Additive Models and All That

© T. W. Yee

University of Auckland

2017-08-28 @ COMPASS

t.yee@auckland.ac.nz
https://www.stat.auckland.ac.nz/~yee

Opening Comments

1. Purpose is to give a framework for statistical regression modelling. In particular,
   1. VGLMs (*breadth*),
   2. VGAMs (*smoothing*),
   3. RR-VGLMs (*latent variables/dimension reduction*),
   4. Give ideas & techniques on how to perform regression.

   Emphasis is on the conceptual.
2. Try to see the forest, not the trees! The keyword is *infrastructure (for regression)*, or *framework (for regression)*.
3. Lots of ground will be covered. . . but try not to get information overload. . . . Lots of technical details have been removed.
4. Feel free to ask quick questions.
5. Reference book: Yee (2015).

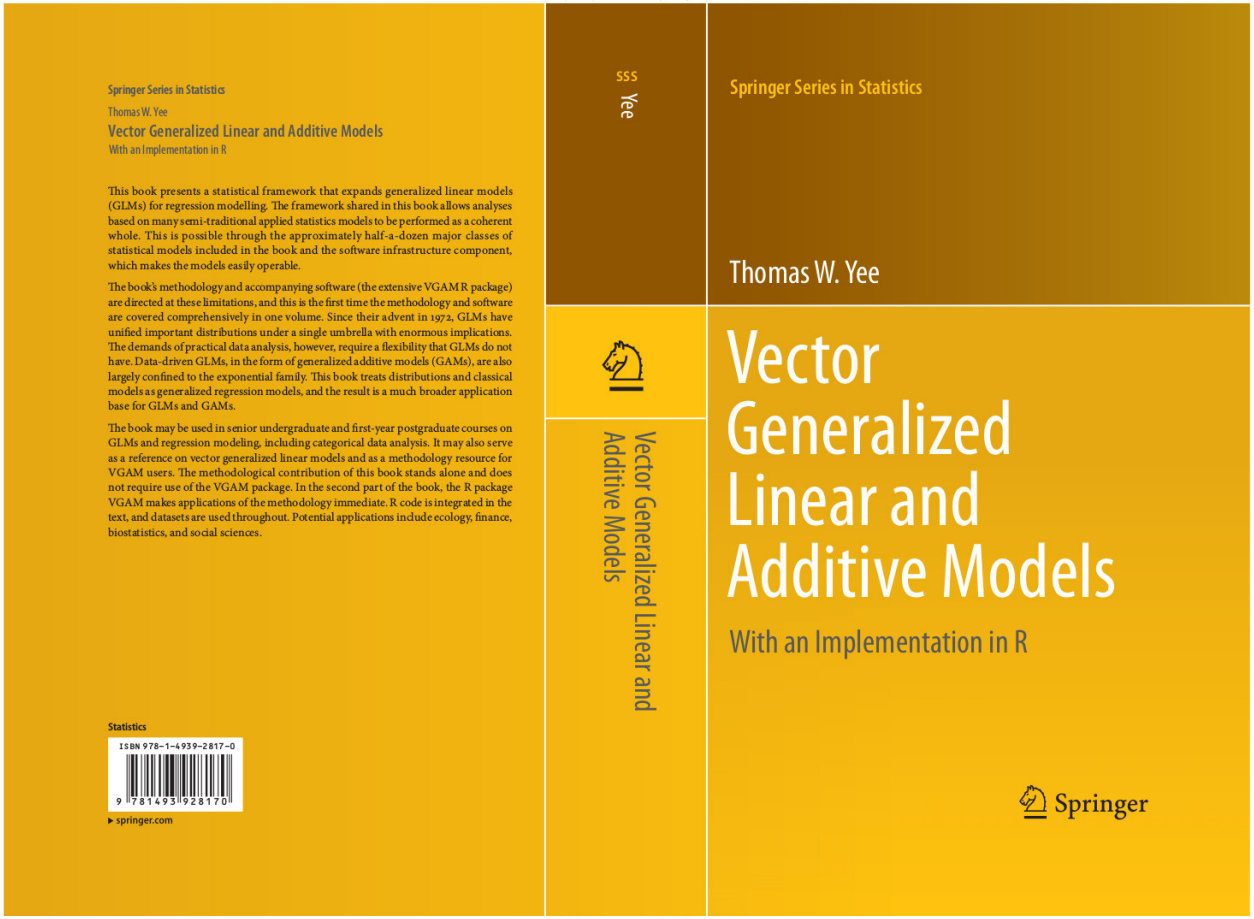


Figure : Called "VGLAM" here.

Outline

What is Regression?

Regression: we have a vector of responses $\mathbf{y}_i$ modelled by a vector of explanatory variables $\mathbf{x}_i$, for n independent individuals.

The main purposes are:

- interpretation (to help understand what the data is saying),
- inference (to know how sure to be),
- prediction.

Regression is used much in this world...

Modern Regression

Since the era of modern computing, a plethora of 'modern' methods have became available, e.g.,

- smoothing, generalized additive models (GAMs), vector GAMs (VGAMs), ...,
- neural networks (NNs), classification and regression trees, projection pursuit regression (PPR), multivariate adaptive splines (MARS), smoothing-spline ANOVA (SS-ANOVA), ...,
- wavelets, k-nearest neighbours (KNN), ensemble methods, $p \gg n$ methods, self-organizing maps (SOMs), ...,
- support vector machines (SVMs), independent component analysis (ICA), ...,
- least angle regression (LARS), least absolute shrinkage and selection operator (LASSO), ...
- bootstrapping, boosting, bagging (bootstrap aggregating), random forests, ...

Why R?

- Free!
- Fully featured (e.g., 11,000+ in mid-2017)
- Runs on many hardware and software platforms.
- Award winning (ACM software prize to John Chambers).
- Is very powerful, has superb graphics.
- Started[1] at the Statistics Department at the University of Auckland.
- **R** tends to be the first to implement any new statistical methodology.
- Incidentally, `iascars2017.com` gives a few details about a December 2017 conference at Auckland to mark the retirement of Ross Ihaka.

Its command- or language driven!

Figure : NZSA/IASC-ARS Conference in December.

R Pre-Conference Workshops

There are four half-day workshops scheduled for Sunday December 10. Two each will run in parallel during the morning and afternoon. These are:

1. "Faster R code" by Thomas Lumley,
2. "Getting to Know Grid Graphics" by Paul Murrell,
3. "Analysing spatial point patterns using spatstat" by Rolf Turner,
4. "Graphics in R" by Chris Wild.

They are open to conference non-attendees.

Recommended!

[1] by Ross Ihaka and Robert Gentleman.

S Model Formulas

A typical call might look something like

```
lm(y ~ x2 + x3 + f1*x3 + f2/f3, data = my.frame)
```

The S model formula adopted from Wilkinson and Rogers (1973).

Form: `response` $\sim$ `expression`

LHS = the response (usually a vector in a data frame or a matrix).

RHS = explanatory variables.

Consider

```
y ~ -1 + x1 + x2 + x3 + f1:f2 + f1*x1 + f2/f3 +
     f3:f4:f5 + (f6 + f7)^2
```

where variables beginning with an `x` are numeric and those beginning with an `f` are factors.

By default an intercept is fitted, which is `1`. Suppress intercepts by `-1`.

The *interaction* `f1*f2` is expanded to `1 + f1 + f2 + f1:f2`. The terms `f1` and `f2` are *main effects*.

A second-order interaction between two factors can be expressed using `factor:factor`: γ_{ij}. There are other types of interactions. Interactions between a factor and numeric, `factor:numeric`, produce $\beta_j x$. Interactions between two numerics, `numeric:numeric`, produce a cross-product term such as $\beta\, x_2\, x_3$.

`(f6 + f7)^2` expands to `f6 + f7 + f6:f7`.

`(f6 + f7 + f8)^2 - f7:f8` expands to all main effects and all second-order interactions except for `f7:f8`.

Nesting is achieved by `/`, e.g., `f2/f3` is shorthand for `1 + f2 + f3:f2`, or equivalently,

```
1 + f2 + f3 %in% f2
```

Example: `f2` = state and `f3` = county.

There are times when you need to use the *identity function* `I()`, e.g., because "^" has special meaning,

```
lm(y ~ -1 + offset(a) + x1 + I(x2 - 1) + I(x3^3), data = ldata)
```

fits

$$\begin{aligned} y_i &= a_i + \beta_1 x_{i1} + \beta_2 (x_{i2} - 1) + \beta_3 x_{i3}^3 + \varepsilon_i, \\ \varepsilon_i &\sim \text{iid } N(0, \sigma^2), \quad i = 1, \ldots, n, \end{aligned}$$

where `a` is a vector containing the (known) a_i.

Other functions: `factor()`, `as.factor()`, `ordered()` `terms()`, `levels()`, `options()`.

Table : S formula operators.

t

Operator/function	Comment
`+`	Addition of a term
`1`	Intercept (present by default)
`-`	Omit the following term, e.g., `-1` suppresses an intercept
`.`	All variables in a data frame except for the response
`0`	No intercept (alternative method)
`:`	Interaction (tensor product) between two terms
`*`	Interaction (expansion), e.g., `A * B` = `A + B + A:B`
`/`	Nesting, same as `%in%`, e.g., `A / B` = `A + B:A`
`^`	Higher-order 'expansion', e.g., `(A + B)^2` = `A + B + A:B`
$\sim$	"is modelled as a function of", defines a S formula
`offset()`	offset, a vector or matrix of fixed and known values, e.g., `offset(log.time)`
`I()`	Identity or insulate, allows standard arithmetic operations to have their usual meaning, e.g., `I((x2 - x3)^2)` for the variable $(x_2 - x_3)^2$

Table : Logical operators and some commonly used arguments in modelling functions such as `glm()` and `vglm()`.

t

Operator	Comment
`&`	Vector operator: and
`|`	Vector operator: or
`!`	Vector operator: not

Argument	Comment
`contrasts`	Handling of factor contrasts. e.g., `contrasts = c("contr.sum", "contr.poly")`
`na.action`	Handling of missing values. e.g., `na.action = na.pass`
`offset`	Offset, an alternative to `offset()` in the `formula` argument, e.g., `offset = log(followup.time)`
`subset`	Subset selection, e.g., `subset = 20 < age & sex == "M"`
`weight`	Prior weights, known and fixed

S Generic Functions

Generic functions are available for `lm` objects. They include

- `add1()`
- `anova()`
- `coef()`, $\widehat{\boldsymbol{\beta}}$
- `deviance()`, $\sum r_i^2$
- `df.residual()`, $n - p$
- `drop1()`
- `model.matrix()`, $\mathbf{X}$
- `plot()`
- `predict()`, $\widehat{\mu}_i$
- `print()`
- `residuals()`, $r_i = y_i - \widehat{\mu}_i$
- `step()`
- `summary()`, $\widehat{\boldsymbol{\beta}}$, $\widehat{\mathrm{Var}}(\widehat{\boldsymbol{\beta}})$, ...
- `update()`.

Other less used generic functions are `alias()`, `effects()`, `family()`, `kappa()`, `labels()`, `proj()`.

The lm() Function

```
> args(lm)

function (formula, data, subset, weights, na.action, method = "qr",
    model = TRUE, x = FALSE, y = FALSE, qr = TRUE, singular.ok = TRUE,
    contrasts = NULL, offset, ...)
NULL
```

The most useful arguments are

- weights,
- subset, e.g., subset = age < 30 & country == "italy",
- na.action—na.fail(), na.omit(), na.exclude(), na.pass(),
- contrasts.

Data frames: read.table(), write.table(), na.omit(), transform(), with(), subset().

The Penalty Function Approach

In mathematical modelling it is common to estimate the parameters by balancing 2 opposing quantities. One can do this, in general, by solving

$$\min_{\boldsymbol{\theta}} \; A + \lambda B \tag{1}$$

where $\boldsymbol{\theta}(\lambda)$ is the vector of parameters to be estimated and λ (≥ 0) is the *balancing* or *trade-off parameter*.

The smaller quantity A is, the closer the fit is with the data (this tends to *overfit*). Simply minimizing A would result in an extreme fit that would not generalize well for future data. But if we add a quantity B to the objective function that increases as A decreases then we can regularize the fit. If A is too large then this tends to *underfit*.

As $\lambda \to 0^+$ the fit will become complicated because B becomes negligible. As $\lambda \to \infty$ the fit becomes simpler because λB is forced to remain small

relative to A, i.e., the penalty B is forced to decrease quicker relative to the increase in A.

In the subject of statistics, the penalty approach (1) is adopted commonly. Here are some examples.

AIC, *BIC* The *Akaike information criterion* and *Bayesian information criterion* are commonly used to compare models, e.g.,

$$\text{AIC} = -2\ell + 2p. \tag{2}$$

They balance goodness of fit by the number of parameters. These information criteria have known λ and are traditionally used on multiple models that are not nested for the purpose of model selection.

Smoothing splines

The quantity A is a residual sum of squares, and B measures the wiggliness of the smoother. There are techniques such as cross-validation which are used to try and obtain a reasonable value for λ for a given data set.

LASSO

This method estimates the β_k of a LM by minimizing

$$\sum_{i=1}^{n} \left(y_i - \beta_1 - \sum_{k=2}^{p} x_{ik} \beta_k \right)^2 + \lambda \sum_{k=2}^{p} |\beta_k|, \tag{3}$$

and called the 'least absolute shrinkage and selection operator' (*LASSO*). With increasing λ, the shrinking is such that $\beta_k = 0$ for values of k belong to some set of variables, and thus x_k is no longer selected in the regression. For λ sufficiently large, all the coefficients become 0 (except the intercept term which is unpenalized). This can be seen in

the following figure where the paths of the LASSO coefficients based on an LM fitted to the `azpro` data frame from COUNT are traced. The first plot has λ on a log-scale as its x-axis, and the second plot has the l_1 norm of $(\beta_2, \ldots, \beta_p)^T$. Package glmnet is used in the figure.

Trees In the topic of classification and regression trees, a popular algorithm for choosing a tree of reasonable size is to contrast the number of leaves (the penalty term B) with some measure of impurity, such as the Gini index or deviance.

P-splines Penalized splines are similar to smoothing splines and have gained widespread use.

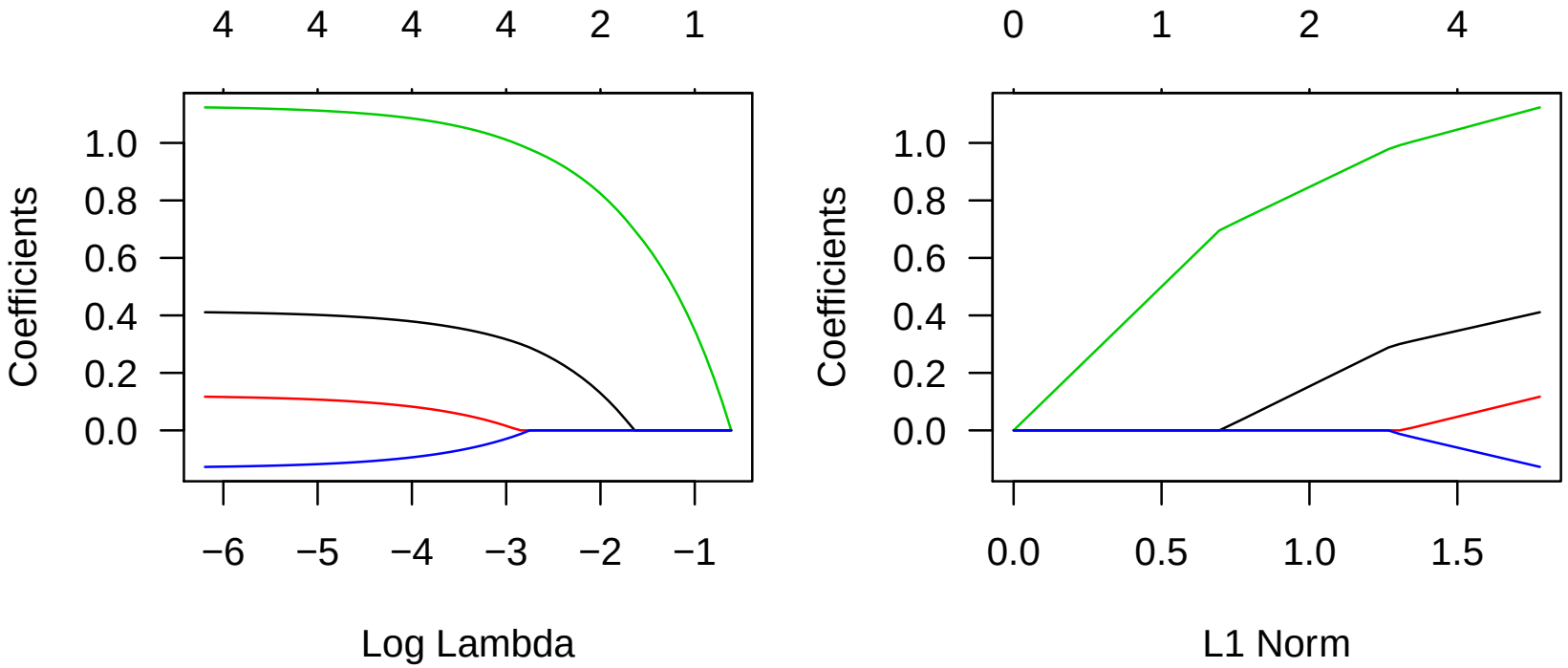


Figure : Paths of the estimated LASSO coefficients in a LM. The response `log(los)` is regressed against `admit` (black), `age75` (red), `procedure` (green) and `sex` (blue) in `azpro`. The LHS has log λ as its x-axis; the RHS has $\sum_{k=2}^{p} |\beta_k|$ in (3). The upper numbers are the number of variables in the model.

Smoothing

A powerful tool for exploratory data analysis. Allows a *data-driven* approach rather than *model-driven* approach. Allows the data to "speak for itself".

The central idea is *localness*, i.e., local behaviour versus global behaviour of a function.

Scatterplot data (x_i, y_i), $i = 1, \ldots, n$.

The *classical smoothing problem* is

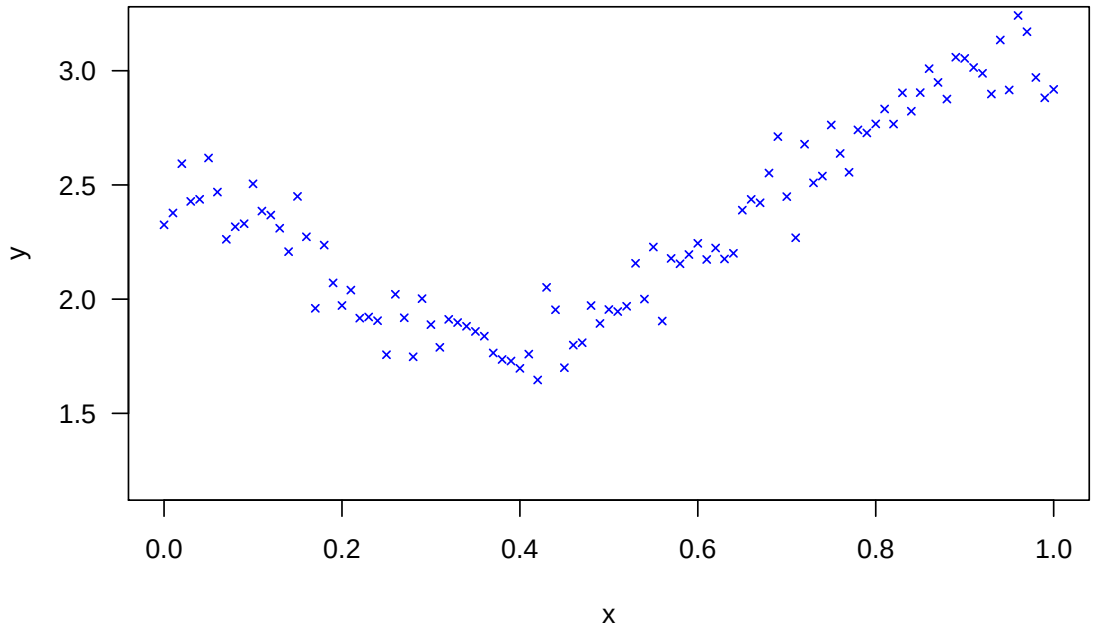
$$y_i = f(x_i) + \varepsilon_i, \quad \varepsilon_i \sim (0, \sigma_i) \tag{4}$$

independently. Here, f is an arbitary smooth function, and $i = 1, \ldots, n$.

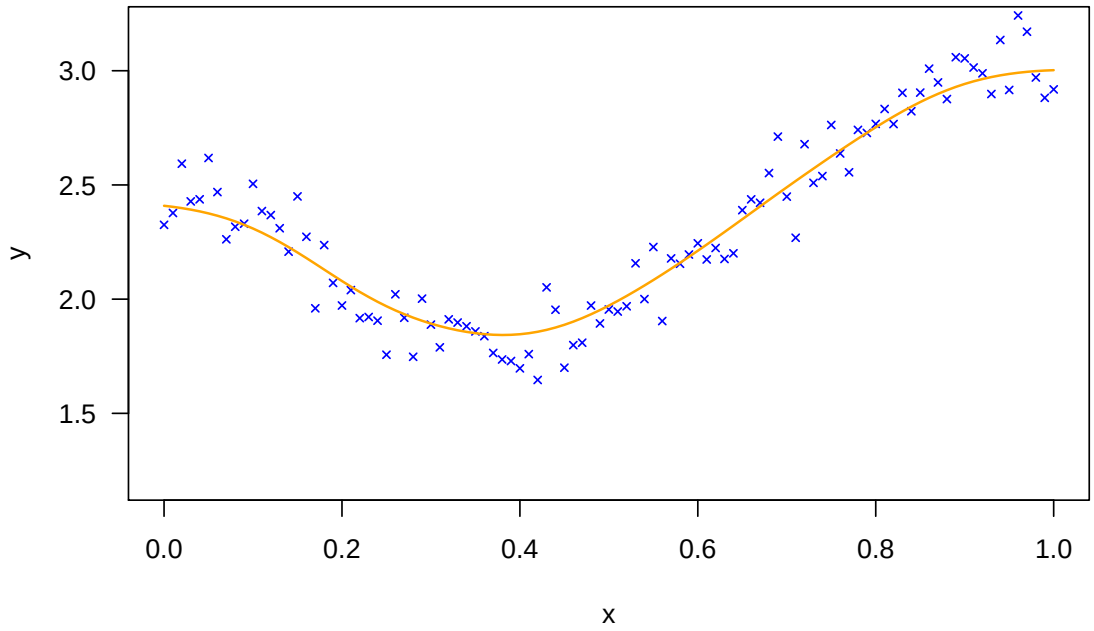
Q: How can f be estimated?

A: If there is no *a priori* function form for f, one solution is the smoother.

Example



Example



Uses of Smoothing

Smoothing has many uses, e.g.,

- data visualization and EDA
- prediction
- derivative estimation, e.g., growth curves, acceleration
- used as a basis for many modern statistical techniques

Three Classes of Smoothers

Of the many types of smoothers, there are 2 or 3 common methods:

1. Regression smoothers (polynomials, regression splines), e.g., `ns()` and `bs()`. These are the most important.
2. Kernel smoothers (N-W, locally weighted averages, local regression, loess).
3. Smoothing splines (roughness penalties), `smooth.spline()`.

Polynomial Regression

This is a common technique that involves fitting polynomial functions of each x_k. It provides more flexibility than the usual linear $\beta_k x_k$ term. It's easy too, e.g.,

```
myquadraticfit <- lm(y ~ poly(x2, 2),              data = pdata)
myquadraticfit <- lm(y ~ poly(x2, 2, raw = TRUE), data = pdata)
myquadraticfit <- lm(y ~ I(x2^2),                  data = pdata)
```

to fit a 2nd degree polynomial (*quadratic* or *parabola*). The default uses *orthogonal polynomials* which are numerically stable, but have coefficients that are not so interpretable. Setting the argument `raw = TRUE` creates terms such as `I(x2^2)`.

Myth: fitting a sufficiently high order polynomial will be a good idea for estimating most fs in general [cf. Stone-Weierstrass Theorem].

But this is not the case because polynomials...

- are *global functions*, not *local functions*. e.g., individual observations can have a large influence on remote parts of the curve.
- have *edge effects*. They often do not model the boundaries well, especially if the degree of the polynomial is high. This results in significant bias in regions of the x-space, e.g., dangerous for prediction.
- are sensitive to outliers and high-leverage points.
- polynomial degree cannot be controlled continuously.

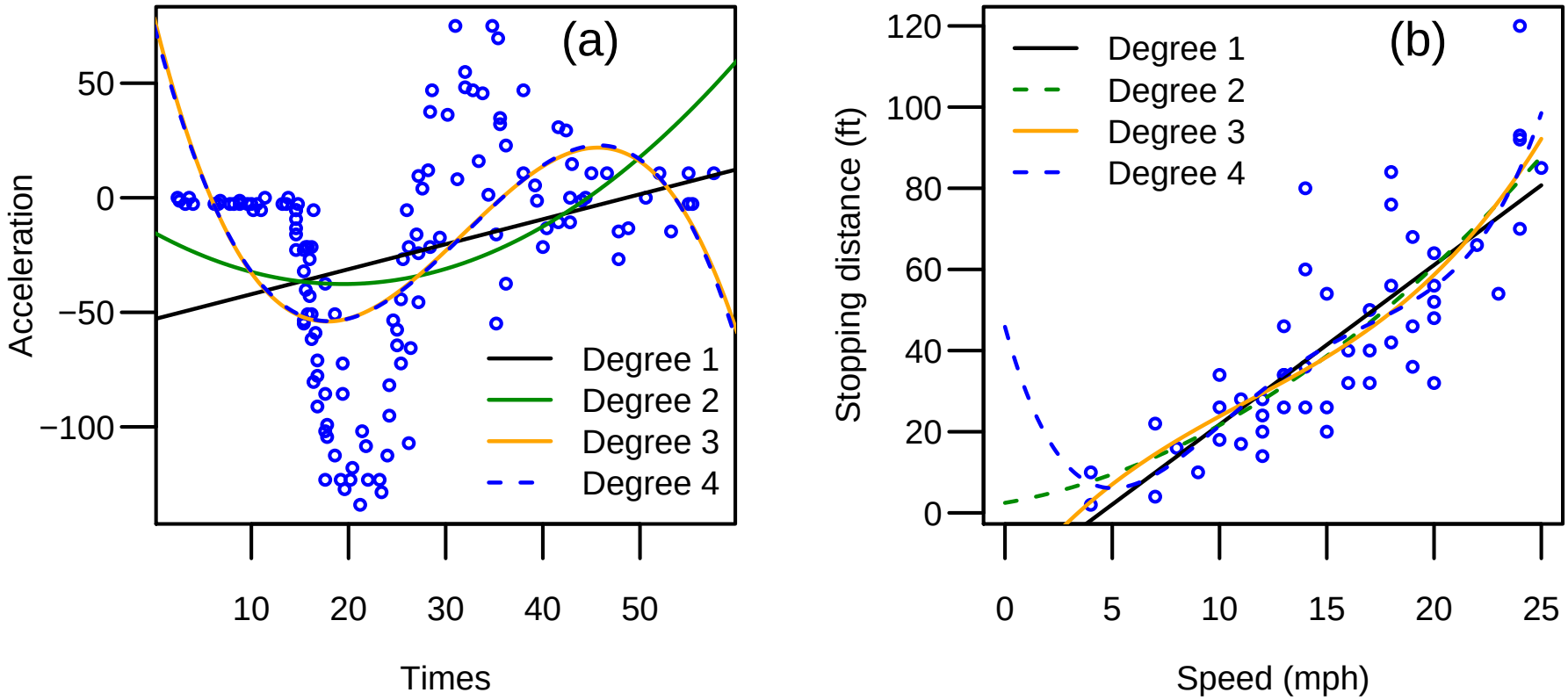


Figure : Polynomials of degree 1–4 fitted to two data sets. (a) `mcycles` from MASS. (b) `cars` from datasets.

