all (zeros) and active hunting

Zero-inflated Poisson location-scale models (family = ziplss)

- Models the probability of zeros seperately from mean counts given that you've observed more than zero at a location.
- Example uses: Counts of prey caught when a predator might switch between not hunting at all (zeros) and active hunting
- mgcv code: `formula = list(y~s(x1)+s(x2), ~s(x2)+s(x3)), family=ziplss`

The end of the distribution zoo

That's the end of this section! We convene after lunch (1:00 PM). You'll get to work through a few more advanced examples of your choice.