Conclusions

These are probably safe general recommendations:

- avoid fitting 4th degree polynomials (quartics) or higher;
- even fitting a 3rd degree polynomial (cubic) should be done very cautiously and with trepidation;
- a better solution is to use regression splines, e.g., `bs()` and `ns()`.

In fact, if you fit a cubic or higher then you probably should be shot!

Regression Splines

So high degree polynomials are bad—because they are global functions. What's good?

Answer (better): *Regression splines* use a *piecewise polynomial of usually low degree*, e.g., 1 or 2 or 3. The regions are separated by *knots* ξ_j (or *breakpoints*). The positions where each pair of *segments* join are called *joints*. The more knots, the more flexible the family of curves become.

It is customary to force the piecewise polynomials to join smoothly at these knots. A popular choice are piecewise cubic polynomials with continuous 0th, 1st and 2nd derivatives called *cubic splines*. Using splines of degree > 3 seldom yields any advantage.

Given a set of knots, the smooth is computed by multiple regression on a set of *basis functions*.

Definition: A function $f \in \mathcal{C}^k[a, b]$ if derivatives $f', f'', \ldots, f^{(k)}$ all exist and are continuous in $[a, b]$, e.g., $|x| \notin \mathcal{C}^1[a, b]$.
Notes:

1. $f \in \mathcal{C}^k[a, b] \Longrightarrow f \in \mathcal{C}^{k-1}[a, b]$.
2. $\mathcal{C}[a, b] \equiv \mathcal{C}^0[a, b] = \{f(t) : f(t)$ continuous and real valued, $a \leq t \leq b\}$.

There are at least two bases for cubic splines:

1. truncated power series: TPSs are easier to understand but is unused in practice because they are numerically unstable.
2. B-splines: harder to understand but is used in practice because they are numerically stable.

Here's a regression spline.

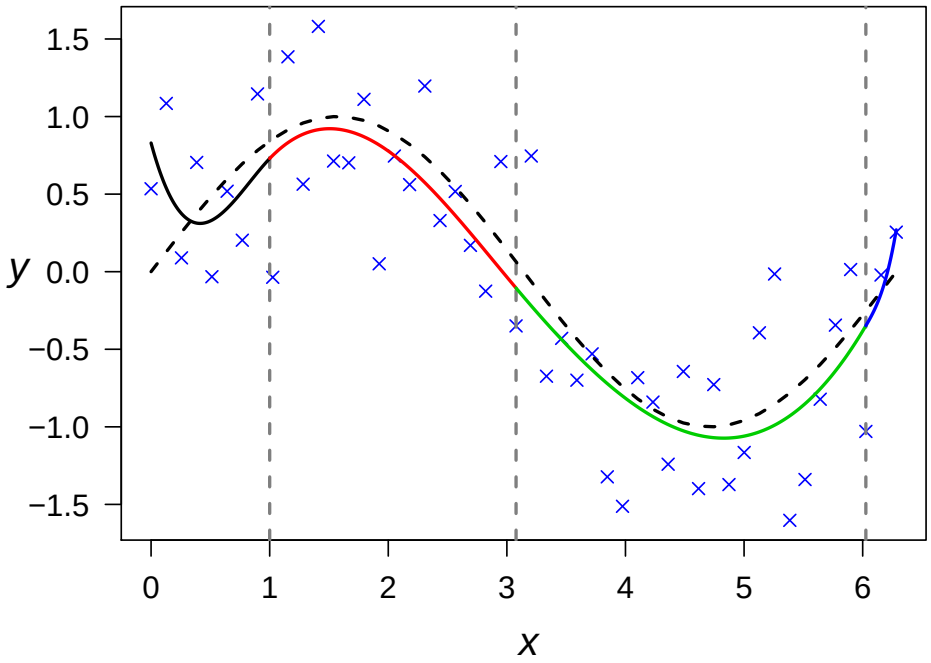


Figure : Smoothing some data with a regression spline (B-spline). Each segment of the spline is coloured differently. The term is effectively `bs(x, knots = c(1, 3.08, 6.03))`. The true function is the sine function (dashed) and $n = 50$.

B-Splines†

Unlike the TPS basis, B-splines form a numerically stable basis for splines. It is convenient to consider splines of a general order, M say.

- $M = 4$: *cubic spline*.
- $M = 3$: *quadratic spline* which has continuous derivatives up to order $M - 2 = 1$ at the knots—this is aka a *parabolic spline*.
- $M = 2$: *linear spline* which has continuous derivatives up to order $M - 2 = 0$ at the knots—i.e., the function is continuous.

In practice, use something like

```
fit1 <- vglm(y ~ bs(x2, df = 4) + ns(x3, df = 4),
             VGAMfamilyfunction, data = vdata)
plot(as(fit1, "vgam"), se = TRUE)
```

```
> library("splines")  # bs() and ns() here
> args(bs)

function (x, df = NULL, knots = NULL, degree = 3, intercept = FALSE,
    Boundary.knots = range(x))
NULL

> args(ns)

function (x, df = NULL, knots = NULL, intercept = FALSE, Boundary.knots = range(x))
NULL
```

Note that `ns()` gives *cubic* splines only, and are linear beyond the boundaries. Also, `order = degree + 1`.
In fact, for `bs()`, `df` should be `length(knots) + degree + intercept`.

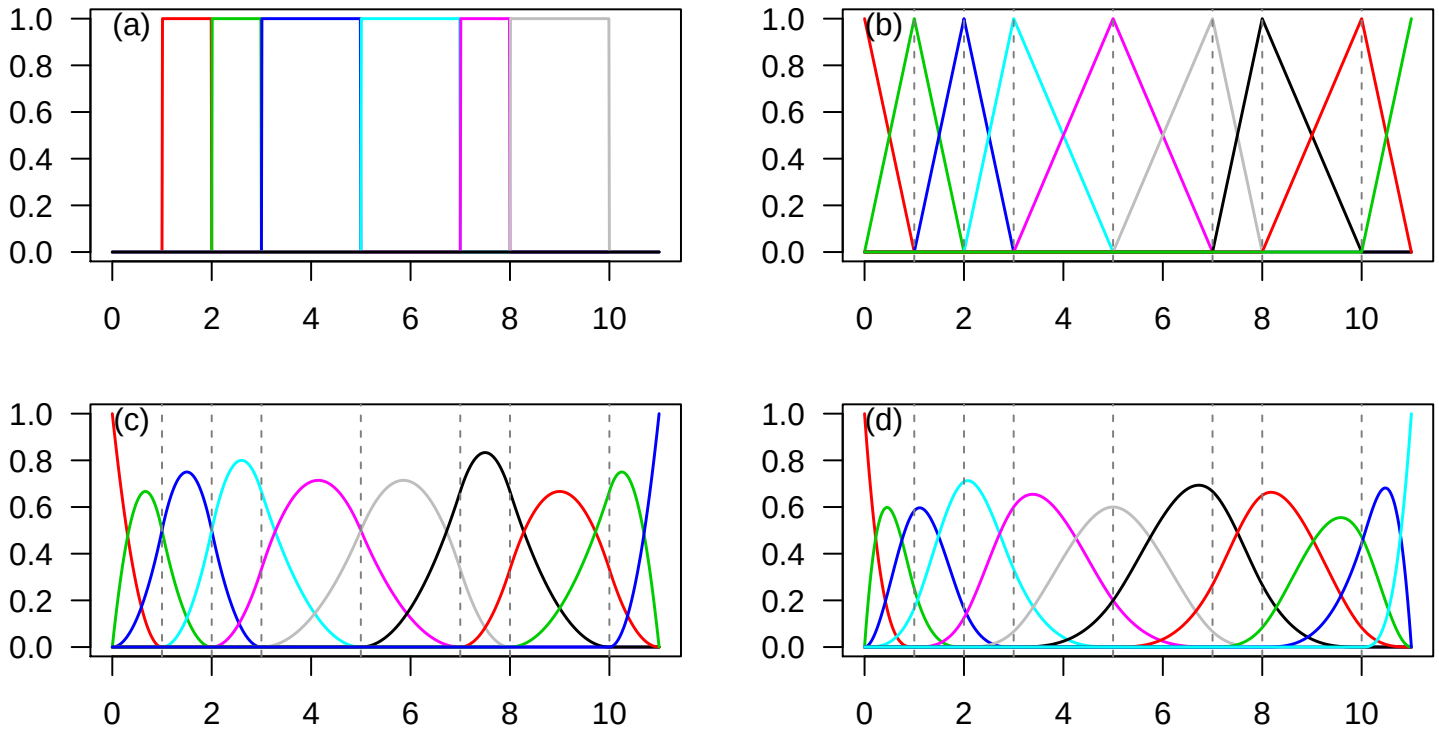


Figure : B-splines of order 1–4 ((a)–(d)) where the interior knots are denoted by vertical lines. The basis functions have been plotted left to right.

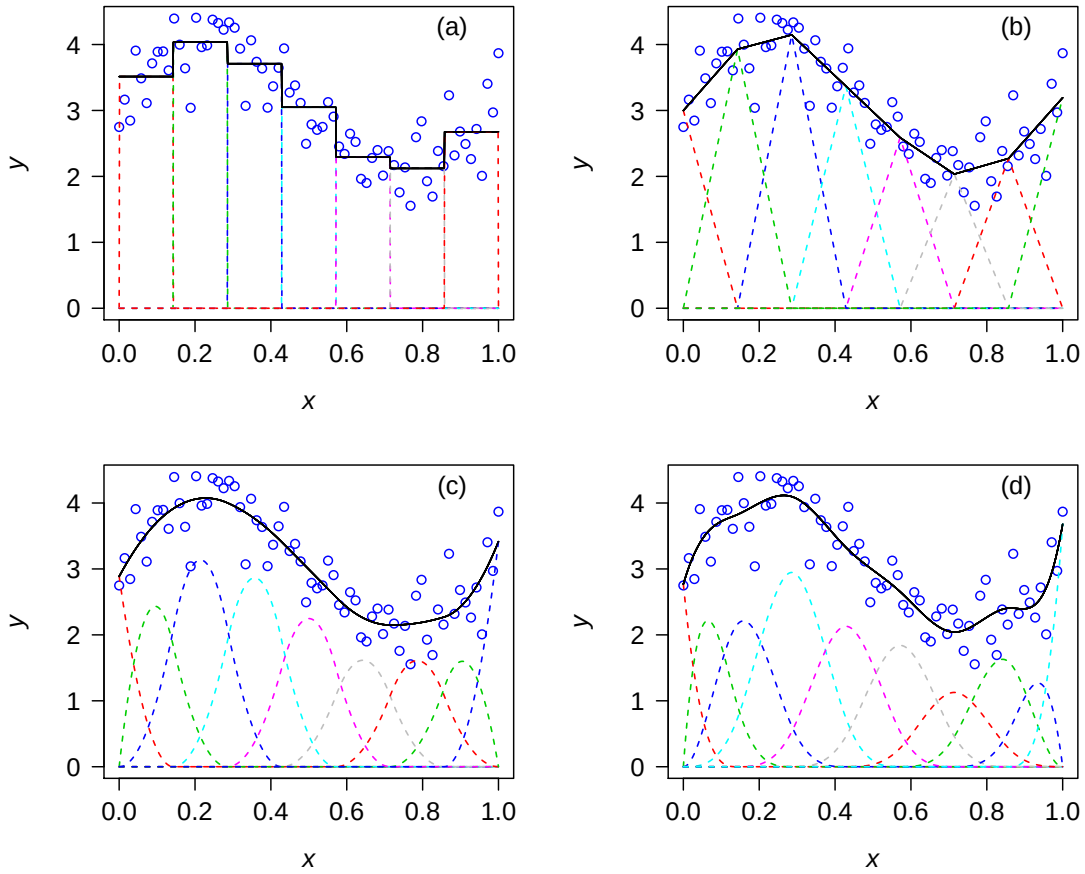


Figure : (a)–(d) Linear combinations of B-splines of degrees 0–3 fitted to some scatter plot data. The knots are equally-spaced on the unit interval.

Advantages of regression splines:

- computationally and statistically simple,
- standard parametric inferences are available, e.g., testing whether a knot can be removed and the same polynomial equation used to explain two adjacent segments can be tested by $H_0 : \theta_j = 0$, which is one of the t-tests statistics always printed by a regression program.
- The effects package can plot additive models for `lm()` and `glm()` objects, hence no need for a specialized GAM-fitting package.

Disadvantages of regression splines:

- difficult to choose the number of knots,
- difficult to choose the position of the knots,
- the smoothness of the estimate cannot be varied continuously as a function of a single smoothing parameter,
- often don't handle the boundaries well.

Generalized Linear Models (GLMs)

Introduction

Response $Y \sim$ *Exponential family* (normal, binomial, Poisson, ...)

$$g(\mu) \;=\; \eta(\mathbf{x}) \;=\; \boldsymbol{\beta}^T \mathbf{x} \;=\; \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_p x_p \tag{5}$$

Here, $x_1 \equiv 1$ if there is an intercept.

Random and systematic components.
g is the *link function* (known, monotonic, twice differentiable).

$\eta = \sum_{k=1}^{p} \beta_k x_k$ is known as the *linear predictor*.

Most commonly, $g =$ identity, logit and log links. But there are others....

The glm() Function

```
> args(glm)

function (formula, family = gaussian, data, weights, subset,
    na.action, start = NULL, etastart, mustart, offset, control = list(...),
    model = TRUE, method = "glm.fit", x = FALSE, y = TRUE, contrasts = NULL,
    ...)
NULL
```

Use, e.g., `glm(y ~ x2 + x3 + x4, family = binomial, bdata)`

- Family functions: about 6.
- Generic functions include `anova()`, `coef()`, `fitted()`, `plot()`, `predict()`, `print()`, `resid()`, `summary()`, `update()`.

The glm() Function

```
> args(glm)

function (formula, family = gaussian, data, weights, subset,
    na.action, start = NULL, etastart, mustart, offset, control = list(...),
    model = TRUE, method = "glm.fit", x = FALSE, y = TRUE, contrasts = NULL,
    ...)
NULL
```

Use, e.g., `glm(y ~ x2 + x3 + x4, family = binomial, bdata)`

- Family functions: about 6.
- Generic functions include `anova()`, `coef()`, `fitted()`, `plot()`, `predict()`, `print()`, `resid()`, `summary()`, `update()`.

Use, e.g., `vglm(y ~ x2 + x3 + x4, family = binomialff, bdata)`

- Family functions: 150+.
- Generic functions include `anova()`, `coef()`, `fitted()`, `plot()`, `predict()`, `print()`, `resid()`, `summary()`, `update()`.

Logistic Regression Example

```
data(chinese.nz, package = "VGAMdata")
mypch <- c(solidcirc = 16, solidbox = 15); konst1 <- 0.012

plot(female / (male + female) ~ year, data = chinese.nz,
     ylab = "Proportion female", cex = konst1 * sqrt(male + female),
     pch = mypch[1], col = "blue", las = 1, ylim = c(0, 0.55))
abline(h = 0.5, lty = "dashed", col = "gray50")

fit1.cnz <- glm(cbind(female, male) ~ year,          binomial, data = chinese.nz)
fit2.cnz <- glm(cbind(female, male) ~ poly(year, 2), binomial, data = chinese.nz)
fit4.cnz <- glm(cbind(female, male) ~ bs(year, 4),   binomial, data = chinese.nz)

mylty <- c(1, 2, 1); mycol <- c("green", "purple", "orange")
lines(fitted(fit1.cnz) ~ year, chinese.nz, col = mycol[1], lty = mylty[1])
lines(fitted(fit2.cnz) ~ year, chinese.nz, col = mycol[2], lty = mylty[2])
lines(fitted(fit4.cnz) ~ year, chinese.nz, col = mycol[3], lty = mylty[3])
legend("bottomright", col = mycol, lty = mylty,
       legend = c("linear", "quadratic", "B-spline"))
```

```
> head(chinese.nz)

  year male female     nz
1 1867 1213      6 217416
2 1871 2637      4 254948
3 1874 4814      2 344985
4 1878 4424      9 458007
5 1881 4995      9 534030
6 1886 4527     15 620451

> tail(chinese.nz)

   year  male female      nz
22 1976  8081   6779 3129384
23 1981  8649   8004 3143307
24 1986  9903   9603 3263283
25 1991 18750  18939 3373926
26 1996 39624  42696 3618303
27 2001 50460  55020 3737277
```

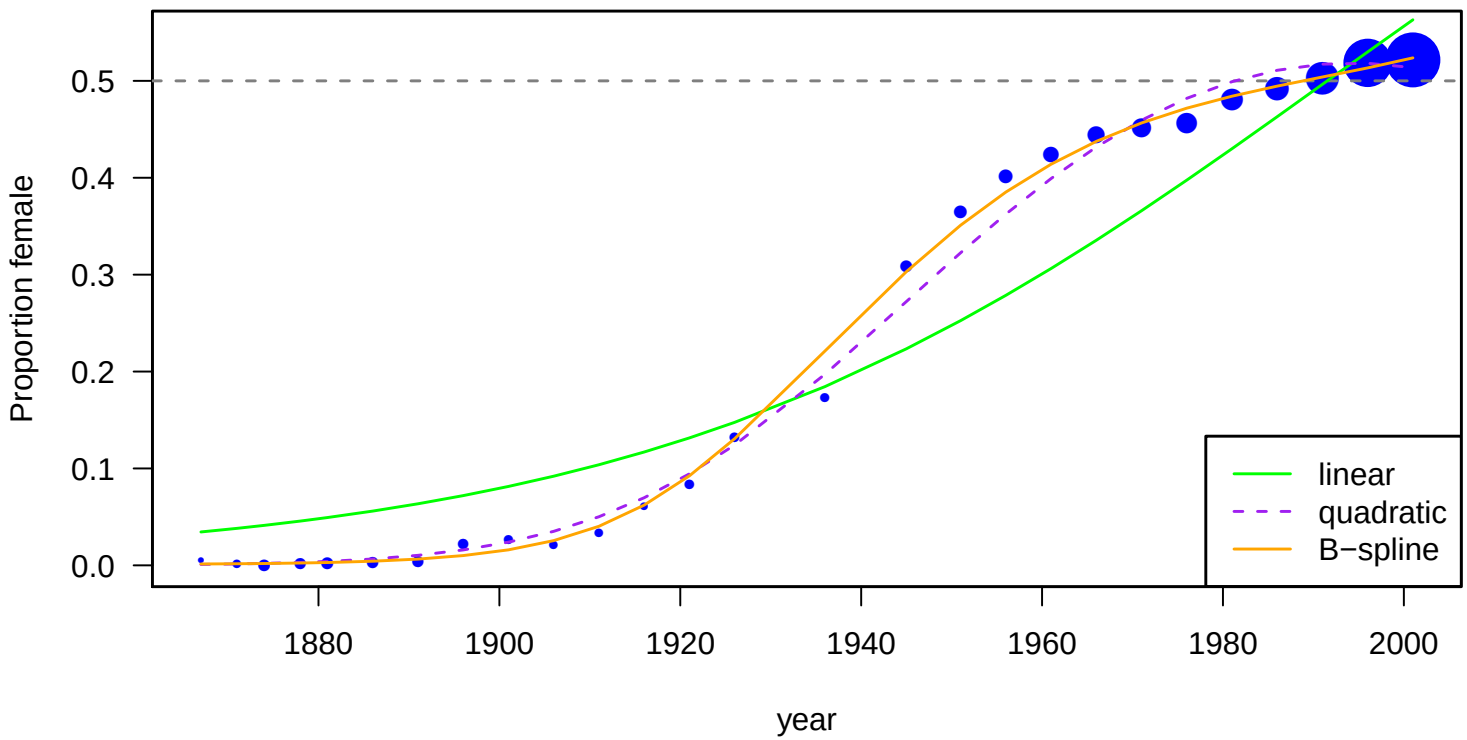


Figure : Some logistic regression models fitted to `chinese.nz`. The terms are `year`, `poly(year, 2)`, `bs(year, 4)`. Area sizes of the points $\propto$ number of people. *Note the bias–variance tradeoff.*

Multinomial Logit Example

Look at `xs.nz`, a large cross-sectional study of a workforce company about 20 years ago...

$$\log \frac{\Pr(Y = j|\mathbf{x})}{\Pr(Y = \text{married}|\mathbf{x})} = \eta_j(\mathbf{x}), \tag{6}$$

$j = \text{single}, \text{separated/divorced}, \text{widowed}.$

With smoothing ...

```
> males <- subset(xs.nz, sex == "M")[, c("marital", "age")]  # For simplicity
> ooo <- with(males, order(age))  # Sort rows wrt 'age'
> males <- na.omit(males[ooo, ])
> head(males, 3)  # Look at some data

     marital age
7974  single  16
315   single  17
774   single  17

> tail(males, 3)  # Look at some data

     marital age
9121 married  86
6513 married  88
7435 married  88

> with(males, table(marital))

marital
  single  married divorced  widowed
    1075     5880      435      128
```

```
> mfit <- vgam(marital ~ sm.os(age), multinomial(refLevel = "married"),
               data = males)
> plot(fitted(mfit)[, 1] ~ age, males, type = "n", ylim = 0:1,
       ylab = "Fitted probability", las = 1,
       main = "Males in xs.nz; marital status")
> matlines(with(males, age), fitted(mfit), col = 1:4, lty = 1:4, lwd = 2)
> legend("topright", col = 1:4, lty = 1:4, lwd = 2,
         legend = colnames(fitted(mfit)))
```

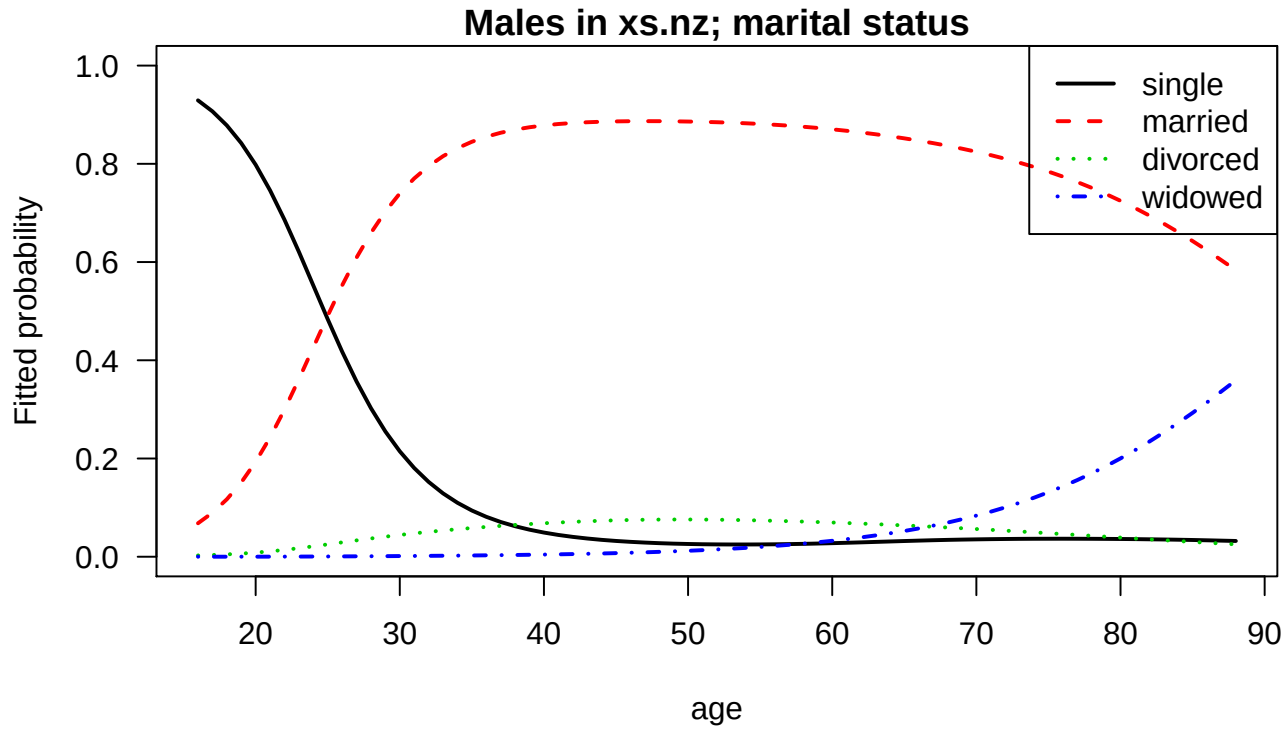


Figure : Fitted multinomial logit model to males in the `xs.nz` data frame in VGAMdata. The response is `marital`.

Look at the component functions:

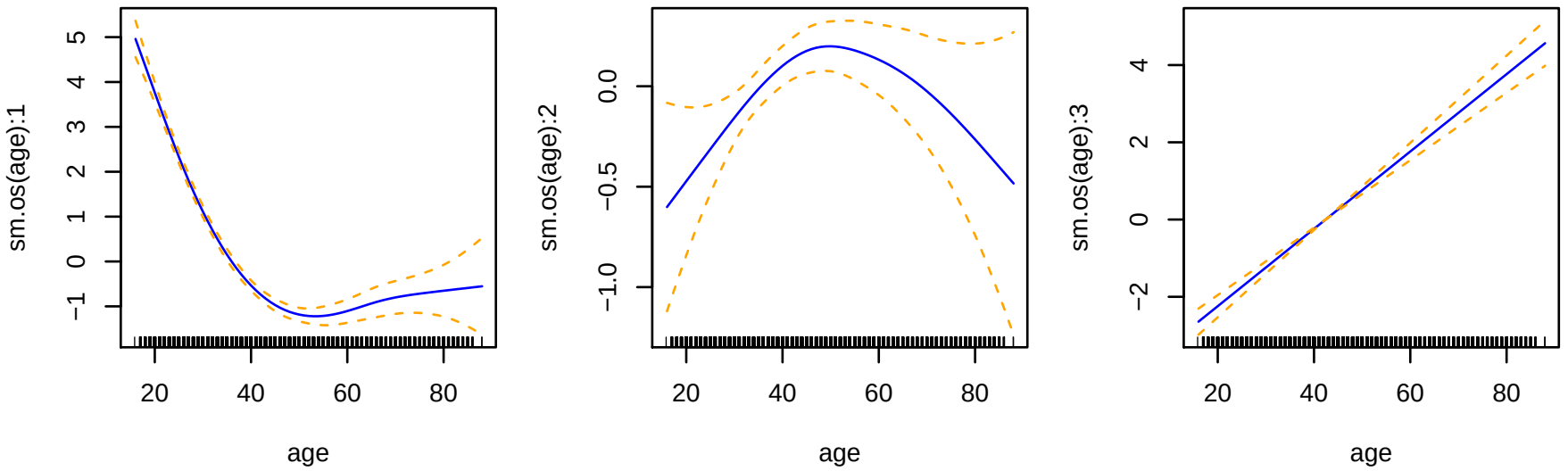


Figure : Estimated component functions, with pointwise $\pm$ 2 SE bands.

Without smoothing . . .

```
> mfit2 <- vglm(marital ~ age, multinomial, data = males)
>
> plot(fitted(mfit2)[, 1] ~ age, males, type = "n", ylim = 0:1,
       ylab = "Fitted probability", las = 1,
       main = "Males in xs.nz; marital status")
> matlines(with(males, age), fitted(mfit2), col = 1:4, lty = 1:4, lwd = 2)
> legend("topright", col = 1:4, lty = 1:4, lwd = 2,
         legend = colnames(fitted(mfit2)))
```

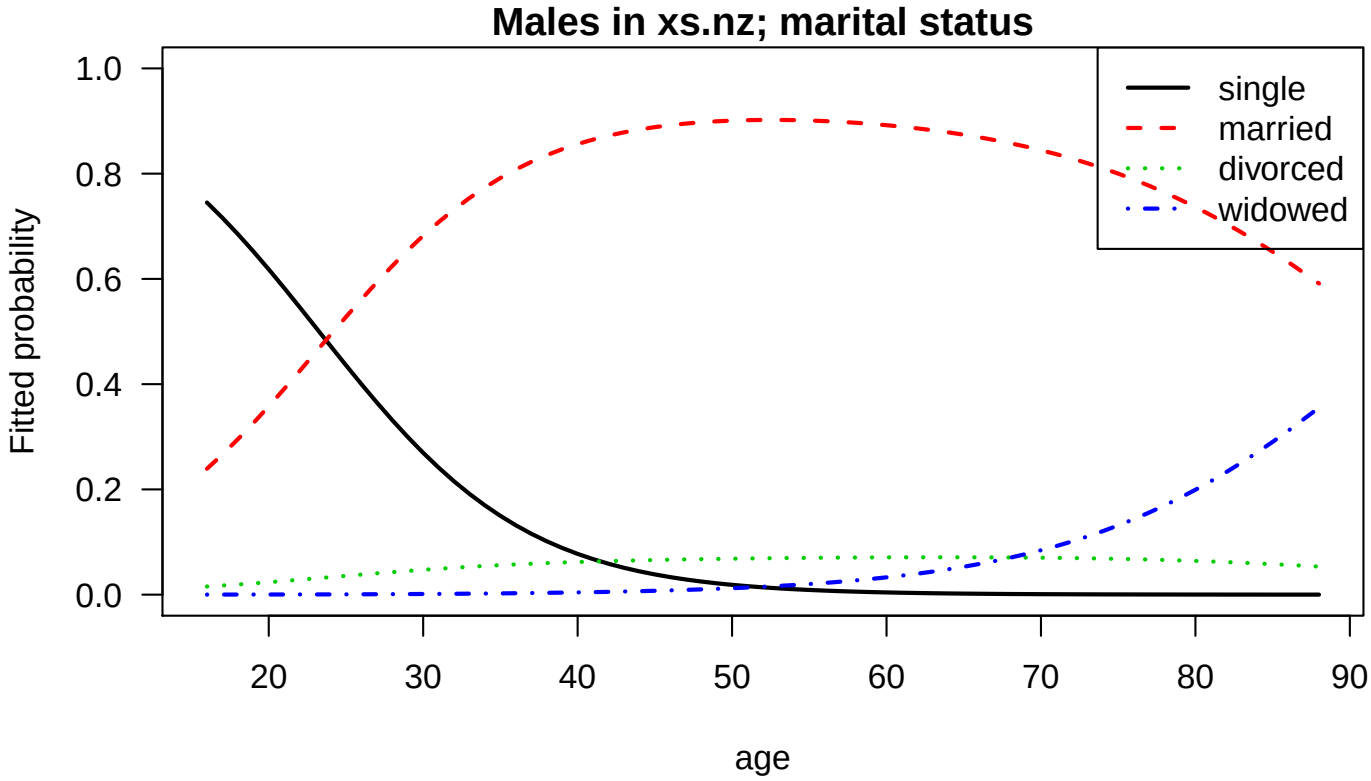


Figure : Without smoothing...

Generalized Additive Models (GAMs)

The LM

$$Y = \beta_1 X_1 + \cdots + \beta_p X_p + \varepsilon, \quad \varepsilon \sim N(0, \sigma^2) \text{ independently}, \tag{7}$$

has some strong assumptions:

1. Linearity, i.e., the effect of each X_k on $E(Y)$ is linear,
2. Normal errors with 0 mean, constant variance, and independent,
3. Additivity, i.e., X_s and X_t do not interact; they have an additive effect on the response.

We relax the linearity assumption using smoothers (just like for the LM).

The *linear predictor* becomes an *additive predictor*:

$$\eta(\mathbf{x}) \;=\; f_1(x_1) + \cdots + f_p(x_p), \tag{8}$$

a sum of arbitary smooth functions.

Additivity is still assumed. Easy to interpret.

Identifiability: the $f_k(x_k)$ are centred.

Very useful for exploratory data analysis. Allows the data to "speak for itself". *Data-driven*, not *model-driven*.

Some GAM books are Hastie and Tibshirani (1990) and Wood (2006, 2017).

In theory, the following are simple GAMs.

```
 lm(y ~ bs(x2) + ns(x3), data = ldata)
glm(y ~ bs(x2) + ns(x3, df = 5), binomial, data = bdata)
glm(y ~ bs(x2, df = 4) + ns(x3), poisson, data = pdata)
```

Problem: the fitted smooths aren't plotted!

Quick-and-dirty remedy: use effects.

Remedy: use some specialized packages that fit GAMs. The following are popular.

- gam written by Trevor Hastie, is similar to the **S-PLUS** version.
- gamlss from London.
- mgcv by Simon Wood, has an emphasis on smoothing parameter selection. The most cutting edge implementation.
- VGAM by Thomas Yee, at Auckland.

Examples

Example 1 Kauri tree data

Y = presence/absence of a tree species, agaaus, which is *Agathis australis*, better known as Kauri. Data is from 392 sites from the Hunua forest.

Figure : Big Kauri tree.

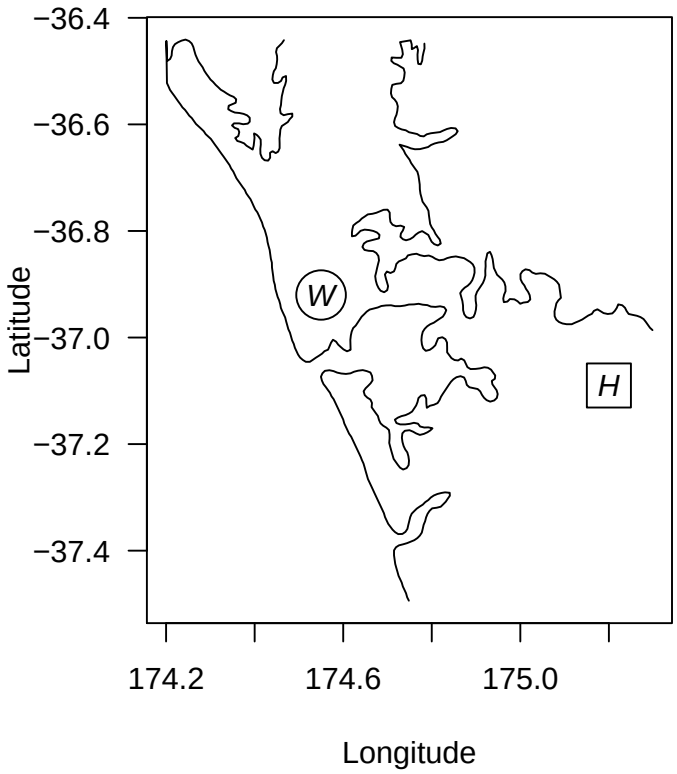


Figure : The Hunua and Waitakere Ranges.

$$\text{logit}\, P[Y_{\texttt{agaaus}} = 1] \;=\; f(\texttt{altitude})$$

where f is a smooth function determined from the data.

```
> ooo <- with(hunua, order(altitude))
> shunua <- hunua[ooo, ]  # Sort by altitude
> fit.h <- vgam(agaaus ~ sm.ps(altitude),  # Use sm.os() or sm.ps()
                binomialff, data = shunua)  # trace = TRUE
> plot(fit.h, se = TRUE, lcol = "blue", scol = "orange", llwd = 2, slwd = 2)
> plot(fitted(fit.h) ~ altitude, data = shunua,
       type= "l", ylim = c(0, 1),  # ylim contains 1 for a reason!
       lwd = 2, col = "blue", xlab = "altitude", ylab = "Fitted value")
> with(hunua, points(altitude, jitter(agaaus, f = 0.1), col = "orange"))
```

The smooth appears to be nonlinear. Possibly, one might infer that the optimal altitude for the species is around 120 m.

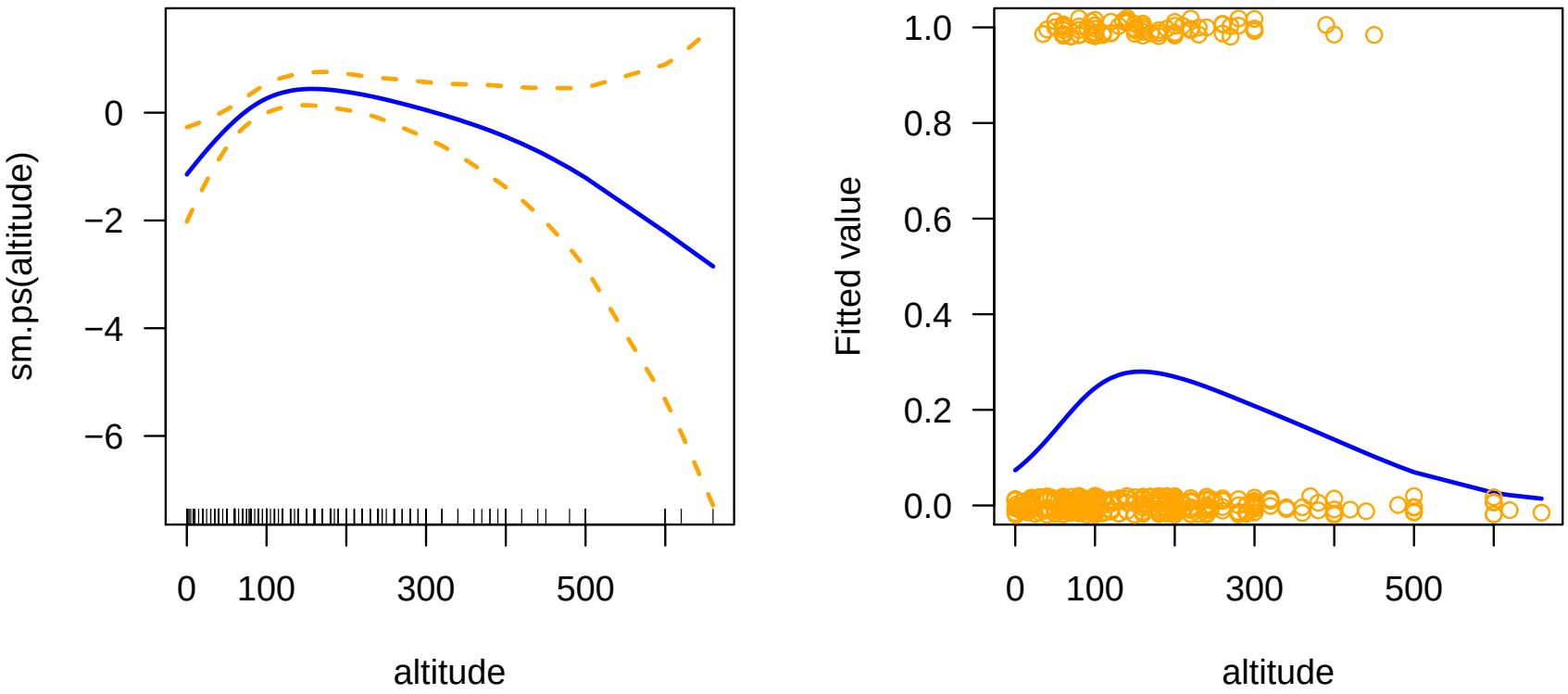


Figure : GAM plots of Kauri data in the Hunua ranges.

```
> summary(fit.h, presid = FALSE)



Call:
vgam(formula = agaaus ~ sm.ps(altitude), family = binomialff,
    data = shunua)


Parametric coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)    -1.39       0.13     -10   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                edf Est.rank Chi.sq p-value
sm.ps(altitude)   3        7     11     0.1

Number of linear/additive predictors:    1

Name of linear/additive predictor: logit(prob)

(Default) Dispersion Parameter for binomialff family:   1

Residual deviance:  388.6 on 400 degrees of freedom

Log-likelihood: -194.3 on 400 degrees of freedom

Number of outer iterations:  7

Number of IRLS iterations at final outer iteration:  2
```

Now apply a transformation (try the ladder of powers).

```
> hfit2 <- vgam(agaaus ~ sm.ps(sqrtalt),  # Use sm.os() or sm.ps()
                binomialff, data = shunua)
>
> hfit3 <- vglm(agaaus ~
                poly(sqrtalt, 2, raw = TRUE),  # Same as: sqrtalt + altitude
                binomialff, data = shunua)
>
> # Plot the parametric and nonparametric fits together
> plot(hfit2, se = TRUE, shade = TRUE)
> mycol <- "orange"
> plot(as(hfit3, "vgam"), add = TRUE, se = TRUE, lcol = mycol, scol = mycol)
>
> plot(fitted(hfit2) ~ sqrtalt, shunua, type = "l")
> lines(fitted(hfit3) ~ sqrtalt, shunua, col = mycol)
> with(shunua, rug(sqrtalt))
```

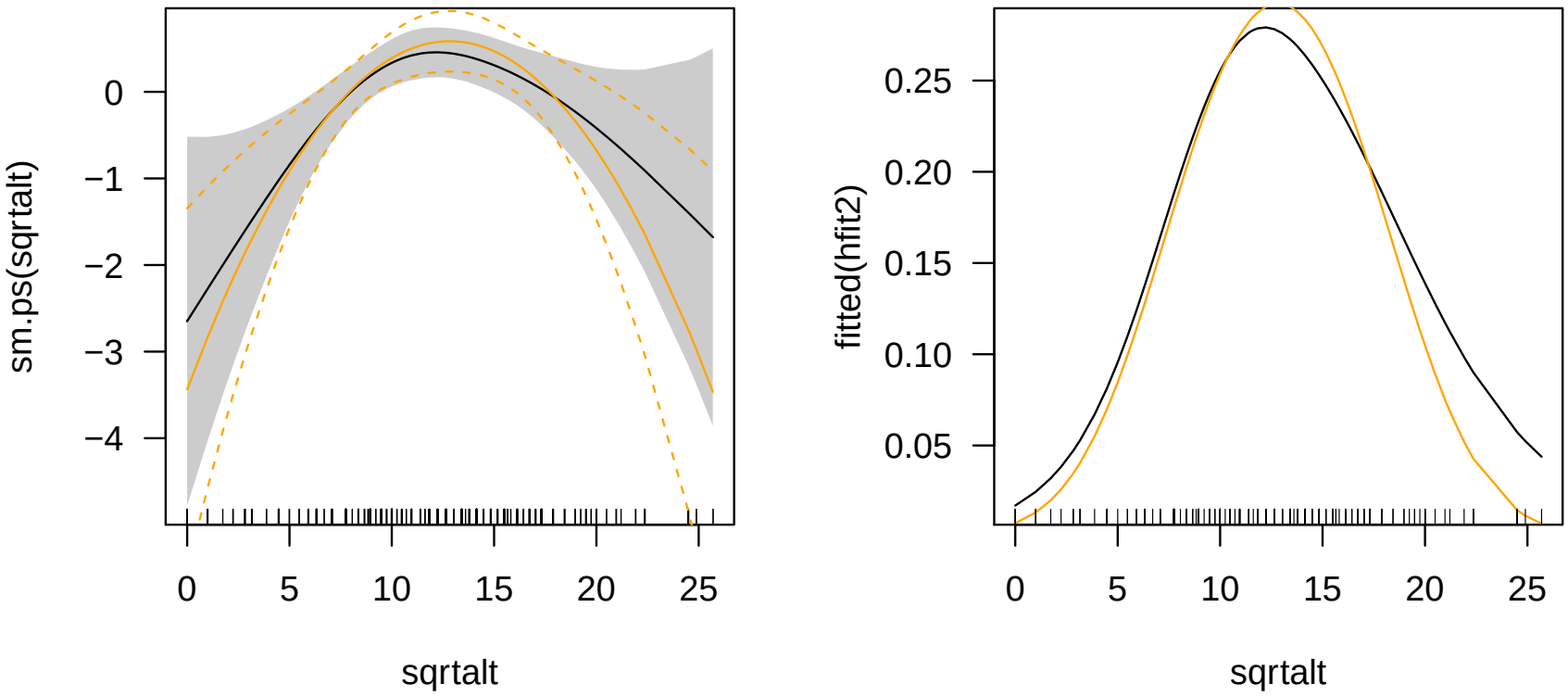


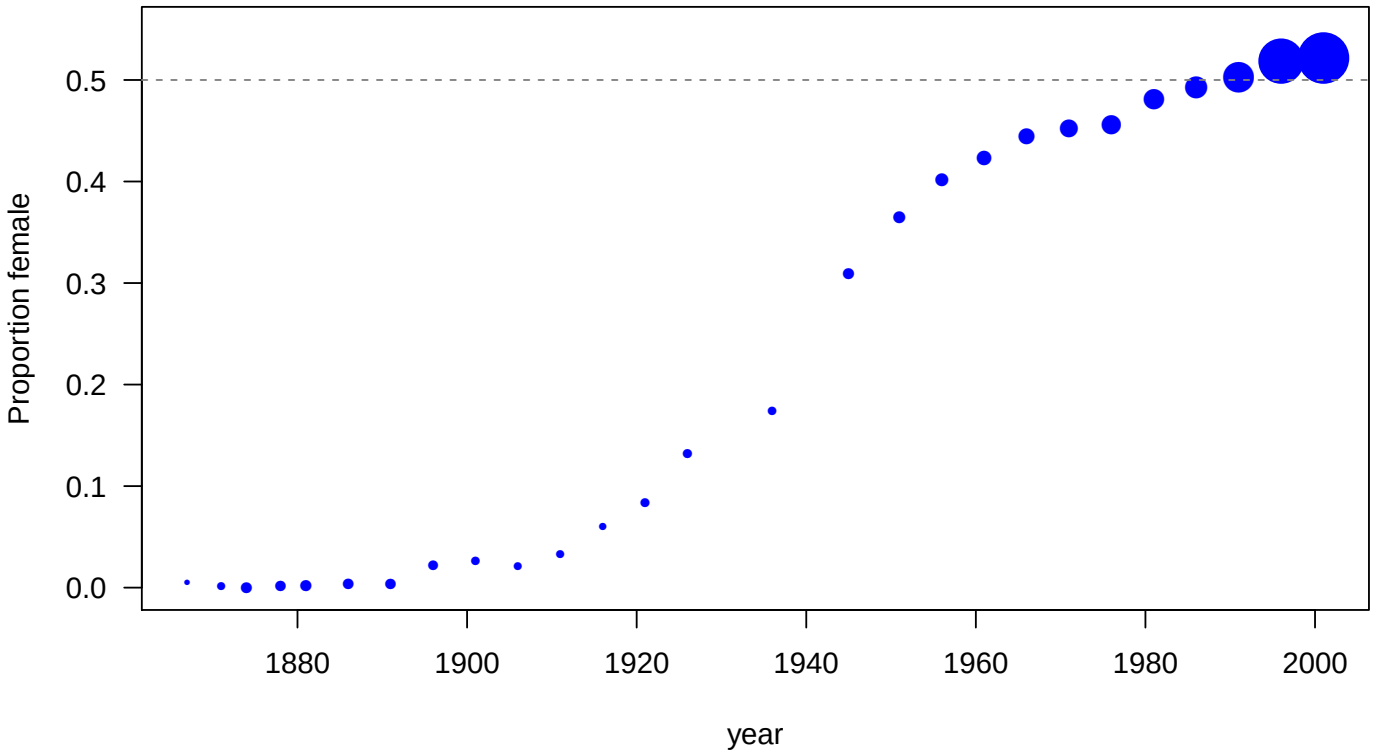
Figure : GAM plots of Kauri data in the Hunua ranges, using $\sqrt{\texttt{altitude}}$ as explanatory.

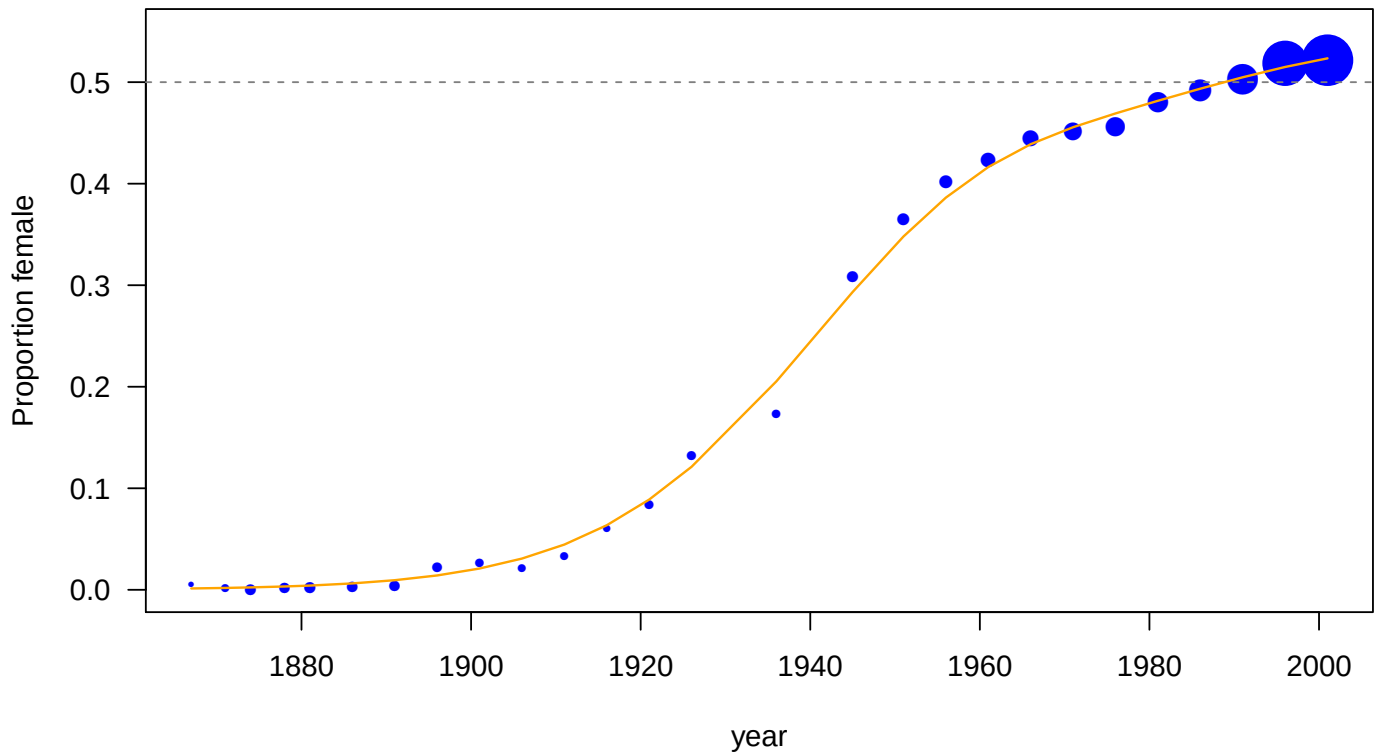
Using the additive model, we have obtained the parametric model

$$\text{logit } P[Y_{\texttt{agaaus}} = 1] \;=\; \beta_1 + \beta_2 \cdot \sqrt{\texttt{altitude}} + \beta_3 \cdot \texttt{altitude}.$$

This illustrates an important use of smoothing: *suggesting transformations of the covariates that results in a parametric (linear) model*.

Example 2 NZ Chinese Data





The curve is a logistic regression GAM which uses a smoothing spline having 4 EDF (1 = linear fit).

Six Illustrative Models

Some central VGLM/VGAM concepts are:

- parameter link functions $g_j(\theta_j)$ applied to all parameters,
- multivariate responses, and sometimes multiple responses too,
- linear predictors $\eta_j = \boldsymbol{\beta}_j^T \mathbf{x}$ and additive predictors $\eta_j = \sum_{k=1}^{d} f_{(j)k}(x_k)$,
- constraints on the functions $(\mathbf{H}_1, \mathbf{H}_2, \ldots, \mathbf{H}_p)$,
- η_j-specific covariates (i.e., $\eta_j(\mathbf{x}_{ij})$) via the `xij` facility,
- reduced-rank regression (RRR), latent variables $\boldsymbol{\nu} = \mathbf{C}^T \mathbf{x}_2$, ordination,
- Fisher scoring, iteratively reweighted least squares (IRLS), maximum likelihood estimation,
- the VGAM package, which presently fits over 150 models and distributions.

To make these concepts concrete let's look at 6 statistical models.

Data: $(\mathbf{x}_i, \mathbf{y}_i)$ for $i = 1, \ldots, n$.

- $\mathbf{x}_i$ is a vector of explanatory variables. Sometimes we drop the i and write $\mathbf{x} = (x_1, \ldots, x_p)^T$ to focus only on the variables, with $x_1 = 1$ denoting an intercept.
- $\mathbf{y}_i$ is a (possibly vector) response.
- n independent observations.

To fit a regression model involving parameters θ_j, VGLMs model each parameter, transformed if necessary, as a linear combination of the explanatory variables. That is,

$$g_j(\theta_j) \;=\; \eta_j \;=\; \boldsymbol{\beta}_j^T \mathbf{x} \;=\; \beta_{(j)1}\, x_1 + \cdots + \beta_{(j)p}\, x_p, \quad j = 1, \ldots, M, \quad (9)$$

where g_j is a *parameter link function*. Potentially *every* parameter is modelled using *all* explanatory variables x_k. And the parameters need not be a mean (such as for GLMs).

VGAMs extend (9) to

$$g_j(\theta_j) \;=\; \eta_j \;=\; \sum_{k=1}^{d} f_{(j)k}(x_k), \qquad j = 1, \ldots, M, \tag{10}$$

i.e., an additive model for each parameter. The functions $f_{(j)k}$ are merely assumed to be smooth and are estimated by smoothers such as splines, therefore the whole approach is *data-driven* rather than *model-driven*.
In (10) $f_{(j)1}(x_1) = \beta_{(j)1}$ is simply an intercept.

(1) The Linear Model

Linear model (LM): for a response $Y \sim N(\mu,\ \sigma^2)$ with

$$\mu \ = \ \eta_1 \ = \ \boldsymbol{\beta}_1^T \mathbf{x}. \tag{11}$$

Standard GLM theory treats σ as a *scale parameter* (the exponential family is restricted to 1 parameter). But VGLMs/VGAMs couple (11) with

$$\log \sigma \ = \ \eta_2 \ = \ \boldsymbol{\beta}_2^T \mathbf{x}. \tag{12}$$

A log link is generally suitable because $\sigma > 0$.

`lm()` cannot fit (12)!

Modelling η_2 as *intercept-only* means that the typical assumption of constant variance (*homoscedasticity*) is made:

$$\log \sigma \ = \ \eta_2 \ = \ \beta_{(2)1}. \tag{13}$$

If $\exists\ x_2$ then we might test $H_0 : \beta_{(2)2} = 0$ in

$$\log \sigma \;=\; \eta_2 \;=\; \beta_{(2)1} + \beta_{(2)2}\, x_2 \tag{14}$$

as a test for no *heteroscedasticity*.

The VGAM family `uninormal(zero = NULL)` implements (11)–(12), the response being an univariate normal. Typical call of the form:

```
vglm(y ~ x2 + x3 + x4, family = uninormal, data = udata)

# cf.
  lm(y ~ x2 + x3 + x4, data = udata)
```

Function `uninormal()` is assigned to the `family` argument; known as a *VGAM family function*. It makes the usual LM assumptions:

- independence and
- normality of the errors $y_i - \mu_i$,
- linearity (11), and
- constant variance (13).

((2) & (3)) Poisson and Negative Binomial Regression

The Poisson distribution is as fundamental to the analysis of count data as the normal (Gaussian) is to continuous responses. Its *PMF* is

$$\Pr(Y = y; \mu) \;=\; e^{-\mu}\,\mu^y/y!\,, \qquad y = 0, 1, 2, \ldots, \qquad \mu > 0. \tag{15}$$

This gives $E(Y) = \mu = \mathrm{Var}(Y)$. As μ is positive, use

$$\eta \;=\; \log\,\mu. \tag{16}$$

Interpretation

An increase of 1 unit for x_k, keeping other variables fixed, implies

$$\mu(x_1, \ldots, x_{k-1}, x_k + 1, x_{k+1}, \ldots, x_p) \;=\; \mu(\mathbf{x} + \mathbf{e}_k) \;=\; \mu(\mathbf{x}) \cdot e^{\beta_k}, \tag{17}$$

i.e., e^{β_k} is the *multiplicative* effect on $\mu(\mathbf{x})$ of increasing x_k by one unit. So a positive/negative value of β_k corresponds to an increasing/decreasing effect respectively.

Offsets

Counts sometimes arise from an underlying *rate*, e.g., if λ is the mean rate per unit time then $\mu = \lambda t$ is the mean number of events during a period of time t.

Example: λ is the mean number of earthquakes per annum in Sardinia which exceed a certain magnitude, so $\mu = \lambda t =$ the expected number of earthquakes during a t-year period.

Use

$$\log \mu \equiv \eta = (\log t) + \log \lambda. \tag{18}$$

When we want to model the rate, adjusting for time t, and provided that t is known, then the Poisson regression (18) involves (known) *offsets*.

```
vglm(y ~ x2 + x3 + x4 + offset(log(time.period)), family = poissonff, pdata)
vglm(y ~ x2 + x3 + x4, offset = log(time.period), family = poissonff, pdata)
```

Overdispersion

Mean $=$ variance is unrealistic!

Usually mean $<$ variance: "*overdispersion* with respect to the Poisson distribution."

Common remedy:

$$\mathrm{Var}(Y) \;=\; \phi \cdot \mu \tag{19}$$

in the standard Poisson regression (16), where ϕ is estimated by the method of moments. *Ahhhhhhhhhhhhh!*

Then $\widehat{\phi} > 1$ indicates overdispersion relative to a Poisson distribution. The quasi-Poisson estimate $\widehat{\boldsymbol{\beta}}$ coincides with the usual maximum likelihood estimate (MLE).

Negative Binomial Regression

This is a better method of handling overdispersion. A NB random variable Y has PMF

$$\Pr(Y = y; \mu, k) = \binom{y + k - 1}{y} \left(\frac{\mu}{\mu + k}\right)^y \left(\frac{k}{k + \mu}\right)^k, \quad y = 0, 1, 2, \ldots, \tag{20}$$

with positive parameters μ $(= E(Y))$ and k. The quantity k^{-1} is known as the *dispersion* or *ancillary parameter*.

```
> args(dnbinom)

function (x, size, prob, mu, log = FALSE)
NULL
```

Some notes:

- Poisson distribution is the limit as $k \to \infty$.
- Overdispersion relative to the Poisson is accommodated. But *underdispersion* ($\phi < 1$) isn't:

$$\mathrm{Var}(Y) = \mu + \frac{\mu^2}{k} = \mu\left(1 + \frac{\mu}{k}\right) \geq \mu \tag{21}$$

The VGLM/VGAM framework can naturally fit

$$\log \mu = \eta_1 = \boldsymbol{\beta}_1^T \mathbf{x}, \tag{22}$$
$$\log k = \eta_2 = \boldsymbol{\beta}_2^T \mathbf{x}, \tag{23}$$

which is known as a *NB-H*. Use:

```
vglm(y ~ x2 + x3 + x4, family = negbinomial(zero = NULL), data = ndata)
```

Many VGAM family functions can handle *multiple responses*, e.g.,

```
vglm(cbind(y1, y2) ~ x2 + x3 + x4, family = negbinomial(zero = NULL), ndata)
```

regresses 2 independent responses simultaneously.
Then $\boldsymbol{\eta} = (\eta_1, \eta_2, \eta_3, \eta_4)^T = (\log \mu_1, \log k_1, \log \mu_2, \log k_2)^T$.

We'll see many NB variants later!

Table : A simplified summary of VGAM and most of its framework. The latent variables $\boldsymbol{\nu} = \mathbf{C}^T \mathbf{x}_2$, or $\nu = \mathbf{c}^T \mathbf{x}_2$ if rank $R = 1$. Here, $\mathbf{x}^T = (\mathbf{x}_1^T, \mathbf{x}_2^T)$. and $\mathbf{x}_1 = 1$.

t

$\boldsymbol{\eta} = (\eta_1, \ldots, \eta_M)^T$	**Model**	**Modelling function**	**Reference**
$\mathbf{B}_1^T \mathbf{x}_1 + \mathbf{B}_2^T \mathbf{x}_2 \ (= \mathbf{B}^T \mathbf{x})$	VGLM	`vglm()`	Yee & Hastie (2003)
$\mathbf{B}_1^T \mathbf{x}_1 + \sum_{k=p_1+1}^{p_1+p_2} \mathbf{H}_k \, \mathbf{f}_k^*(x_k)$	VGAM	`vgam()`	Yee & Wild (1996)
$\mathbf{B}_1^T \mathbf{x}_1 + \mathbf{A} \, \boldsymbol{\nu}$	RR-VGLM	`rrvglm()`	Yee & Hastie (2003)
$\mathbf{B}_1^T \mathbf{x}_1 + \mathbf{A} \, \boldsymbol{\nu} + \begin{pmatrix} \boldsymbol{\nu}^T \mathbf{D}_1 \boldsymbol{\nu} \\ \vdots \\ \boldsymbol{\nu}^T \mathbf{D}_M \boldsymbol{\nu} \end{pmatrix}$	QRR-VGLM	`cqo()`	Yee (2004)
$(\beta_0 + \alpha_i) \, \mathbf{1} + \boldsymbol{\gamma} + \mathbf{A} \boldsymbol{\nu}_i$	RCIM	`rcim()`	Yee and Hadi (2014)

(4) Bivariate Odds Ratio Model

Logistic Regression

$Y = 1$ ("success") or 0 ("failure"), i.e., 1 binary response.
Result: $E(Y) = \Pr(Y = 1) = p$, say.

Then the logistic regression model can be written

$$\text{logit } p(\mathbf{x}) \equiv \log \frac{p(\mathbf{x})}{1 - p(\mathbf{x})} = \eta(\mathbf{x}), \tag{24}$$

where the quantity $p/(1-p)$ is known as the *odds* of event $Y = 1$.

Interpretation:

- e.g., odds $= 3$ means that the event is 3 times more likely to occur than not occur.
- $$\text{odds}(Y = 1|x_k + \Delta) = \exp\{\beta_{(1)k}\,\Delta\} \cdot \text{odds}(Y = 1|x_k),$$

  keeping all other variables in $\mathbf{x}$ fixed.

Now 2 Responses...

In some applications it is natural to measure *two* binary responses, Y_1 and Y_2, say,

Examples:

- measurements of deafness in both ears,
- presence/absence of cataracts in elderly patients' eyes,
- presence/absence of 2 plant species at sites in a huge forest,
- `coalminers`,
- `hunua`,
- `xs.nz[, c("cat", "dog")]`, etc.

Fit the *bivariate odds ratio model*

$$\text{logit } p_j(\mathbf{x}) = \eta_j(\mathbf{x}), \quad j = 1, 2, \tag{25}$$

$$\log \psi(\mathbf{x}) = \eta_3(\mathbf{x}). \tag{26}$$

Notes:

- This couples 2 logistic regressions together with an equation for the *odds ratio*. *Very natural!*
- The responses are often dependent, and the odds ratio is a natural measure for the association between 2 binary variables.
- Its a full-likelihood model. The joint probability $p_{11}(\mathbf{x})$ can be obtained from the 2 marginals $p_j(\mathbf{x}) = \Pr(Y_j = 1|\mathbf{x})$ and the odds ratio

$$\psi(\mathbf{x}) = \frac{p_{00}(\mathbf{x})\, p_{11}(\mathbf{x})}{p_{01}(\mathbf{x})\, p_{10}(\mathbf{x})} = \frac{\Pr(Y_1 = 0, Y_2 = 0|\mathbf{x}) \Pr(Y_1 = 1, Y_2 = 1|\mathbf{x})}{\Pr(Y_1 = 0, Y_2 = 1|\mathbf{x}) \Pr(Y_1 = 1, Y_2 = 0|\mathbf{x})}.$$

Then Y_1 and Y_2 are independent iff $\psi = 1$.

- ψ is the ratio of two odds:

$$\psi(\mathbf{x}) = \frac{\text{odds}(Y_1 = 1 | Y_2 = 1, \mathbf{x})}{\text{odds}(Y_1 = 1 | Y_2 = 0, \mathbf{x})}. \tag{27}$$

The typical call is of the form

```
vglm(cbind(y00, y01, y10, y11) ~ x2 + x3, binom2.or, data = bdata)
```

where the LHS matrix contains the joint frequencies, e.g., `y01` $= (Y_1 = 0, Y_2 = 1)$.

Idea: use a different link for η_1 and η_2,

```
vglm(cbind(y00, y01, y10, y11) ~ x2 + x3, data = bdata,
     binom2.or(lmu = "cloglog"))
```

fits

$$\begin{aligned} \log(-\log(1 - p_j(\mathbf{x}))) &= \eta_j(\mathbf{x}), \quad j = 1, 2, \\ \log\ \psi &= \eta_3. \end{aligned}$$

Exchangeability

Sometimes we want to constrain

$$p_1(\mathbf{x}) \;=\; p_2(\mathbf{x}),$$

e.g., $Y_j =$ presence/absence of deafness in the LHS and RHS ears.

This corresponds to an *exchangeable* error structure, and constraining $\eta_1 = \eta_2$ can be handled with the *constraints-on-the-functions* framework.

Also, usually constrain ψ to be *intercept-only*.

With eyes data, because of symmetry, the *exchangeable* model constrains $\eta_1 = \eta_2$:

$$\begin{aligned} \texttt{logit}\ p_j(\mathbf{x}) &= \eta_1(\mathbf{x}), \qquad j = 1, 2, \\ \log\ \psi(\mathbf{x}) &= \eta_3(\mathbf{x}). \end{aligned}$$

For example,

$$\begin{aligned} \eta_1(\mathbf{x}_i) &= \beta^*_{(1)1} + \beta^*_{(1)2}\, x_{i2} + \beta^*_{(1)3}\, x_{i3}, \\ \eta_2(\mathbf{x}_i) &= \beta^*_{(1)1} + \beta^*_{(1)2}\, x_{i2} + \beta^*_{(1)3}\, x_{i3}, \\ \eta_3(\mathbf{x}_i) &= \beta^*_{(2)1}. \end{aligned}$$

The odds-ratio is intercept-only.

Can write this as

$$\begin{pmatrix} \eta_1(\mathbf{x}_i) \\ \eta_2(\mathbf{x}_i) \\ \eta_3(\mathbf{x}_i) \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \beta^*_{(1)1} \\ \beta^*_{(2)1} \end{pmatrix} x_{i1} + \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \beta^*_{(1)2}\, x_{i2} + \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \beta^*_{(1)3}\, x_{i3}.$$

The general form is

$$\boldsymbol{\eta}(\mathbf{x}_i) \quad = \quad \sum_{k=1}^{p} \mathbf{H}_k \, \boldsymbol{\beta}^*_{(k)} \, x_{ik}. \tag{28}$$

((5) & (6)) Proportional Odds and Multinomial Logit Models

(I) Ordinal Response

Response Y is *ordinal* (an ordered categorical or grouping variable or factor), e.g., $Y = 1 =$ 'low', $Y = 2 =$ 'medium', $Y = 3 =$ 'high'.

Ordinal responses are naturally modelled using cumulative probabilities $\Pr(Y \leq j|\mathbf{x})$.

The *proportional odds model* for a general ordinal Y taking levels $\{1, 2, \ldots, M+1\}$ is:

$$\text{logit}\, \Pr(Y \leq j|\mathbf{x}) \;=\; \eta_j(\mathbf{x}), \tag{29}$$

subject to the constraint that

$$\eta_j(\mathbf{x}) \;=\; \beta^*_{(j)1} + \mathbf{x}_{[-1]}^T \boldsymbol{\beta}^*_{[-(1:M)]}, \qquad j = 1, \ldots, M. \tag{30}$$

Notes:

- Here, $\mathbf{x}_{[-1]}$ is $\mathbf{x}$ with the first element (the intercept) deleted.
- The superscript "$*$" denotes regression coefficients that are to be estimated.
- Equation (30) describes M *parallel* surfaces in $(p-1)$-dimensional space.
- The VGAM family functions `cumulative()` and `propodds()` fit this model and variants thereof.

Some further comments:

(i) *Selecting different link functions*

Let $\gamma_j(\mathbf{x}) = \Pr(Y \leq j|\mathbf{x})$. The proportional odds model is also known as the *cumulative logit model*; there are M simultaneous logistic regressions applied to the γ_j. If we replace the logit link in (29) by a probit link say, then this may be referred to as a *cumulative probit model*.

Use `cumulative(link = "probit")`.

(ii) *Non-proportional odds model*

In (30) the η_j are *parallel* on the logit scale because the estimable regression coefficients $\boldsymbol{\beta}^*_{[-(1:M)]}$ in (30) are common for all j.
They do not intersect!
So $\Pr(Y = j|\mathbf{x})$ are not negative or greater than unity for some $\mathbf{x}$.
This is known as the so-called *parallelism* or *proportional odds assumption*.

(iii) *Partial proportional odds model*

Some explanatory variables parallel and others not.
Example: suppose $p = 4$, $M = 2$ and

$$\begin{aligned} \eta_1 &= \beta^*_{(1)1} + \beta^*_{(1)2}\, x_2 + \beta^*_{(1)3}\, x_3 + \beta^*_{(1)4}\, x_4, \\ \eta_2 &= \beta^*_{(2)1} + \beta^*_{(1)2}\, x_2 + \beta^*_{(2)3}\, x_3 + \beta^*_{(1)4}\, x_4. \end{aligned}$$

The parallelism assumption applies to x_2 and x_4 only. This may be fitted by

```
vglm(ymatrix ~ x2 + x3 + x4, cumulative(parallel = TRUE ~ x2 + x4 - 1), cdata)
```

or equivalently,

```
vglm(ymatrix ~ x2 + x3 + x4, cumulative(parallel = FALSE ~ x3), data = cdata)
```

(iv) *Common VGAM family function arguments*

Rather than (29) many authors define the proportional odds model as

$$\text{logit}\,\Pr(Y \geq j+1|\mathbf{x}) \;=\; \eta_j(\mathbf{x}), \qquad j = 1, \ldots, M, \tag{31}$$

because $M = 1$ coincides with logistic regression.

The direction has changed!

Many VGAM categorical family functions share a number of common arguments, e.g., `reverse`. Here, setting `reverse = TRUE` will fit (31).
Other common arguments include `link`, `parallel`, `zero`.

(II) Nominal Response

For unordered Y `multinomial()` can fit a *multinomial logit model*

$$\log \frac{\Pr(Y = j|\mathbf{x})}{\Pr(Y = M+1|\mathbf{x})} = \eta_j(\mathbf{x}), \quad j = 1, \ldots, M. \tag{32}$$

The last level is the *baseline group* or *reference group*. Equivalently,

$$\Pr(Y = j|\mathbf{x}) = \frac{\exp\{\eta_j(\mathbf{x})\}}{\sum_{s=1}^{M+1} \exp\{\eta_s(\mathbf{x})\}}, \quad j = 1, \ldots, M. \tag{33}$$

Interpretation: coefficient $\beta_{(j)k}$ is based on increasing the kth variable by one unit, keeping other variables fixed:

$$\beta_{(j)k} = \log \frac{\Pr(Y = j|x_1, \ldots, x_{k-1}, x_k + 1, x_{k+1}, \ldots, x_p)}{\Pr(Y = j|x_1, \ldots, x_{k-1}, x_k, x_{k+1}, \ldots, x_p)}. \tag{34}$$

Introduction to VGLMs and VGAMs

The VGAM package implements several large classes of regression models of which *vector generalized linear and additive models (VGLMs/VGAMs)* are most commonly used.

The primary key words are

- *additive models,*
- *maximum likelihood estimation,*
- *iteratively reweighted least squares (IRLS),*
- *Fisher scoring.*

Other concepts are

- *latent variables (reduced-rank regression),*
- *constrained ordination,*
- *vector smoothing.*

Basically ...

VGLMs model each parameter θ_j, transformed if necessary, as a linear combination of the explanatory variables $\mathbf{x}$. That is,

$$g_j(\theta_j) \;=\; \eta_j \;=\; \boldsymbol{\beta}_j^T \mathbf{x} \;=\; \beta_{(j)1}\, x_1 + \cdots + \beta_{(j)p}\, x_p \tag{35}$$

where g_j is a parameter link function ($-\infty < \eta_j < \infty$).

VGAMs extend (35) to

$$g_j(\theta_j) \;=\; \eta_j \;=\; f_{(j)1}(x_1) + \cdots + f_{(j)p}(x_p), \tag{36}$$

i.e., an *additive model* for each parameter. Estimated by smoothers, this is a *data-driven* approach.

The formula

$$\eta_j(\mathbf{x}) \quad = \quad g_j(\theta_j) \tag{37}$$

represents M surfaces in d-dimensional space. We usually approximate the surface by a plane (35) or an additive model (36).

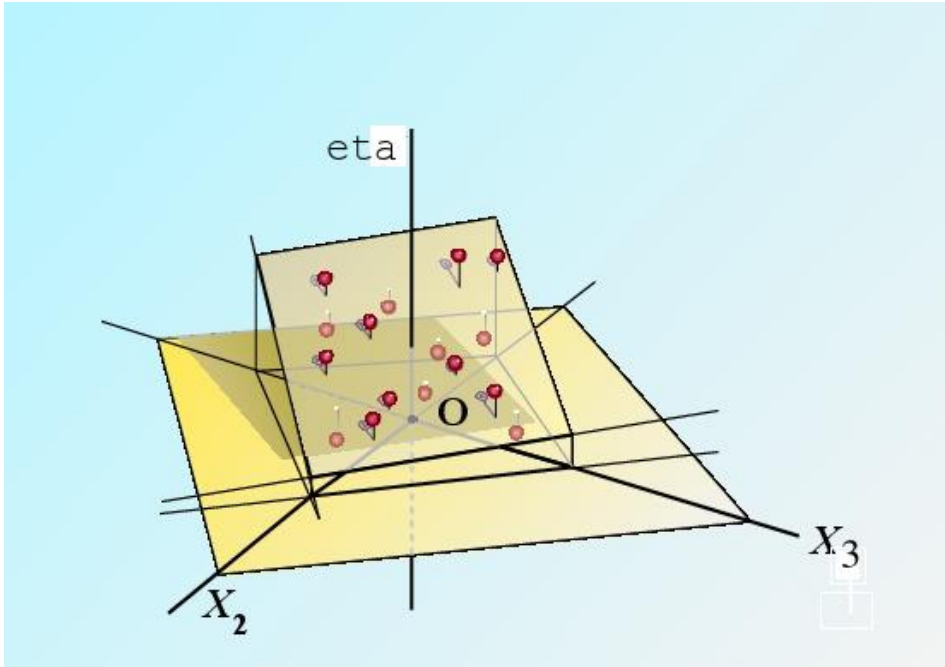


Figure : A linear predictor η_j with covariates $(x_1, x_2, x_3)^T$.

The framework extends GLMs and GAMs in three main ways:

(i) response **y** not restricted to the exponential family,

(ii) multivariate and/or multiple responses **y** and/or *linear/additive predictors* $\boldsymbol{\eta}$ are handled,

(iii) η_j need not be a function of a mean μ: $\eta_j = g_j(\theta_j)$ for *any* parameter θ_j.

This formulation is deliberately general so that it encompasses as many distributions and models as possible. *We wish to be limited only by the assumption that the regression coefficients enter through a set of linear or additive predictors* η_j.

Given the covariates, the conditional distribution of the response is intended to be completely general. *More general* $\Longrightarrow$ *more useful.*

Table : Some VGAM link functions (grouped approximately by their domains).

t

Function	Link $g_j(\theta_j)$	Domain of θ_j	Link name
`cauchit()`	$\tan(\pi(\theta - \frac{1}{2}))$	$(0, 1)$	cauchit
`cloglog()`	$\log\{-\log(1-\theta)\}$	$(0, 1)$	complementary log-log
`logit()`	$\log \frac{\theta}{1-\theta}$	$(0, 1)$	logit
`multilogit()`	$\log \frac{\theta_j}{\theta_{M+1}}$	$(0, 1)^M$	multi-logit; $\sum_{j=1}^{M+1} \theta_j = 1$
`probit()`	$\Phi^{-1}(\theta)$	$(0, 1)$	probit (for "probability unit")
`rhobit()`	$\log \frac{1+\theta}{1-\theta}$	$(-1, 1)$	rhobit
`loge()`	$\log\ \theta$	$(0, \infty)$	log (logarithmic)
`extlogit()`	$\log \frac{\theta - A}{B - \theta}$	(A, B)	extended logit
`explink()`	e^θ	$(-\infty, \infty)$	exponential
`identitylink()`	θ	$(-\infty, \infty)$	identity
`loglog()`	$\log\log(\theta)$	$(1, \infty)$	log-log
`logoff(`θ`, offset =` A`)`	$\log(\theta + A)$	$(-A, \infty)$	log with offset

Two Introductory Models + 3 Pieces of Infrastructure

For the purposes of illustration, let's quickly consider 2 statistical models.

We want to motivate

(i) *multiple responses*,

(ii) *constraint matrices*,

(iii) `xij`.

Data: $(\mathbf{x}_i, \mathbf{y}_i)$ for $i = 1, \ldots, n$ independently.

Explanatory $\mathbf{x}_i$.

Response $\mathbf{y}_i$.

Sometimes drop the subscript i and write $\mathbf{x} = (x_1, \ldots, x_p)$ with $x_1 = 1$ denoting the *intercept*.

1. Poisson and negative binomial distributions

The *Poisson distribution* has probability function

$$\Pr(Y = y; \mu) = e^{-\mu}\,\mu^y/y!\,, \qquad y = 0, 1, 2, \ldots, \qquad \mu > 0, \tag{38}$$

resulting in $E(Y) = \mu = \mathrm{Var}(Y)$. But with 'real' data it is common for $\overline{y} \ll s_y^2$ (*overdispersion*).

Data sets to look at:

- `V1`,
- `machinists`, `pirates1`, `pirates2`,
- `data("azpro", package = "COUNT")`,
- `xs.nz[, c("drinkweek", "babies")]`, etc.

(i) Multiple responses

```
> twoV1 <- vglm(cbind(hits, hits) ~ 1, poissonff,
                weights = cbind(ofreq, ofreq), data = V1)
> coef(twoV1, matrix = TRUE)

            loge(E[hits]) loge(E[hits])
(Intercept)      -0.07011      -0.07011

> head(predict(twoV1))

  loge(E[hits]) loge(E[hits])
1      -0.07011      -0.07011
2      -0.07011      -0.07011
3      -0.07011      -0.07011
4      -0.07011      -0.07011
5      -0.07011      -0.07011
6      -0.07011      -0.07011
```

```
> summary(twoV1, presid = FALSE)


Call:
vglm(formula = cbind(hits, hits) ~ 1, family = poissonff, data = V1,
    weights = cbind(ofreq, ofreq))


Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept):1  -0.0701     0.0432   -1.62      0.1
(Intercept):2  -0.0701     0.0432   -1.62      0.1

Number of linear predictors:  2

Names of linear predictors: loge(E[hits]), loge(E[hits])

Residual deviance: 1337 on 10 degrees of freedom

Log-likelihood: -1465 on 10 degrees of freedom

Number of iterations: 5

No Hauck-Donner effect found in any of the estimates
```

The scope of VGAM is very broad; it potentially covers

- univariate and multivariate distributions,
- categorical data analysis,
- quantile and expectile regression,
- time series,
- survival analysis,
- mixture models,
- extreme value analysis,
- nonlinear regression,
- reduced-rank regression,
- ordination, ...

It conveys GLM/GAM-type modelling to a much broader range of models.

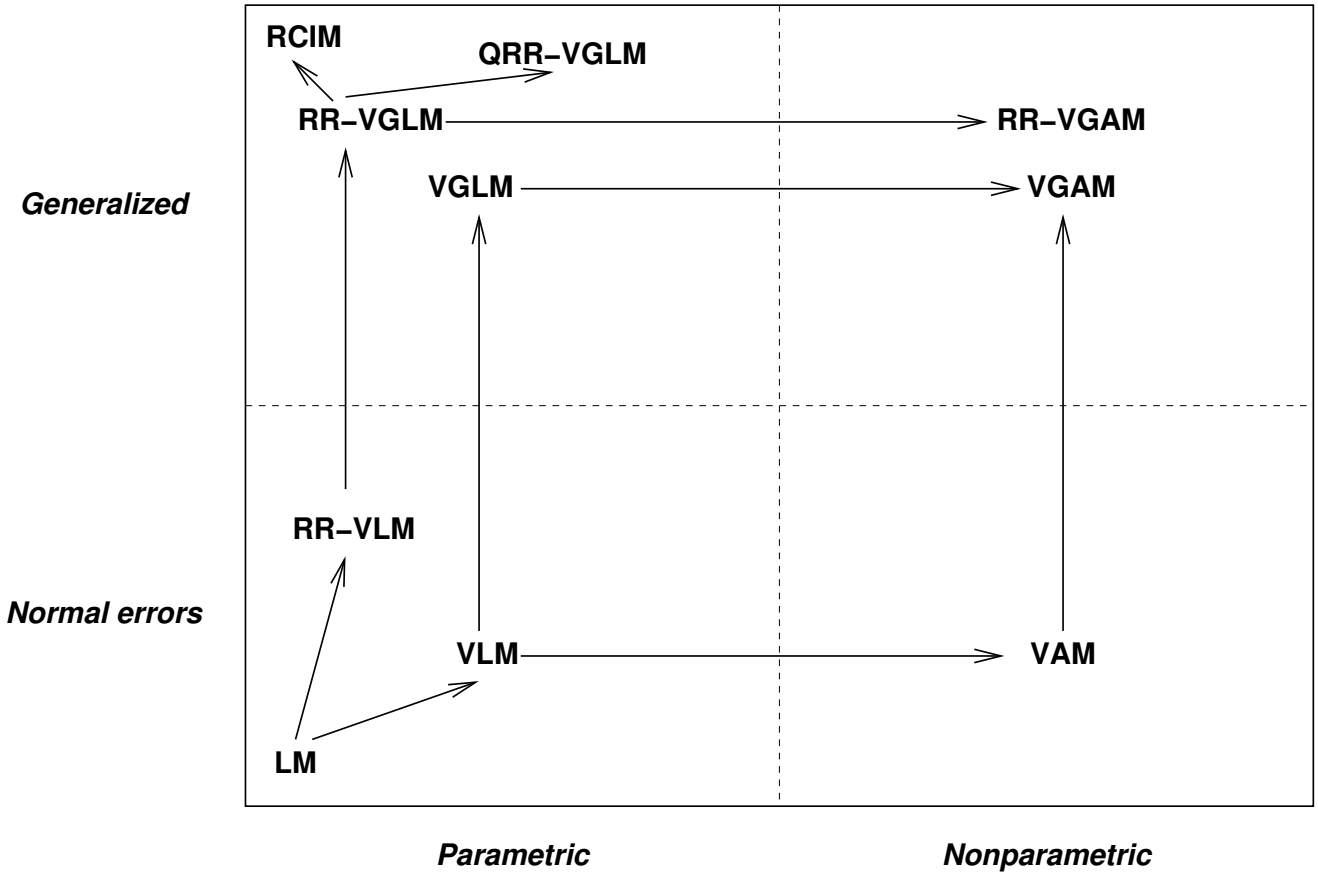


Figure : Flowchart for different classes of models. Legend: **LM** = linear model, **V** = vector, **G** = generalized, **A** = additive, **RR** = reduced-rank, **Q** = quadratic.

t

$\boldsymbol{\eta}$	Model	S function	Reference
$\mathbf{B}_1^T\mathbf{x}_1 + \mathbf{B}_2^T\mathbf{x}_2\ (= \mathbf{B}^T\mathbf{x})$	VGLM	`vglm()`	Yee & Hastie (2003)
$\mathbf{B}_1^T\mathbf{x}_1 + \sum_{k=p_1+1}^{p_1+p_2} \mathbf{H}_k \mathbf{f}_k^*(x_k)$	VGAM	`vgam()`	Yee & Wild (1996)
$\mathbf{B}_1^T\mathbf{x}_1 + \mathbf{A}\,\boldsymbol{\nu}$	RR-VGLM	`rrvglm()`	Yee & Hastie (2003)
$\mathbf{B}_1^T\mathbf{x}_1 + \mathbf{A}\,\boldsymbol{\nu} + \begin{pmatrix} \boldsymbol{\nu}^T\mathbf{D}_1\boldsymbol{\nu} \\ \vdots \\ \boldsymbol{\nu}^T\mathbf{D}_M\boldsymbol{\nu} \end{pmatrix}$	QRR-VGLM	`cqo()`	Yee (2004)
$\mathbf{B}_1^T\mathbf{x}_1 + \sum_{r=1}^{R} \mathbf{f}_r(\nu_r)$	RR-VGAM	`cao()`	Yee (2006)

Table : A summary of VGAM and its framework. The latent variables $\boldsymbol{\nu} = \mathbf{C}^T\mathbf{x}_2$, or $\nu = \mathbf{c}^T\mathbf{x}_2$ if rank $R = 1$. Here, $\mathbf{x}^T = (\mathbf{x}_1^T, \mathbf{x}_2^T)$. Abbreviations: A = additive, C = constrained, L = linear, O = ordination, Q = quadratic, RR = reduced-rank, VGLM = vector generalized linear model.

Vector Generalized Linear Models

Data $(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)$ on n independent "individuals".

Definition Conditional distribution of $\mathbf{y}$ given $\mathbf{x}$ is

$$f(\mathbf{y}|\mathbf{x}; \boldsymbol{\beta}) = f(\mathbf{y}, \eta_1, \ldots, \eta_M, \boldsymbol{\phi}),$$

for $j = 1, \ldots, M$ and some function f,

$$\begin{aligned} \eta_j &= \eta_j(\mathbf{x}) = \boldsymbol{\beta}_j^T \mathbf{x}, \\ \boldsymbol{\beta}_j &= (\beta_{(j)1}, \ldots, \beta_{(j)p})^T, \\ \boldsymbol{\beta} &= (\boldsymbol{\beta}_1^T, \ldots, \boldsymbol{\beta}_M^T)^T, \\ \boldsymbol{\phi} &= \text{a vector of scale factors.} \end{aligned} \tag{48}$$

Often $g_j(\theta_j) = \eta_j$ for parameters θ_j and link functions g_j.

The formulation is deliberately general so that it encompasses as many distributions and models as possible. *We wish to be limited only by the assumption that the regression coefficients enter through a set of linear predictors*.

Given the covariates, the conditional distribution of the response is intended to be completely general.

More general $\Longrightarrow$ *more useful.*

VGLM Examples

1. GLMs $M = 1$

t

Distribution	Density/probability function $f(y)$	Range of y	VGAM family function
Gaussian	$(2\pi\sigma^2)^{-\frac{1}{2}} \exp\left\{-\frac{1}{2}(y-\mu)^2/\sigma^2\right\}$	$(-\infty, \infty)$	`gaussianff()`
Binomial	$\binom{A}{Ay} p^{Ay}(1-p)^{A(1-y)}$	$0(1/A)1$	`binomialff()`
Poisson	$\frac{\exp\{-\lambda\}\lambda^y}{y!}$	$0(1)\infty$	`poissonff()`
Gamma	$\frac{(k/\mu)^k y^{k-1} \exp\{-ky/\mu\}}{\Gamma(k)}$	$(0, \infty)$	`gammaff()`
Inverse Gaussian	$\left(\frac{\lambda}{2\pi y^3}\right)^{\frac{1}{2}} \exp\left\{-\frac{\lambda}{2\mu^2}\frac{(y-\mu)^2}{y}\right\}$	$(0, \infty)$	`inverse.gaussianff()`

Table : Summary of GLMs supported by VGAM. The known prior weight is A. These are incompatible with `glm()`.

2 Zero-inflated Poisson distribution

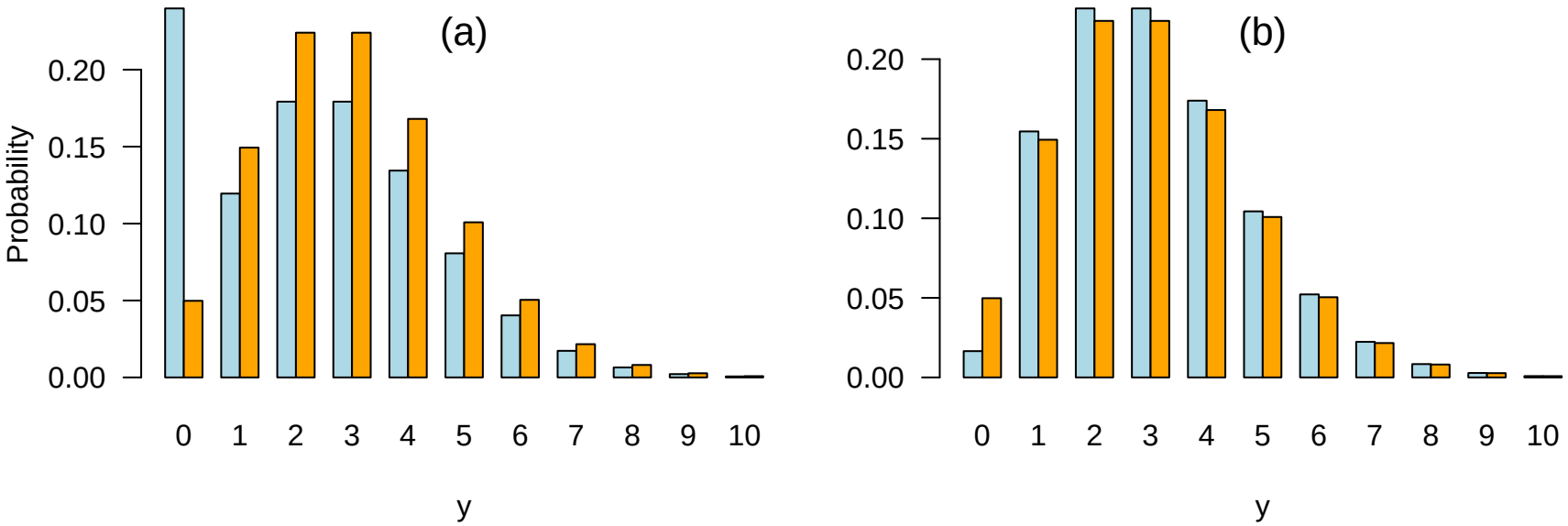


Figure : Probability functions of a (a) zero-inflated Poisson with $\phi = 0.2$, (b) *zero-deflated* Poisson with $\phi = -0.035$. Both are compared to their parent distribution which is a Poisson($\mu = 3$) in orange. Note: $\Pr(Y = y) = I(y = 0)\phi + (1 - \phi)e^{-\lambda}\lambda^y/y!$, $y = 0(1)\infty$, $\lambda > 0$.

3 Negative binomial distribution

For $y = 0, 1, 2, \ldots,$

$$f(y; \mu, k) = \binom{y + k - 1}{y} \left(\frac{\mu}{\mu + k}\right)^y \left(\frac{k}{k + \mu}\right)^k, \quad \mu > 0, \; k > 0.$$

Good choice:

$$\begin{aligned} \eta_1 &= \log \mu, \\ \eta_2 &= \log k. \end{aligned}$$

```
> vglm(y ~ x2 + x3 + ..., family = negbinomial(zero = NULL), ndata)
```

4 Bivariate logistic odds-ratio model

Data: (Y_1, Y_2) where $Y_j = 0$ or 1.

Examples

- $Y_1 = 1$ if left eye is blind, $Y_2 = 1$ if right eye is blind.
- $Y_1 = 1/0$ for presence/absence of cancer,
  $Y_2 = 1/0$ for presence/absence of diabetes.
- $Y_j = 1/0$ if Species j is present/absent.

Table : The coalminers data set from UK collieries. Note: $B =$ Breathlessness, $W =$ Wheeze, $1 =$ presence, $0 =$ absence.

t

Age group	$(B = 1, W = 1)$	$(B = 1, W = 0)$	$(B = 0, W = 1)$	$(B = 0, W = 0)$
20–24	9	7	95	1841
25–29	23	9	105	1654
30–34	54	19	177	1863
35–39	121	48	257	2357
40–44	169	54	273	1778
45–49	269	88	324	1712
50–54	404	117	245	1324
55–59	406	152	225	967
60–64	372	106	132	526

$$
\begin{aligned}
p_j &= \Pr(Y_j = 1), \quad \text{marginal probabilities} \\
p_{rs} &= \Pr(Y_1 = r, Y_2 = s), \quad r, s = 0, 1, \quad \text{joint probabilities} \\
\psi &= \frac{p_{00}\, p_{11}}{p_{01}\, p_{10}} \quad \text{(odds ratio).}
\end{aligned}
$$

Model:

$$\begin{aligned} \text{logit}\, p_j(\mathbf{x}) &= \eta_j(\mathbf{x}), \qquad j = 1, 2\,, \\ \log\ \psi(\mathbf{x}) &= \eta_3(\mathbf{x}). \end{aligned}$$

Recover p_{rs}'s from p_1, p_2 and ψ.

Q: Why not allow a probit or complementary log-log link?

Exchangeable data $\Longrightarrow$ constrain $\eta_1 = \eta_2$ (e.g., eyes), i.e.,

$$\begin{aligned} \texttt{cloglog}\, p_j(\mathbf{x}) &= \eta_1(\mathbf{x}), \qquad j = 1, 2\,, \\ \log\ \psi(\mathbf{x}) &= \eta_3(\mathbf{x}). \end{aligned}$$

Note:

$$\begin{pmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \end{pmatrix} = \sum_{k=1}^{p} \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \beta^*_{(1)k} \\ \beta^*_{(2)k} \end{pmatrix} x_k = \sum_{k=1}^{p} \begin{pmatrix} \beta^*_{(1)k} \\ \beta^*_{(1)k} \\ \beta^*_{(2)k} \end{pmatrix} x_k.$$

```
vglm(..., family = binom2.or("cloglog", exchangeable = TRUE))
```

Arguments such as `exchangeable`, `zero` and `parallel` create the $\mathbf{H}_k$ most conveniently.

5. Models for a categorical response i.e., $Y \in \{1, 2, \ldots, M+1\}$.

Y may be unordered (*nominal*) or ordered (*ordinal*).

Table : Period of exposure (years) and severity of pneumoconiosis amongst a group of coalminers.

t Exposure Time	Normal	Mild	Severe
5.8	98	0	0
15.0	51	2	1
21.5	34	6	3
27.5	35	5	8
33.5	32	10	9
39.5	23	7	8
46.0	12	6	10
51.5	4	2	5

(i) Multinomial logit model (nominal Y)

$$\Pr(Y = j|\mathbf{x}) = \frac{\exp\{\eta_j(\mathbf{x})\}}{\sum_{t=1}^{M+1} \exp\{\eta_t(\mathbf{x})\}}, \quad j = 1, \ldots, M+1.$$

For identifiability: $\eta_{M+1} \equiv 0$.

Equivalently,

$$\log\left(\frac{\Pr(Y=j|\mathbf{x})}{\Pr(Y=M+1|\mathbf{x})}\right) = \eta_j(\mathbf{x}), \quad j=1,\ldots,M+1.$$

```
> vglm(ymatrix ~ x2 + x3 + ..., family = multinomial, data = mdata)
```

(ii) Nonproportional odds model (ordinal Y)

$$\text{logit}\,\Pr(Y \leq j|\mathbf{x}) \quad = \quad \eta_j(\mathbf{x}), \qquad j = 1, \ldots, M. \tag{49}$$

Proportional odds model: constrain

$$\eta_j(\mathbf{x}) = \alpha_j + \eta(\mathbf{x})$$

(aka the *parallelism* assumption, which stops the probabilities from becoming negative or greater than 1).

```
> vglm(ymatrix ~ x2 + ..., family = cumulative(parallel = TRUE))
```

Other links (for $0 < p < 1$):

probit	$\Phi^{-1}(p)$,
complementary log-log	$\log\{-\log(1-p)\}$,
cauchit	$\tan(\pi(p - \frac{1}{2}))$.

So

```
vglm(ymatrix ~ x2 + ..., cumulative(link = probit, parallel = TRUE))
```

replaces `logit` in (49) by a `probit` link.

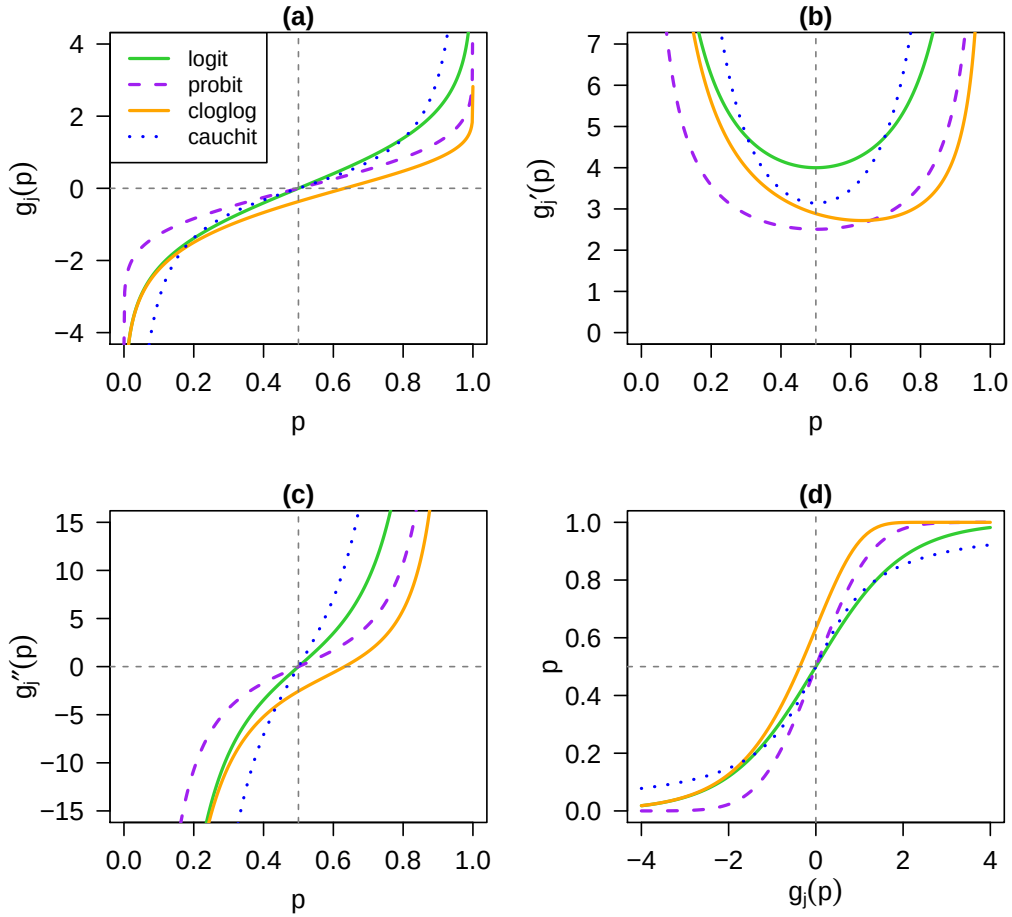


Figure : Some common link functions g_j for a probability. (a) $g_j(p)$; (b) $g_j'(p)$; (c) $g_j''(p)$; (d) $g_j^{-1}(p)$. Calls to (a)–(c) are of the form `link.function(p, deriv = d)` for `d = 0`, `1` and `2`.

VGLM Algorithm†

Models with log-likelihood

$$\ell(\boldsymbol{\beta}) = \sum_{i=1}^{n} \ell_i\{\eta_1(\mathbf{x}_i), \ldots, \eta_M(\mathbf{x}_i)\},$$

where $\eta_j = \boldsymbol{\beta}_j^T \mathbf{x}_i$. Then

$$\frac{\partial \ell}{\partial \boldsymbol{\beta}_j} = \sum_{i=1}^{n} \frac{\partial \ell_i}{\partial \eta_j} \mathbf{x}_i$$

and

$$\frac{\partial^2 \ell}{\partial \boldsymbol{\beta}_j \, \partial \boldsymbol{\beta}_k^T} = \sum_{i=1}^{n} \frac{\partial^2 \ell_i}{\partial \eta_j \, \partial \eta_k} \mathbf{x}_i \, \mathbf{x}_i^T.$$

Newton-Raphson algorithm

$$\boldsymbol{\beta}^{(a+1)} = \boldsymbol{\beta}^{(a)} + \mathcal{J}\left(\boldsymbol{\beta}^{(a)}\right)^{-1} \mathbf{U}\left(\boldsymbol{\beta}^{(a)}\right)$$

written in iteratively reweighted least squares (IRLS) form is

$$\begin{aligned}
\boldsymbol{\beta}^{(a+1)} &= \left(\mathbf{X}^T \mathbf{W} \mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{W} \mathbf{X} \boldsymbol{\beta}^{(a)} + \left(\mathbf{X}^T \mathbf{W} \mathbf{X}\right)^{-1} \mathbf{X}^T \mathbf{W} \mathbf{W}^{-1} \mathbf{u} \\
&= \left(\mathbf{X}_{\text{VLM}}^T \mathbf{W}^{(a)} \mathbf{X}_{\text{VLM}}\right)^{-1} \mathbf{X}_{\text{VLM}}^T \mathbf{W}^{(a)} \mathbf{z}^{(a)}.
\end{aligned}$$

Let $\mathbf{z} = (\mathbf{z}_1^T, \ldots, \mathbf{z}_n^T)^T$ and $\mathbf{u} = (\mathbf{u}_1^T, \ldots, \mathbf{u}_n^T)^T$, where $\mathbf{u}_i$ has jth element

$$(\mathbf{u}_i)_j = \frac{\partial \ell_i}{\partial \eta_j},$$

and $\mathbf{z}_i = \boldsymbol{\eta}(\mathbf{x}_i) + \mathbf{W}_i^{-1} \mathbf{u}_i$ (*adjusted dependent vector* or *pseudo-response*).

Also, $\mathbf{W} = \text{Diag}(\mathbf{W}_1, \ldots, \mathbf{W}_n)$, $\quad (\mathbf{W}_i)_{jk} = -\dfrac{\partial^2 \ell_i}{\partial \eta_j \, \partial \eta_k}$,

$\mathbf{X}_{\text{VLM}} = (\mathbf{X}_1^T, \ldots, \mathbf{X}_n^T)^T$, $\quad \mathbf{X}_i = \text{Diag}(\mathbf{x}_i^T, \ldots, \mathbf{x}_i^T) = \mathbf{I}_M \otimes \mathbf{x}_i^T$.
Then $\boldsymbol{\beta}^{(a+1)}$ is the solution to

$$\mathbf{z}^{(a)} \;=\; \mathbf{X}_{\text{VLM}} \, \boldsymbol{\beta}^{(a+1)} + \boldsymbol{\varepsilon}^{(a)}, \qquad \text{Var}(\boldsymbol{\varepsilon}^{(a)}) \;=\; \phi \, \mathbf{W}^{(a)^{-1}}.$$

Fisher scoring:

$$(\mathbf{W}_i)_{jk} \;=\; -\, E\left[\frac{\partial^2 \ell_i}{\partial \eta_j \, \partial \eta_k}\right]$$

usually results in slower convergence but is preferable because the *working weight matrices* are positive-definite over a larger parameter space.

Some Notes

1. `wz` computed in `@weight` is usually

$$(\mathbf{W}_i)_{jk} = -E\left(\frac{\partial^2 \ell_i}{\partial \eta_j \, \partial \eta_k}\right), \quad \text{sometimes} \quad -\frac{\partial^2 \ell_i}{\partial \eta_j \, \partial \eta_k}.$$

2. The following formulae are useful.

$$\begin{aligned}
\frac{\partial \ell}{\partial \eta_j} &= \frac{\partial \ell}{\partial \theta_j} \frac{\partial \theta_j}{\partial \eta_j}, \\
\frac{\partial^2 \ell}{\partial \eta_j^2} &= \frac{\partial \ell}{\partial \theta_j} \frac{\partial^2 \theta_j}{\partial \eta_j^2} + \left(\frac{\partial \theta_j}{\partial \eta_j}\right)^2 \frac{\partial^2 \ell}{\partial \theta_j^2}, \\
\frac{\partial^2 \ell}{\partial \eta_j \, \partial \eta_k} &= \left\{ \frac{\partial^2 \ell}{\partial \theta_j \, \partial \theta_k} - \frac{\partial \ell}{\partial \theta_k} \frac{\partial \theta_k}{\partial \eta_k} \frac{\partial^2 \eta_k}{\partial \theta_j \, \partial \theta_k} \right\} \frac{\partial \theta_j}{\partial \eta_j} \frac{\partial \theta_k}{\partial \eta_k}, \quad j \neq k,
\end{aligned}$$

3. The 'big' model matrix (`model.matrix(fit, type = "vlm")`) is

$$\begin{aligned}\mathbf{X}_{\text{VLM}} &= \left((\mathbf{X}\,\mathbf{e}_1) \otimes \mathbf{H}_1 \;\middle|\; (\mathbf{X}\,\mathbf{e}_2) \otimes \mathbf{H}_2 \;\middle|\; \cdots \;\middle|\; (\mathbf{X}\,\mathbf{e}_p) \otimes \mathbf{H}_p \right) \\ &= \mathbf{X}_{\text{LM}} \otimes \mathbf{I}_M \quad \text{with trivial constraints.}\end{aligned}$$

With the `xij` facility,

$$\mathbf{X}_{\text{VLM}} = \begin{pmatrix} \mathbf{X}^{\#}_{(11)}\mathbf{H}_1 & \cdots & \mathbf{X}^{\#}_{(1p)}\mathbf{H}_p \\ \vdots & & \vdots \\ \mathbf{X}^{\#}_{(n1)}\mathbf{H}_1 & \cdots & \mathbf{X}^{\#}_{(np)}\mathbf{H}_p \end{pmatrix}. \tag{50}$$

Inference

Here are a few results useful for inference.

1. MLEs are *asymptotic normal*:

$$\widehat{\boldsymbol{\theta}}_n \xrightarrow{\mathcal{D}} N_p(\boldsymbol{\theta}_*,\ \boldsymbol{\mathcal{I}}_E^{-1}(\boldsymbol{\theta}_*)). \tag{51}$$

2. An approximate $100(1-\alpha)\%$ confidence interval for θ_j is given by

$$\widehat{\theta}_j \pm z(\alpha/2)\,\mathrm{SE}(\widehat{\theta}_j). \tag{52}$$

Wald statistic: under $H_0 : \theta_j = 0$, `summary()` prints out

$$z_0 = \frac{\widehat{\theta}_j - 0}{\sqrt{\widehat{\mathrm{Var}}(\widehat{\theta}_j)}} = \frac{\widehat{\theta}_j}{\mathrm{SE}(\widehat{\theta}_j)}$$

(treated as a Z-statistic—or a t-ratio for LMs).

3. *Likelihood Ratio Test (LRT)* for *2 nested* models:
   Suppose $\mathcal{M}_1 \subseteq \mathcal{M}_2$, i.e., model $\mathcal{M}_1$ is a subset or special case of $\mathcal{M}_2$.

   Then the *likelihood ratio test statistic*

$$-2 \log \lambda \;=\; 2 \log L(\widehat{\boldsymbol{\theta}}_{\mathcal{M}_2}; \mathbf{y}) - 2 \log L(\widehat{\boldsymbol{\theta}}_{\mathcal{M}_1}; \mathbf{y}) \;\to\; \chi^2_\nu$$

   where $\nu = \dim(\mathcal{M}_2) - \dim(\mathcal{M}_1)$, the difference in the number of parameters in the models.

   Use `lrtest(complex.model, simpler.model)`.

   For GLMs, the LRT is aka the *deviance test*.

4. *Delta method* for obtaining approximate SEs of functions of the parameter. Let $\phi = g(\boldsymbol{\theta})$ be some function of the parameter. Then

$$g(\widehat{\boldsymbol{\theta}}_n) - g(\boldsymbol{\theta}_*) \xrightarrow{\mathcal{D}} N_p\left(\mathbf{0},\ \frac{\partial g(\boldsymbol{\theta}_*)}{\partial \boldsymbol{\theta}^T}\, \boldsymbol{\mathcal{I}}_E^{-1}(\boldsymbol{\theta}_*)\, \frac{\partial g(\boldsymbol{\theta}_*)}{\partial \boldsymbol{\theta}}\right). \tag{53}$$

Equivalently (all quantities are computed at the MLE), for large n,

$$\begin{aligned}
\mathrm{SE}(\widehat{\phi}) &\approx \left\{ \sum_{j=1}^{p} \sum_{k=1}^{p} \frac{\partial g}{\partial \theta_j} \frac{\partial g}{\partial \theta_k}\, \widehat{v}_{jk} \right\}^{\frac{1}{2}} = \left\{ \frac{\partial g(\widehat{\boldsymbol{\theta}})}{\partial \boldsymbol{\theta}^T}\, \widehat{\mathrm{Var}}(\widehat{\boldsymbol{\theta}})\, \frac{\partial g(\widehat{\boldsymbol{\theta}})}{\partial \boldsymbol{\theta}} \right\}^{\frac{1}{2}} \\
&= \left| \frac{\mathrm{d}g}{\mathrm{d}\theta} \right| \sqrt{\widehat{v}_{11}}, \quad \text{when } p = 1.
\end{aligned}$$

Use `vcov(vglmObject, untransform = TRUE)` for simple intercept-only models.

5. *Cramér-Rao inequality* (simplified version): under regularity conditions and i.i.d. conditions, for all n and unbiased estimators $\widehat{\boldsymbol{\theta}}_n$,

$$\mathrm{Var}(\widehat{\boldsymbol{\theta}}_n) - \boldsymbol{\mathcal{I}}_E^{-1}(\boldsymbol{\theta}) \tag{54}$$

is positive-semidefinite. For the 1-parameter case:

$$\frac{1}{n\mathcal{I}_{E1}(\theta)} \;=\; \mathcal{I}_E^{-1}(\theta) \;\leq\; \mathrm{Var}(\widehat{\theta}_n). \tag{55}$$

That is, the inverse of the EIM (known as the *Cramér-Rao lower bound*; CRLB) is a lower bound for the variance of an unbiased estimator.

The VGAM Package for R

Written in S, its central core are the functions `vglm()`, `vgam()` and `rrvglm()`.

Generic functions include `coef()`, `fitted()`, `plot()`, `predict()`, `print()`, `resid()`, `summary()`. Others are `lvplot()`, `Coef()`, `df.residual()`, `logLik()`, `vcov()`.

Plus 150+ of VGAM family functions.

Modular construction, flexible, easy to use and useful.
Runs under Version 4 of the S language (Chambers, 1998, 2008) in **R**.
Can install the VGAM package in **R** by typing

```
> install.packages("VGAM")
> install.packages("VGAMdata")
```

The Central Functions of VGAM

`vglm()`	Vector generalized linear models.
`vgam()`	Vector generalized additive models.
`rrvglm()`	Reduced-rank vector generalized linear models.
`cqo()`	Constrained quadratic (Gaussian) ordination (QRR-VGLM).
`cao()`	Constrained additive ordination (RR-VGAM).

Others:

`vlm()`	Vector linear models.
`grc()`	Goodman's RC(R) model.
`rcim()`	Row-column interaction models (not complete).

```
Package: VGAM
Version: 1.0-4
Date: 2017-07-24
Title: Vector Generalized Linear and Additive Models
Author: Thomas W. Yee <t.yee@auckland.ac.nz>
Maintainer: Thomas Yee <t.yee@auckland.ac.nz>
Depends: R (>= 3.4.0), methods, stats, stats4, splines
Suggests: VGAMdata, MASS, mgcv
Description: An implementation of about 6 major classes of
    statistical regression models. At the heart of it are the
    vector generalized linear and additive model (VGLM/VGAM)
    classes, and the book "Vector Generalized Linear and
    Additive Models: With an Implementation in R" (Yee, 2015)
    <DOI: 10.1007/978-1-4939-2818-7>
    gives details of the statistical framework and VGAM package.
    Currently only fixed-effects models are implemented,
    i.e., no random-effects models. Many (150+) models and
    distributions are estimated by maximum likelihood estimation
    (MLE) or penalized MLE, using Fisher scoring. VGLMs can be
    loosely thought of as multivariate GLMs. VGAMs are data-driven
    VGLMs (i.e., with smoothing). The other classes are RR-VGLMs
    (reduced-rank VGLMs), quadratic RR-VGLMs, reduced-rank VGAMs,
    RCIMs (row-column interaction models)---these classes perform
    constrained and unconstrained quadratic ordination (CQO/UQO)
```

```
    models in ecology, as well as constrained additive ordination
    (CAO). Note that these functions are subject to change;
    see the NEWS and ChangeLog files for latest changes.
License: GPL-2 | GPL-3
URL: https://www.stat.auckland.ac.nz/~yee/VGAM
```

See the DESCRIPTION file.

Table : VGAM generic functions applied to a model called `fit`.

Function	Value
`coef(fit)`	$\widehat{\boldsymbol{\beta}}^*$
`coef(fit, matrix = TRUE)`	$\widehat{\mathbf{B}}$
`constraints(fit, type = "lm")`	$\mathbf{H}_k,\ k = 1, \ldots, p$
`deviance(fit)`	Deviance $D = \sum_{i=1}^{n} d_i$
`fitted(fit)`	$\widehat{\mu}_{ij}$ usually
`logLik(fit)`	Log-likelihood $\sum_{i=1}^{n} w_i \, \ell_i$
`model.matrix(fit, type = "lm")`	LM model matrix $(n \times p)$
`model.matrix(fit, type = "vlm")`	VLM model matrix $\mathbf{X}_{\text{VLM}}$
`predict(fit)`	$\widehat{\eta}_{ij}$

Table : VGAM generic functions applied to a model called `fit`.

Function	Value
`predict(fit, type = "response")`	$\widehat{\mu}_{ij}$ usually
`resid(fit, type = "response")`	$y_{ij} - \widehat{\mu}_{ij}$
`resid(fit, type = "deviance")`	$\text{sign}(y_i - \hat{\mu}_i)\sqrt{d_i}$
`resid(fit, type = "pearson")`	$\mathbf{W}_i^{-\frac{1}{2}}\,\mathbf{u}_i$
`resid(fit, type = "working")`	$\mathbf{z}_i - \boldsymbol{\eta}_i = \mathbf{W}_i^{-1}\,\mathbf{u}_i$
`vcov(fit)`	$\widehat{\text{Var}}(\widehat{\boldsymbol{\beta}})$
`weights(fit, type = "prior")`	w_i (`weights` argument)
`weights(fit, type = "working")`	$w_i\mathbf{W}_i$ (in matrix-band format)

The VGAM package employs several feature to make the software more robust, e.g.,

- Parameter link functions, e.g.,
  - $\log \theta$ for $\theta > 0$,
  - $\text{logit}\ \theta$ for $0 < \theta < 1$.
  - $\log(\theta - 1)$ for $\theta > 1$.
- *Half-step sizing*.
- Good initial values, e.g., self-starting VGAM family functions.
- Numerical linear algebra based on orthogonal methods, e.g., QR method in LINPACK. Yet to do: use LAPACK.
- B-splines, not the Reinsch algorithm.

Some Computational and Implementational Details†

- Is S4 object-orientated and very modular—simply have to write a VGAM "family function".

```
> args(vglm.control)

function (checkwz = TRUE, Check.rank = TRUE, Check.cm.rank = TRUE,
    criterion = names(.min.criterion.VGAM), epsilon = 1e-07,
    half.stepsizing = TRUE, maxit = 30, noWarning = FALSE, stepsize = 1,
    save.weights = FALSE, trace = FALSE, wzepsilon = .Machine$double.eps^0.75,
    xij = NULL, ...)
NULL
```

Some VGLM/VGAM Examples

The following are some simple VGAM examples.

Bivariate Odds Ratio Model †

Look at Cat and Dog Pet Ownership in `xs.nz`

(a)

```
> with(xs.nz, table(cat, dog))

   dog
cat    0    1
  0 3734 1376
  1 3230 1879
```

Try mosaic plots.

```
> mosaicplot(with(xs.nz, table(cat, dog)), col = hcl(c(240, 120)))
```

This produces the figure in Slide 140.

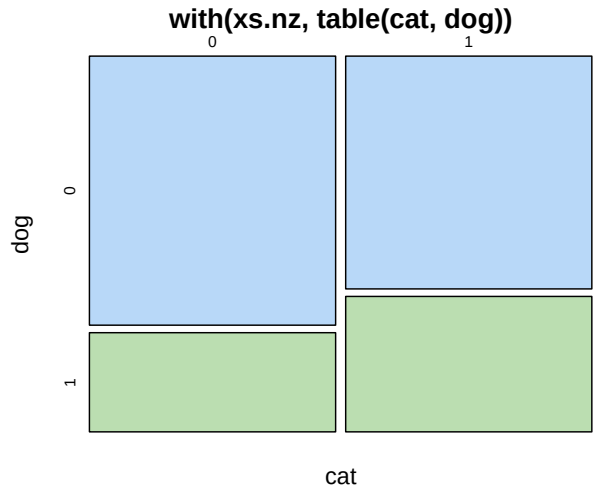


Figure : Mosaic plot of `cat` versus `dog` for the `xs.nz` data frame. The value 0 means none, 1 means yes, and the jitter has mean 0.

(b) The odds ratio is estimated to be

```
> (mytab <- with(xs.nz, table(cat, dog)))

   dog
cat    0    1
  0 3734 1376
  1 3230 1879

> (oratio <- prod(diag(mytab)) / (mytab[1, 2] * mytab[2, 1]))

[1] 1.579
```

The estimate is > 1, $\Rightarrow$ ownership of cats and dogs has a positive association. It is statistically significant.

We say that the odds of owning a cat for a household with a dog is about $\widehat{\psi} \approx 1.58$ times the odds of owning a cat for a household without a dog.

(c) Let's focus on European-type women.

```
> women.eth0 <- subset(xs.nz, sex == "F" &
                               ethnicity == "European")
```

To fit a bivariate odds-ratio model try:

```
> women.eth0.catdog <-
    subset(women.eth0, !is.na(age) &
                       !is.na(cat) &
                       !is.na(dog))
> cd.fit <- vglm(cbind(cat, dog) ~ bs(age, df = 5),
                 binom2.or, data = women.eth0.catdog)
```

```
> ooo <- with(women.eth0.catdog, order(age))
> mycol <- c("tomato", "green", "blue", "purple")
> mylty <- c(1, 1, 2, 3)
> with(women.eth0.catdog,
       matplot(age[ooo], fitted(cd.fit)[ooo, ],
               type = "l", col = mycol, las = 1, lwd = 2,
               xlab = "Age", lty = mylty,
               ylab = "Fitted joint probabilities"))
>
> legend("topleft", c("No cat or dog", "Dog only", "Cat only", "Cat and dog"),
         lty = mylty, lwd = 2, col = mycol)
```

This gives the figure on Slide 144.

The probability of having cats and dogs at home occurs the most at ages 40–50: this probably corresponds with parents of teenagers who like furry animals in the house!

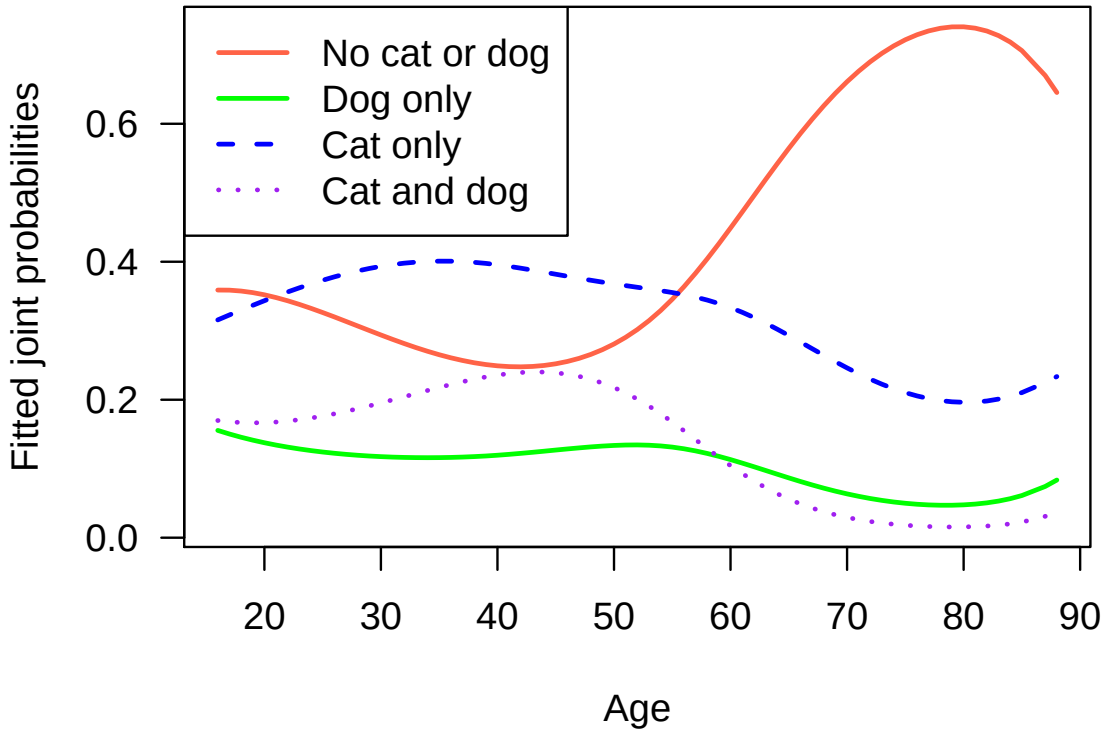


Figure : Bivariate odds-ratio model fitted to $Y_1 =$ `cat` and $Y_2 =$ `dog` versus `age` in female European-type subset of the `xs.nz` data frame.

To show how bad not using smoothing is, try:

```
> cd.fit.linear <- vglm(cbind(cat, dog) ~ age,
                        binom2.or, data = women.eth0.catdog)
```

```
> with(women.eth0.catdog,
       matplot(age[ooo], fitted(cd.fit.linear)[ooo, ],
               type = "l", col = mycol, las = 1, lwd = 2,
               xlab = "Age", lty = mylty,
               ylab = "Fitted joint probabilities"))
>
> legend("topleft", c("No cat or dog", "Dog only", "Cat only", "Cat and dog"),
         lty = mylty, lwd = 2, col = mycol)
```

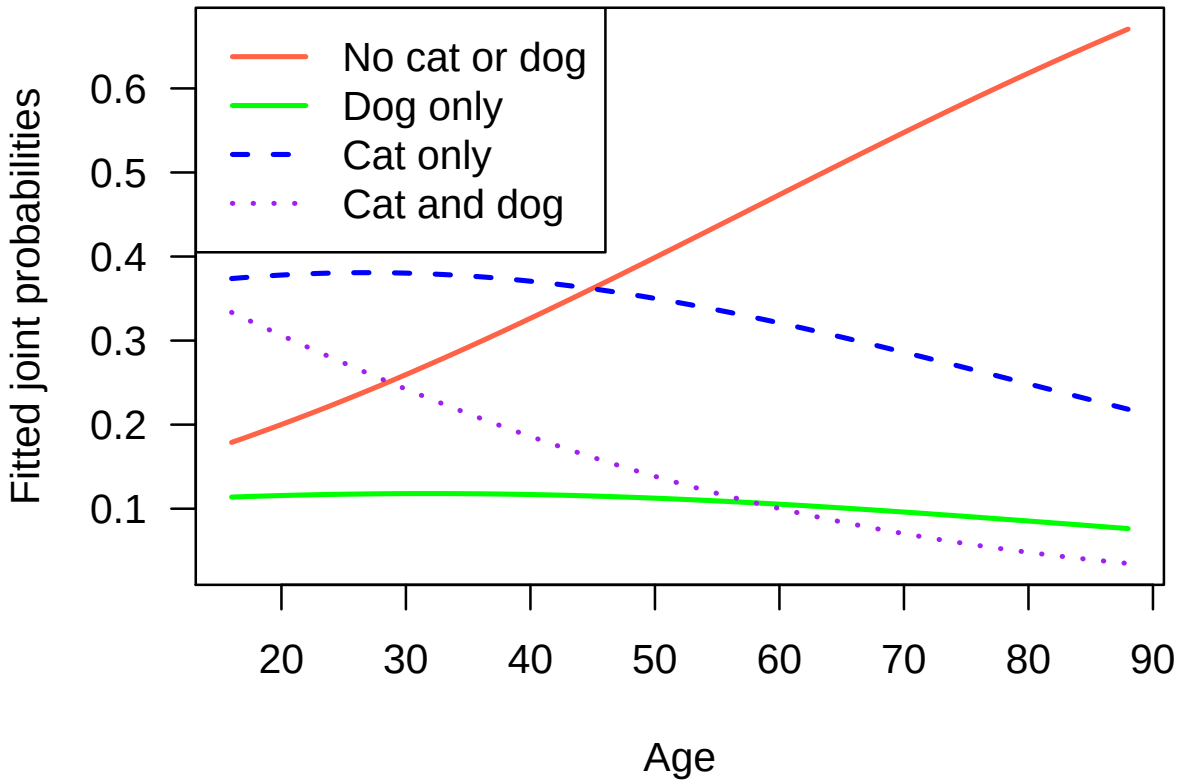


Figure : Linear fit version of the plot on Slide 144.

VGAMs

VGAMs allow additive-model extensions to all η_j in a VGLM, i.e., from

$$\eta_j(\mathbf{x}) \;=\; \beta_{(j)1}\, x_1 + \cdots + \beta_{(j)p}\, x_p \;=\; \boldsymbol{\beta}_j^T \mathbf{x}$$

to M additive predictors:

$$\eta_j(\mathbf{x}) \;=\; f_{(j)1}(x_1) + \cdots + f_{(j)p}(x_p),$$

a sum of arbitary smooth functions. Equivalently,

$$\begin{aligned} \boldsymbol{\eta}(\mathbf{x}) &= \mathbf{f}_1(x_1) + \cdots + \mathbf{f}_p(x_p) \\ &= \mathbf{H}_1\, \mathbf{f}_1^*(x_1) + \cdots + \mathbf{H}_p\, \mathbf{f}_p^*(x_p) \end{aligned} \tag{56}$$

for *constraint matrices* $\mathbf{H}_k$ (default: $\mathbf{H}_k = \mathbf{I}_M$).

- $\mathbf{H}_1, \ldots, \mathbf{H}_p$ are known and of full-column rank,

- $\mathbf{f}_k^* = (f_{(1)k}^*(x_k), f_{(2)k}^*(x_k), \ldots)^T$ contains possibly a reduced set of component functions.

Starred quantities are estimated.

The $\mathbf{f}_k^*$ are centered for identifiability.

Good for exploratory data analysis. And to find transformations that convert the fit to a VGLM.

Some Examples

The following are some simple examples fitting VGAMs.

Example 1: Hunua Tree Species

Let's fit 2 simultaneous logistic regressions: 2 species' presence/absence versus $X_2 =$ `altitude`. Data is from 392 sites from the Hunua forest.

- `agaaus` is *Agathis australis*, better known as "Kauri".
- `kniexc` is *Knightia excelsa*, or "Rewarewa".

```
> fit2 <- vgam(cbind(agaaus, kniexc) ~ s(altitude, df = c(2, 3)),
               binomialff(multiple.responses = TRUE), data = hunua)
> round(coef(fit2, matrix = TRUE), dig = 4)  # Largely uninterpretable

                          logit(E[agaaus]) logit(E[kniexc])
(Intercept)                         -1.302          -0.0721
s(altitude, df = c(2, 3))            0.000           0.0027
```

The output to `coef()` is not really interpretable; they are the coefficients to the linear part of the fit.

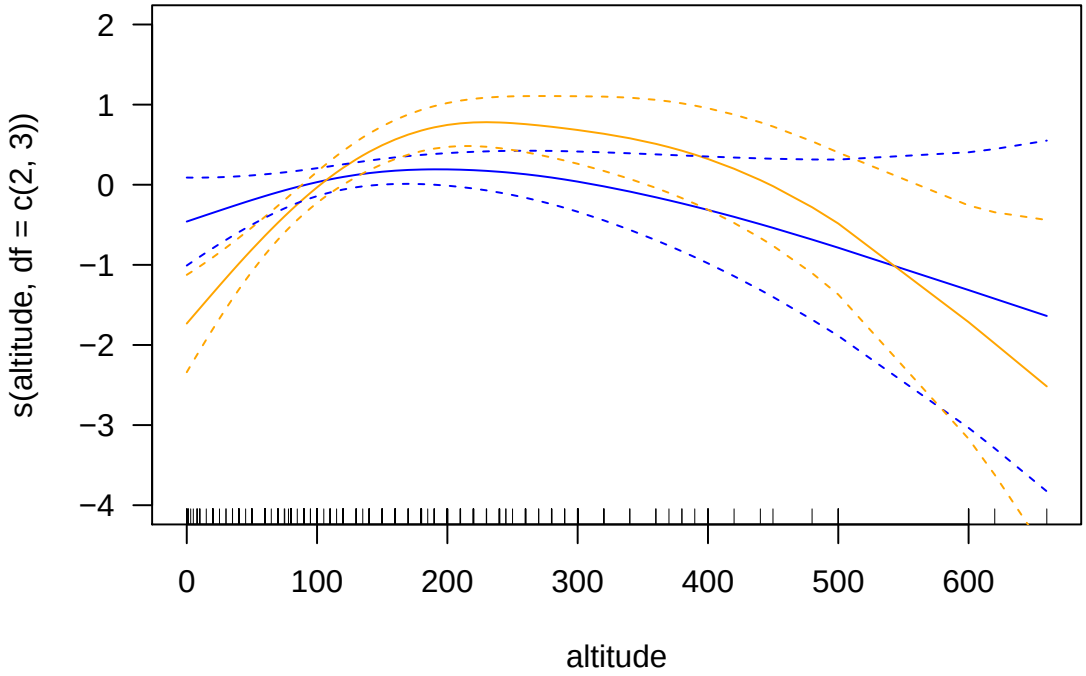


Figure : Two nonparametric logistic regressions fitted as a VGAM. Blue is Kauri, orange is Rewarewa.

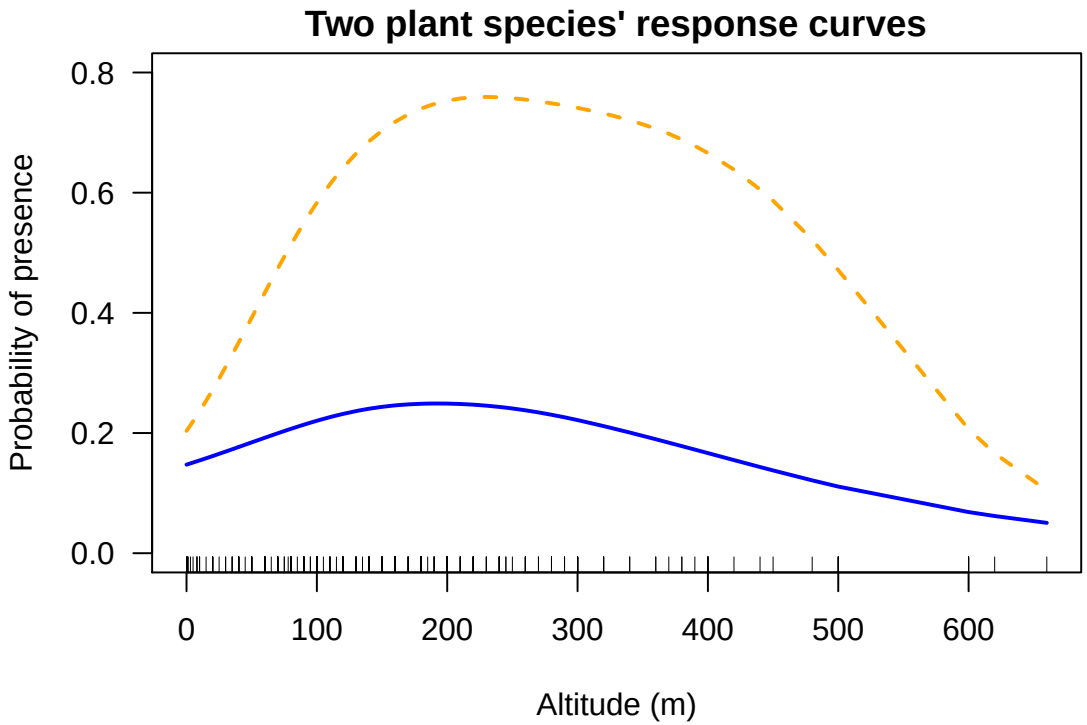


Figure : Two nonparametric logistic regressions fitted as a VGAM, for `hunua`.

P-splines

Let's try the more modern P-VGAMs.

```
> Fit2 <- vgam(cbind(agaaus, kniexc) ~ sm.ps(altitude),
               binomialff(multiple.responses = TRUE), data = hunua)
> round(coef(Fit2, matrix = TRUE), dig = 4)  # Largely uninterpretable

                 logit(E[agaaus]) logit(E[kniexc])
(Intercept)               -1.3861          0.3655
sm.ps(altitude)2          -0.2313         -0.6744
sm.ps(altitude)3           2.6319          3.3511
sm.ps(altitude)4           2.2208          3.5421
sm.ps(altitude)5           1.1028          2.0667
sm.ps(altitude)6           0.0560          1.0172
sm.ps(altitude)7          -0.7838          0.2039
sm.ps(altitude)8          -1.8049         -1.2968
sm.ps(altitude)9          -2.8763         -3.2024
sm.ps(altitude)10         -3.9368         -5.1085
```

The result is a slightly smoother version of the plot on Slide 151.

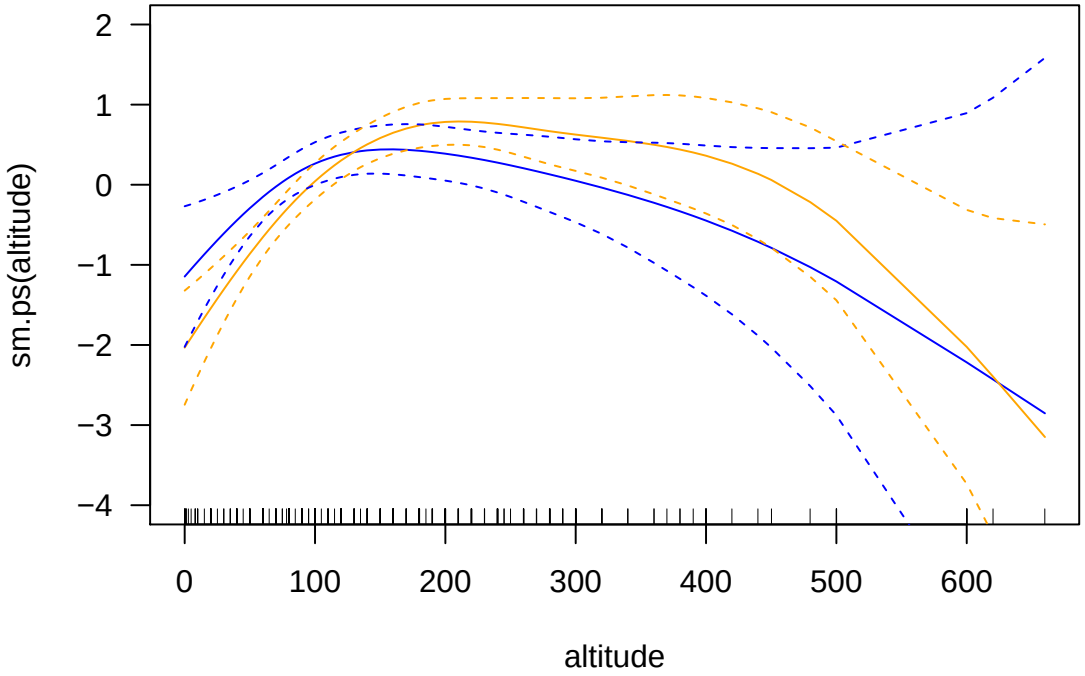


Figure : PS-VGAMs: two nonparametric logistic regressions fitted as a VGAM. Blue is Kauri, orange is Rewarewa.

Example 2: Cats and Dogs Revisited

Let's re-fit VGAMs to the cat-dog data, this time with O-splines and allow the odds ratio to be more flexible.

```
> women.eth0 <- subset(xs.nz, sex == "F" &
                               ethnicity == "European")
> women.eth0.cd <- subset(women.eth0, !is.na(age) &
                           !is.na(cat) & !is.na(dog))
>
> cd.fit <- vgam(cbind(cat, dog) ~ s(age, df = c(4, 4, 2)),
                 binom2.or(zero = NULL),
                 data = women.eth0.cd)
```

Plot the component functions and scale the y-axis to be comparable

```
> plot(cd.fit, se = TRUE, scol = "limegreen", lcol = "blue", scale = 4)
```

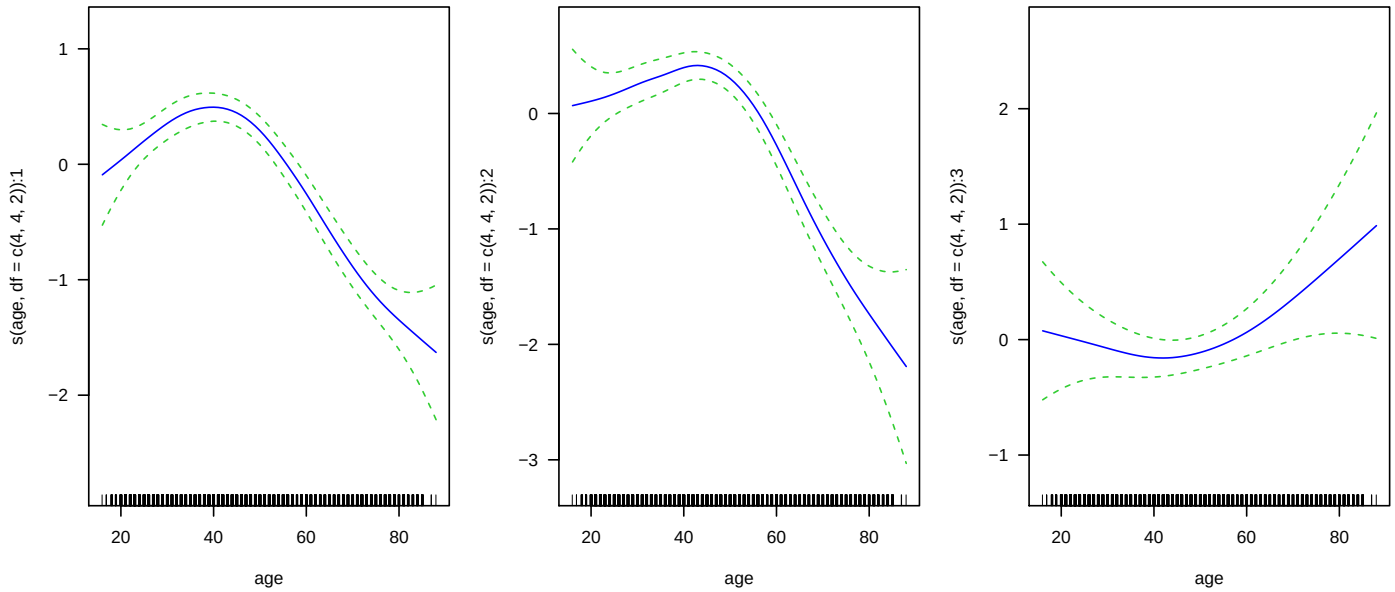


Figure : Two nonparametric logistic regressions fitted as a VGAM.

Q: are the component functions nonlinear?

```
> summary(cd.fit)@anova

                          Df Npar Df Npar Chisq    P(Chi)
(Intercept):1              1      NA          NA        NA
(Intercept):2              1      NA          NA        NA
(Intercept):3              1      NA          NA        NA
s(age, df = c(4, 4, 2)):1  1     3.0      77.221 1.131e-16
s(age, df = c(4, 4, 2)):2  1     2.8      68.806 5.489e-15
s(age, df = c(4, 4, 2)):3  1     0.9       5.847 1.267e-02
```

The plot of fitted probabilities here is better than with regression splines.

```
> ooo <- with(women.eth0.cd, order(age))
> mycol <- c("tomato", "green", "blue", "purple"); mylty <- c(1, 1, 2, 3)
> with(women.eth0.cd,
       matplot(age[ooo], fitted(cd.fit)[ooo,],
               type = "l", col = mycol, las = 1, lwd = 2,
               xlab = "Age", lty = mylty, ylab = "Fitted joint probabilities"))
> legend("topleft", c("No cat or dog", "Dog only", "Cat only", "Cat and dog"),
         lty = mylty, lwd = 2, col = mycol)
```

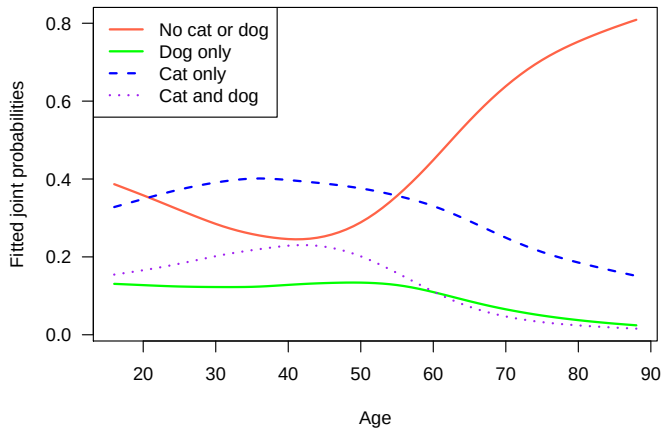


Figure : Fitted values for the O-spline GAM for cat-dog data.

Let's try a a parametric replacement.

```
> Hlist <- list("(Intercept)" = diag(3),
                "bs(age, degree = 1, knot = 40)" = rbind(1, 0, 0),
                "bs(age, degree = 1, knot = 50)" = rbind(0, 1, 0),
                "poly(age, 2)" = rbind(0, 0, 1))
>
> cd.fit2 <- vglm(cbind(cat, dog) ~
                  bs(age, degree = 1, knot = 40) +
                  bs(age, degree = 1, knot = 50) +
                  poly(age, 2),
                  binom2.or(zero = NULL), data = women.eth0.cd,
                  constraints = Hlist)
>
> # Compare them separately
> par(mfrow = c(2, 3))
> plot(cd.fit, se = TRUE, scol = "darkorange", lcol = "blue", scale = 4)
> plot(as(cd.fit2, "vgam"), se = TRUE, scol = "darkorange", lcol="blue",
       scale = 4)
```

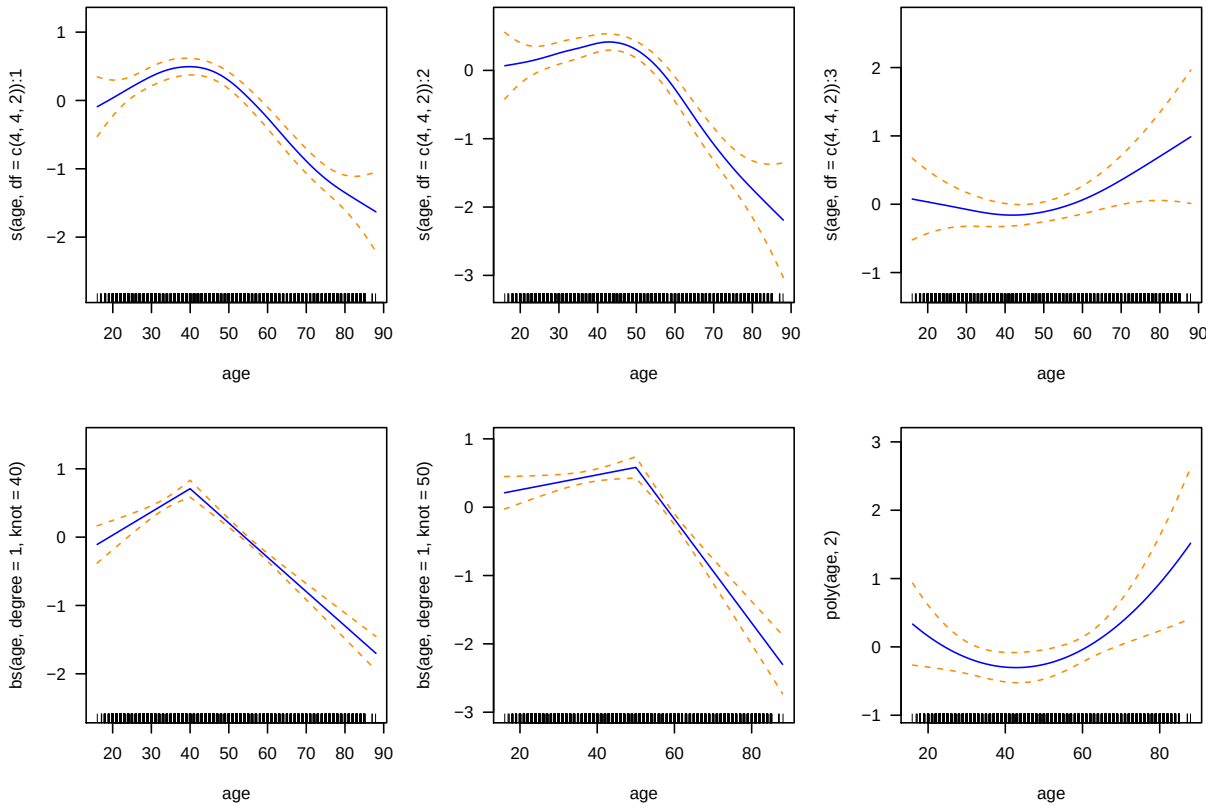


Figure : Fitted component functions.

But its better to overlay them. Note: a `vglm()` fit can be plotted with `plot()` after coercing it into a `"vgam"` object—useful with regression splines.

```
> for (ii in 1:3) {
    plot(cd.fit, which.cf = ii,
         se = TRUE, scol = "darkorange", lcol = "blue", scale = 3.5)
    plot(as(cd.fit2, "vgam"), which.term = ii, raw = TRUE,
         add = TRUE, overlay = TRUE,
         se = TRUE, scol = "purple", lcol = "black", scale = 3.5)
  }
```

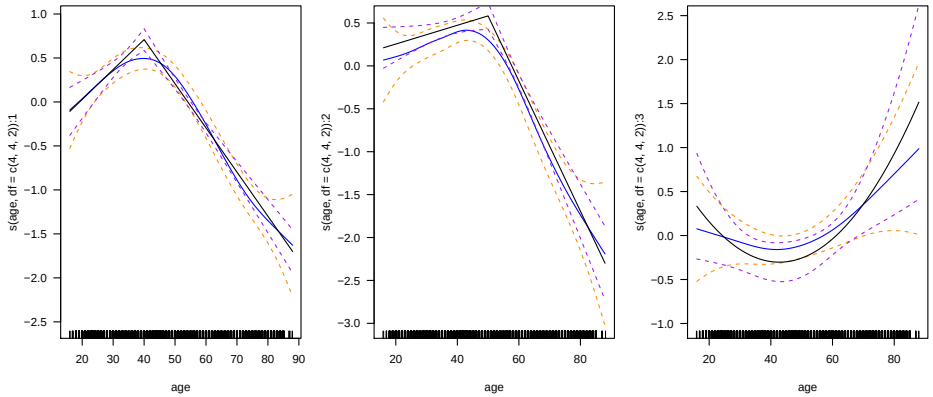


Figure : Fitted component functions.

Is the parametric fit okay? Approximate inference:

```
> pchisq(2 * (logLik(cd.fit) - logLik(cd.fit2)),
         df = df.residual(cd.fit2) - df.residual(cd.fit),
         lower.tail = FALSE)

[1] 0.1076
```

```
> df.residual(cd.fit)  # Check degrees of freedom

[1] 7694

> # Formula is n * M - M - sum(nonlinear df) because
> # there are M intercepts
> nrow(women.eth0.cd) * npred(cd.fit) - npred(cd.fit) - sum(c(4, 4, 2))

[1] 7694
```

P-splines

Let's try the more modern PS-VGAMs.

```
> cd.psvgam <- vgam(cbind(cat, dog) ~ sm.ps(age),
                    binom2.or(zero = NULL),
                    data = women.eth0.cd)
> plot(cd.psvgam, se = TRUE, scol = "limegreen", lcol = "blue", scale = 4)
```

These plots look similar to the ones on Slide 156.

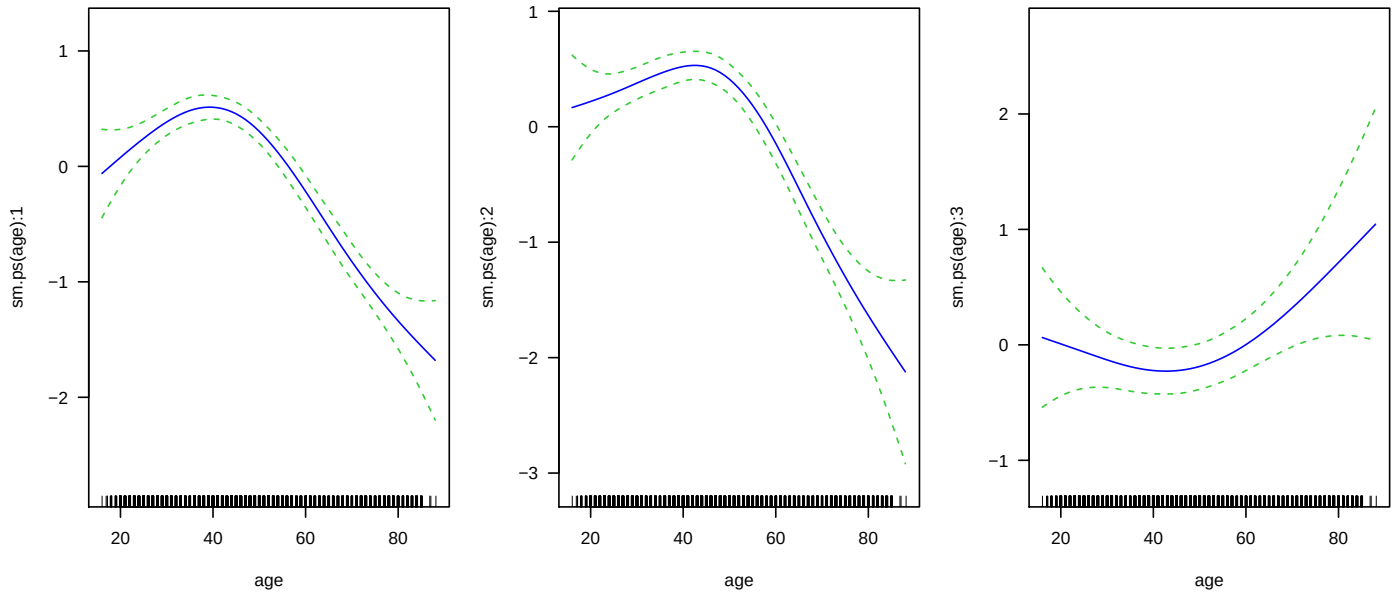


Figure : PS-VGAMs: two nonparametric logistic regressions fitted as a VGAM.

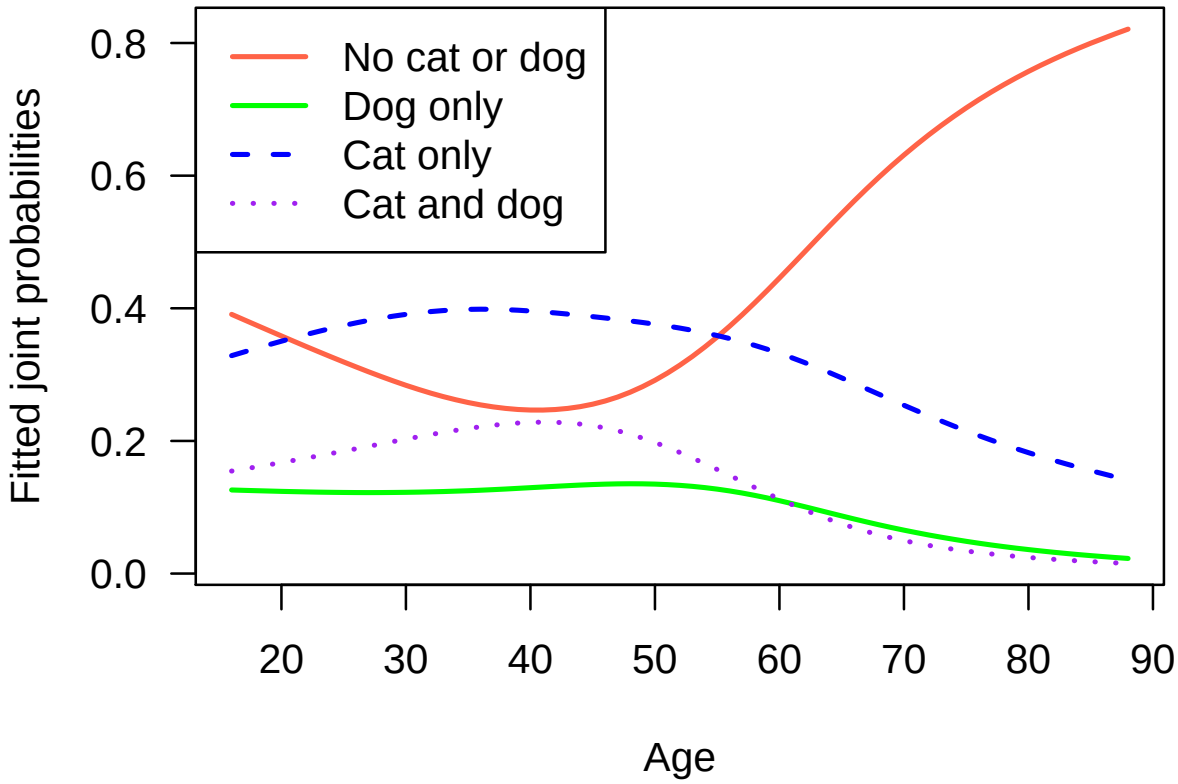


Figure : PS-VGAM fit version of the plot on Slide 144.

Example 3: Education Level and POMs

Let's look at `educ` in `xs.nz`, the highest education level as determined by age, ethnicity and gender.

```
> # Remove NAs and put into a smaller data frame.
> sxs.nz <- na.omit(xs.nz[, c("educ", "age", "ethnicity", "sex")])
> with(xs.nz, is.ordered(educ))

[1] TRUE

> with(xs.nz, is.factor(ethnicity))

[1] TRUE

> sxs.nz <- transform(sxs.nz, Sex = as.numeric(sex))
>
> edu.pom1 <- vgam(educ ~ s(age) + ethnicity + sex,
                   propodds, data = sxs.nz)
```

```
> plot(edu.pom1, se = TRUE, control = plotvgam.control(lcol = "blue",
      scol = "orange", rcol = "green"))
```

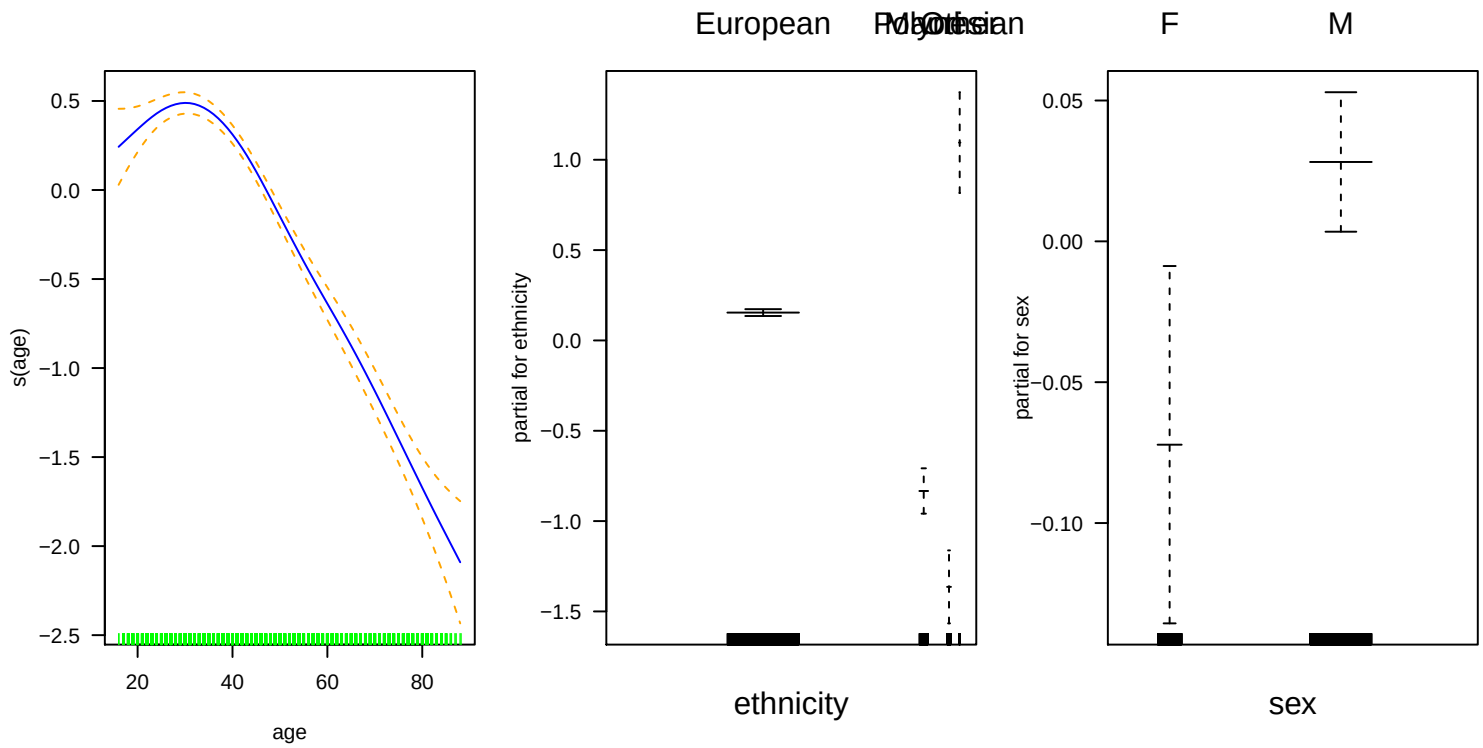


Figure : Fitted component functions.

Let's fit a model using regression splines

```
> sxs.nz <- transform(sxs.nz, log.age = log(age))  # To increase symmetry
> edu.pom2 <- vglm(educ ~ bs(log.age, df = 4) + ethnicity + sex,
                   propodds, data = sxs.nz)
```

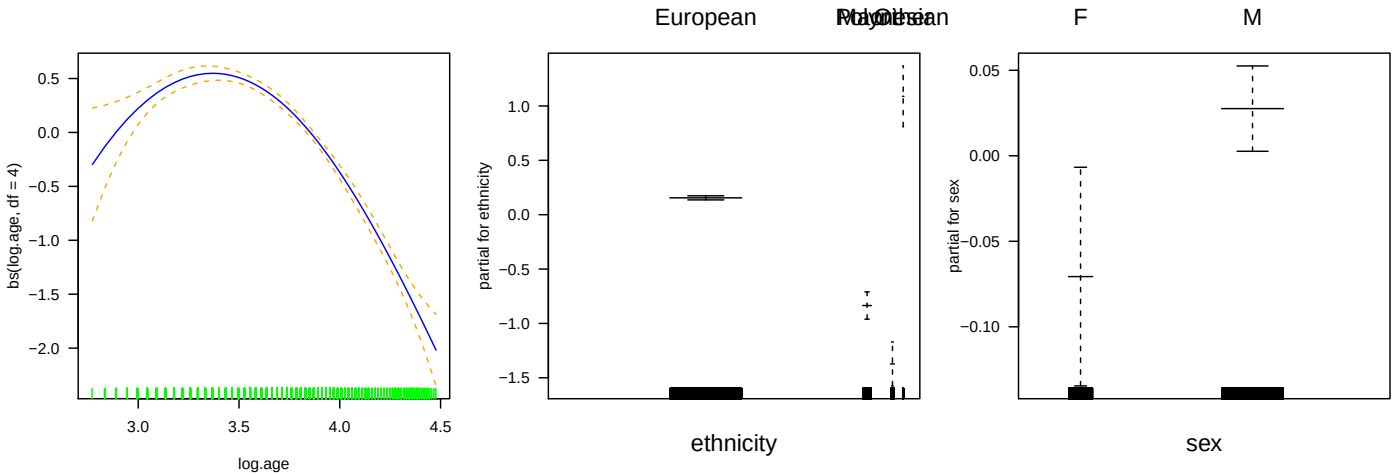


Figure : Fitted component functions.

It looks quadratic wrt `log.age` so let's try

```
> edu.pom3 <- vglm(educ ~ poly(log.age, 2) + ethnicity + sex,
                   propodds, data = sxs.nz)
```

We can conduct a likelihood ratio test to see whether the quadratic model is okay:

```
> pchisq(2 * (logLik(edu.pom2) - logLik(edu.pom3)),
         df = df.residual(edu.pom3) - df.residual(edu.pom2),
         lower.tail = FALSE)

[1] 0.6045
```

Yes, the quadratic model looks okay.

Let's compare them...

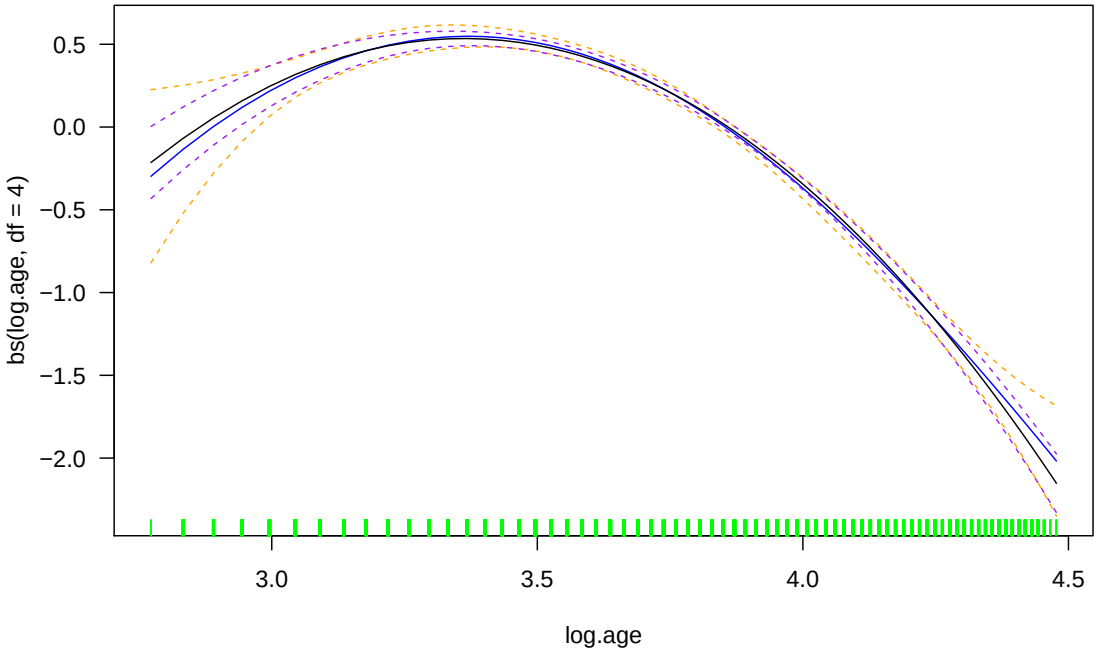


Figure : Fitted component functions.

They look very similar!

With regression splines inference is better.

```
> coef(summary(edu.pom3))

                    Estimate Std. Error z value   Pr(>|z|)
(Intercept):1        3.29910    0.05895  55.969  0.000e+00
(Intercept):2       -0.52613    0.03863 -13.621  2.989e-42
(Intercept):3       -1.84605    0.04329 -42.639  0.000e+00
poly(log.age, 2)1  -51.15534    2.16240 -23.657 1.006e-123
poly(log.age, 2)2  -32.64882    2.12460 -15.367  2.724e-53
ethnicityMaori      -0.98937    0.07029 -14.076  5.371e-45
ethnicityPolynesian -1.52556    0.10651 -14.323  1.572e-46
ethnicityOther       0.93824    0.14208   6.604  4.009e-11
sexM                 0.09603    0.04437   2.164  3.045e-02
```

```
> coef(edu.pom3, matrix = TRUE)

                    logit(P[Y>=2]) logit(P[Y>=3]) logit(P[Y>=4])
(Intercept)                3.29910       -0.52613       -1.84605
poly(log.age, 2)1        -51.15534      -51.15534      -51.15534
poly(log.age, 2)2        -32.64882      -32.64882      -32.64882
ethnicityMaori            -0.98937       -0.98937       -0.98937
ethnicityPolynesian       -1.52556       -1.52556       -1.52556
ethnicityOther             0.93824        0.93824        0.93824
sexM                       0.09603        0.09603        0.09603
```

At what age is the highest educational level attained?

```
> edu.pom3b <- vglm(educ ~ log.age + I(log.age^2) + ethnicity + sex,
                    propodds, data = sxs.nz)
> coef(edu.pom3b)

      (Intercept):1       (Intercept):2       (Intercept):3             log.age
          -20.55671           -24.38194           -25.70186           14.51112
       I(log.age^2)      ethnicityMaori ethnicityPolynesian      ethnicityOther
           -2.15834            -0.98937            -1.52556             0.93824
              sexM
            0.09603

> nadir <- -coef(edu.pom3b)["log.age"] / (2 * coef(edu.pom3b)["I(log.age^2)"])
> c(age = round(exp(as.vector(nadir)), 1))

 age
28.8
```

This may be explained by people slowly upskilling themselves and the amount of postgraduate study available, as well as tough economic times!

P-splines

Let's try the more modern PS-VGAMs.

```
> edu.pom1.psvgam <- vgam(educ ~ sm.ps(age) + ethnicity + sex, propodds, sxs.nz)
> edu.pom3.psvgam <- vgam(educ ~ sm.ps(log.age) + ethnicity + sex,
                          propodds, data = sxs.nz)
```

```
> plot(edu.pom1.psvgam, se = TRUE, control = plotvgam.control(lcol = "blue",
       scol = "orange", rcol = "green"))
```

The plot looks very similar to the one on Slide 167.

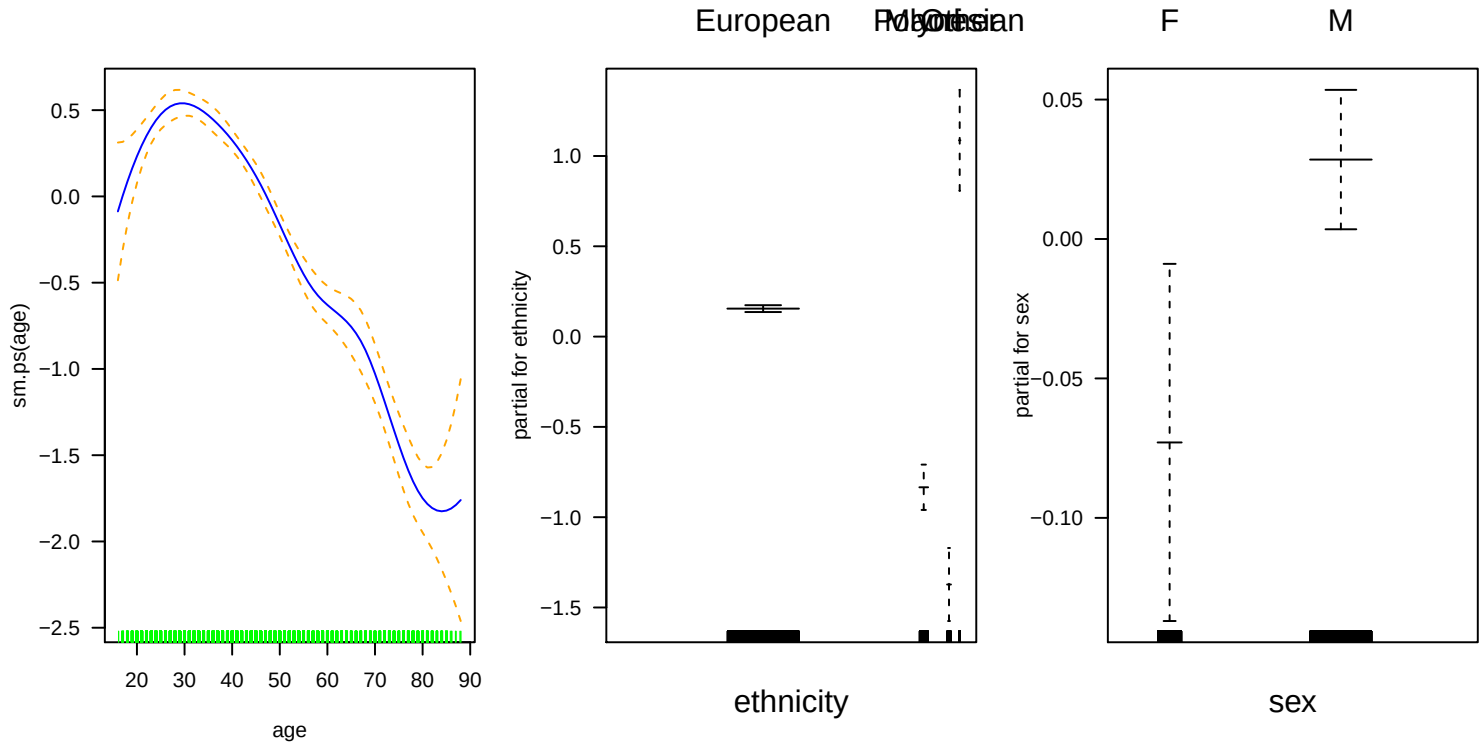


Figure : PS-VGAM: fitted component functions.

And let's repeat the comparison...

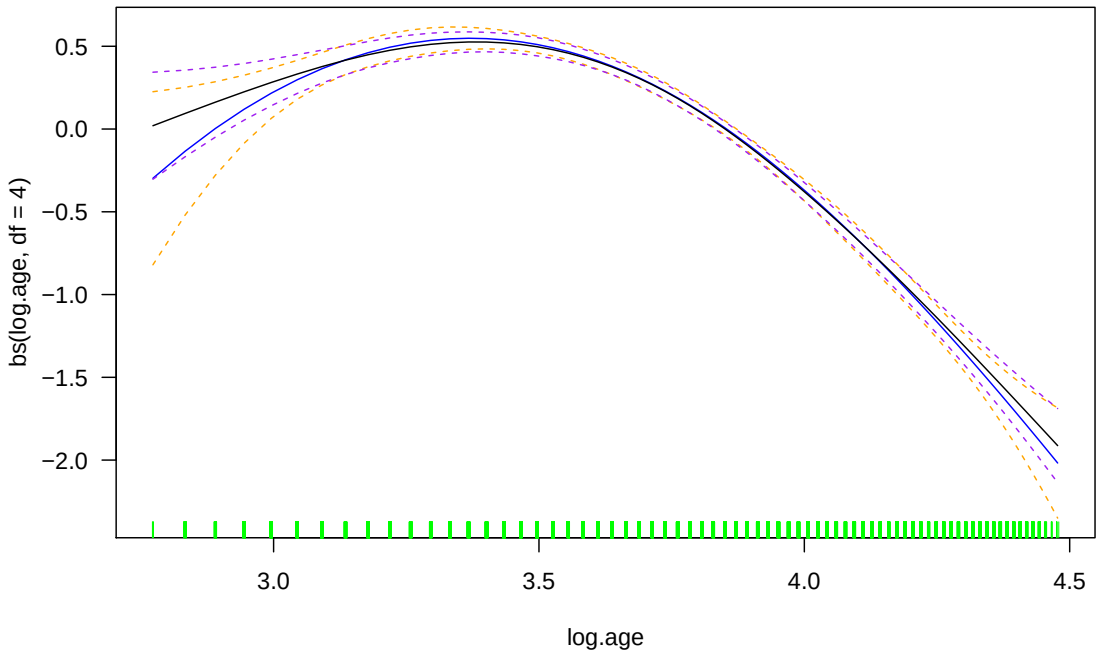


Figure : PS-VGAM: fitted component functions. The black and purple correspond to the PS-VGAM.

They look a little less similar but are still very similar! The PS-VGAM is slightly less nonlinear, especially at the LHS.

Automatic Smoothing Parameter Selection

A Short History of GAMs...

1. *First generation GAMs*: Hastie and Tibshirani, late-1980s/early-1990s, gam.

   Main ideas: smoothing splines, backfitting.

2. *Second generation GAMs*: Wood, early-2000s onwards, mgcv.

   Main ideas: "Penalized B-splines" (Eilers and Marx, 1996). Smoothing parameter selection is much easier, as well as inference.

3. *First generation VGAMs*: Yee and Wild (1996), VGAM.

   Main ideas: vector (smoothing) splines, vector backfitting.

4. *Second generation VGAMs*: VGAM has a rudimentary implementation of P-spline VGAMs. Should be refined by the end of this year. Joint work with Chanatda Somchit and Chris Wild. See `sm.os()` and `sm.ps()`.
   A comparison between O-splines and P-splines is in Wand and Ormerod (2008).

The following gives an example.

Example 1: Fuel Efficiency Data

Consider `mpg` in gamair: fuel efficiency data for c.200 cars in USA.

$Y_1 =$ `city.mpg`	City fuel consumption (miles per gallon),
$Y_2 =$ `hw.mpg`	Highway fuel consumption (miles per gallon),
$X_2 =$ `weight`	Car weight (pounds),
$X_3 =$ `hp`	Engine power (horsepower).

Let's assume

$$\begin{pmatrix} Y_1 \\ Y_2 \end{pmatrix} \sim N_2 \left(\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \sigma_{11}^2 & \rho\,\sigma_{11}\,\sigma_{22} \\ \rho\,\sigma_{11}\,\sigma_{22} & \sigma_{22}^2 \end{pmatrix} \right). \quad (57)$$

The family function `binormal(zero = NULL)` was used, which specifies

$$\eta_1 = \mu_1, \quad \eta_2 = \mu_2, \quad \eta_3 = \log \sigma_{11}, \quad \eta_4 = \log \sigma_{22}, \quad \eta_5 = \log \frac{1+\rho}{1-\rho}, \tag{58}$$

so that the covariances can be modelled with covariates.

```
data("mpg", package = "gamair"); mpg.use <- na.omit(mpg)

mpg.fit1 <- vgam(cbind(city.mpg, hw.mpg) ~ sm.ps(weight) + sm.ps(hp),
# The next line gets around a bug that needs fixing:
       Maxit.outer = 5, maxit = 7,
       binormal(zero = ""), data = mpg.use, trace = FALSE)

plotno <- 1
for (jay in 1:2)
  for (i in 1:5) {
    plot(mpg.fit1, se = TRUE, shade = TRUE, las = 1, which.term = jay,
         mgp = c(2.3, 1, 0), slwd = 3, which.cf = i, main = letters[plotno])
    plotno <- plotno + 1
  }
```

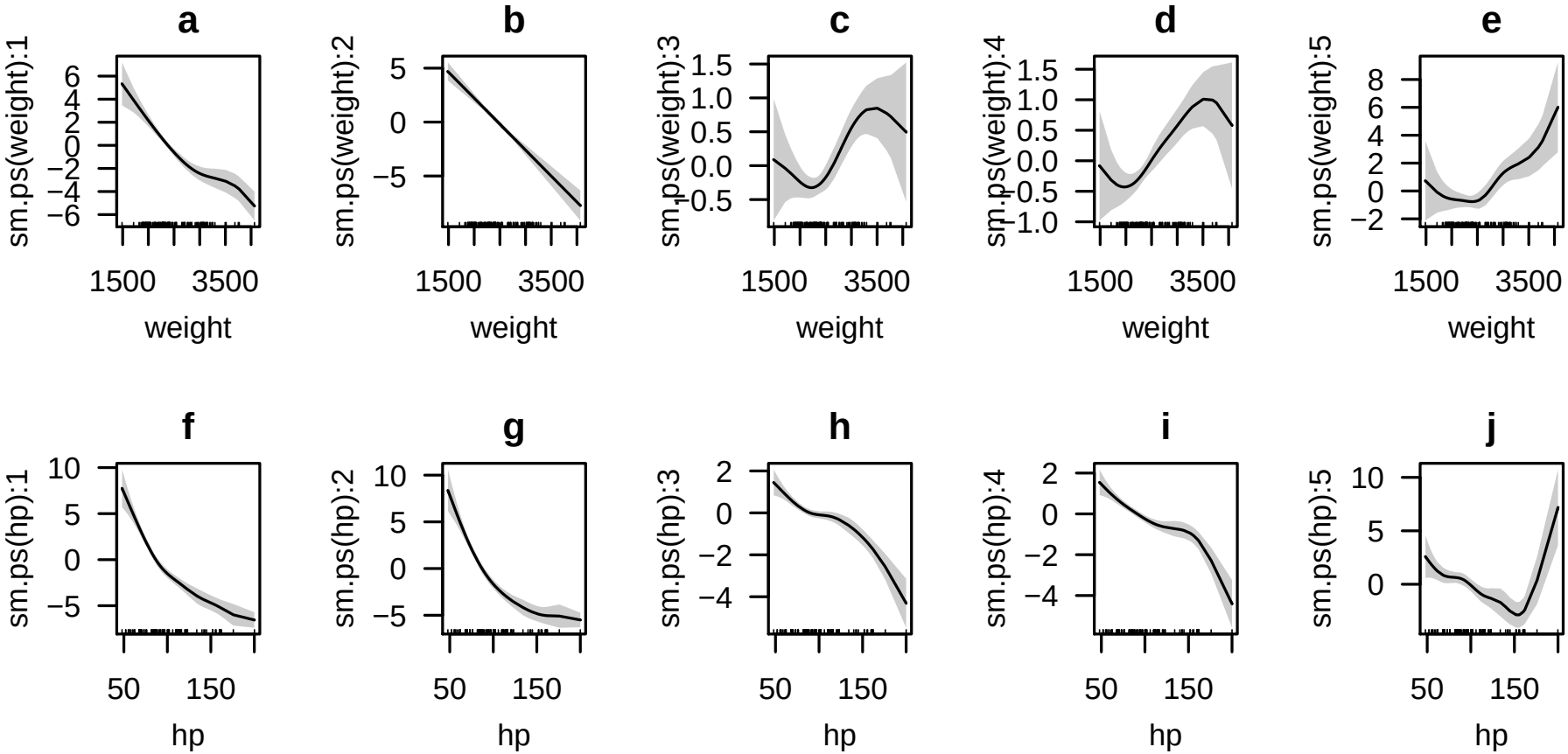


Figure : Using VGAM to estimate the component functions from a $N_2(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ distribution fitted to the `mpg` data in gamair. See (57) and (58).

Notes:

- Could try

```
> library("mgcv")

Loading required package:  nlme
This is mgcv 1.8-15.  For overview type 'help("mgcv-package")'.

Attaching package:  'mgcv'
The following object is masked from 'package:VGAM':
    s
> gamfit <- gam(list(city.mpg ~ s(weight) + s(hp),
                     hw.mpg   ~ s(weight) + s(hp)),
                mvn(d = 2), data = mpg.use)
> plot(gamfit)
```

but the elements of $\boldsymbol{\Sigma}$ are intercept-only. See the figure in Slide 184.

- It would be straightforward using VGAM to constrain plots a,b to differ by a constant (known or unknown), and similarly for plots f,g, for example,

$$\mathbf{H}_1 = \mathbf{I}_5, \quad \mathbf{H}_2 = \mathbf{H}_3 = \begin{pmatrix} 1 & 0 \\ 1 & 0 \\ 0 & \mathbf{I}_3 \end{pmatrix}$$

  in the unknown case.

- Some further details are at Yee (2016).

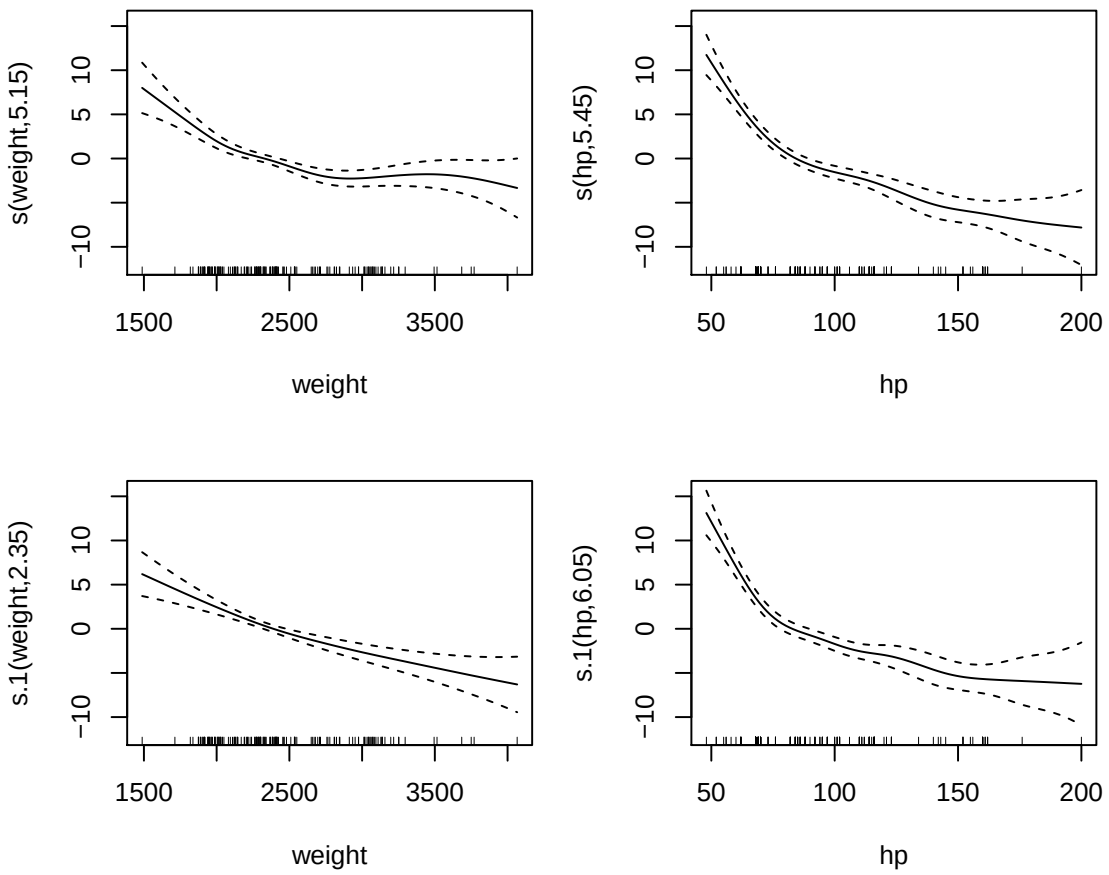


Figure : Using mgcv to estimate the component functions from a $N_2(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ distribution fitted to the `mpg` data in gamair.

```
> summary(mpg.fit1, presid = FALSE)


Call:
vgam(formula = cbind(city.mpg, hw.mpg) ~ sm.ps(weight) + sm.ps(hp),
    family = binormal(zero = ""), data = mpg.use, Maxit.outer = 5,
    maxit = 7, trace = FALSE)


Parametric coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept):1   26.213      0.178     147   <2e-16 ***
(Intercept):2   31.770      0.199     160   <2e-16 ***
(Intercept):3    0.695      0.056      12   <2e-16 ***
(Intercept):4    0.820      0.056      15   <2e-16 ***
(Intercept):5    2.529      0.158      16   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
                 edf Est.rank Chi.sq p-value
sm.ps(weight):1 3.99        8     89   7e-16 ***
sm.ps(weight):2 0.99        2    124  <2e-16 ***
sm.ps(weight):3 4.37        9     29   7e-04 ***
sm.ps(weight):4 4.33        9     29   7e-04 ***
sm.ps(weight):5 4.26        9     31   3e-04 ***
sm.ps(hp):1     4.78        9    266  <2e-16 ***
sm.ps(hp):2     4.33        9    274  <2e-16 ***
sm.ps(hp):3     4.32        9    102  <2e-16 ***
sm.ps(hp):4     4.63        9     93   4e-16 ***
sm.ps(hp):5     5.18        9     53   3e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

What are Latent Variables

Latent variables have various meanings in statistics.

- The word latent means concealed, dormant, hidden.
- Often a random variable which cannot be measured directly or an unobserved or latent trait. Some examples:
  - quality of life,
  - business confidence,
  - morale,
  - happiness.
- They are inferred from other variables that are observed (directly measured). For example, for quality of life, we can measure wealth, employment, environment, physical health, mental health, education, recreation and leisure time and social belonging.
- Other similar names: *hidden variables*, *hypothetical variables* or *hypothetical constructs*

- Often a linear combination of the explanatory variables, e.g.,

$$\nu \quad = \quad \mathbf{c}^T \mathbf{x} \tag{59}$$

  For the $\nu =$ quality of life example, $\mathbf{x} = ($wealth, employment, environment, physical health, mental health, education, recreation and leisure time, social belonging$)^T$. Expect the coefficients to be mostly positive.
- I call the $\mathbf{c}$ the *constrained coefficients*. Also called *loadings* or *weights*.
- It reduces the dimensionality of the model, e.g., from p to 1.
- Used in many disciplines, e.g., psychology, economics, medicine, ecology, social sciences.

Reduced-Rank VGLMs

These can be surprisingly useful.

Motivation: if M and p are large then $\mathbf{B} = (\boldsymbol{\beta}_1\, \boldsymbol{\beta}_2\, \cdots\, \boldsymbol{\beta}_M)$ is "too big" (Mp elements).

Idea: approximate part of $\mathbf{B}$ by a lower rank matrix, i.e, the product of two 'thin' matrices $\mathbf{A}\,\mathbf{C}^T$.

Partition $\mathbf{x} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix}$ and $\mathbf{B} = \begin{pmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 \end{pmatrix}$ accordingly.

RR-VGLMs:

$$\boldsymbol{\eta} \;=\; \mathbf{B}_1^T \mathbf{x}_1 + \mathbf{A}\,\mathbf{C}^T \mathbf{x}_2 \;\; \text{i.e.,} \tag{60}$$

$$\mathbf{B}_2 \;=\; \mathbf{C}\,\mathbf{A}^T, \qquad (p_2 \times R) \times (R \times M) \text{ in dimension.} \tag{61}$$

$$\boldsymbol{\nu} \;=\; \mathbf{C}^T \mathbf{x}_2 \qquad \text{is a } R\text{-vector of } \textit{latent variables} \tag{62}$$

Often $R = 1$, 2, or 3 .... Latent variables are an important concept in ecology, e.g., direct and indirect gradient analysis.

Roles:

- $\mathbf{C}$ can be considered as choosing the 'best' predictors from a linear combination of the original predictors.
- $\mathbf{A}$ are regression coefficients of the new predictors $\boldsymbol{\nu}$.

Example: *Stereotype model* = *RR-multinomial logit model* (Anderson, 1984).

A Simple and Important Result

RR-VGLMs are VGLMs where the constraint matrices are unknown and to be estimated.

Proof: write (60) as

$$\boldsymbol{\eta} \;=\; \mathbf{B}_1^T \mathbf{x}_1 + \mathbf{A}\,\mathbf{C}^T \mathbf{x}_2 \;=\; \mathbf{B}_1^T \mathbf{x}_1 + \sum_{k=1}^{p_2} \mathbf{A} \begin{pmatrix} c_{k1} \\ \vdots \\ c_{kR} \end{pmatrix} x_{2k} \tag{63}$$

and match up the quantities.

Estimation by an *alternating algorithm* exploits this.

Example: Stereotype Model †

We look at the marital status data.

Data: based on `xs.nz`. For homogeneity, the analysis was restricted to a subset of 4105 European males with no `NA`s in any of the variables used.

Aim: we are interested in exploring whether certain lifestyle and psychological variables are associated with marital status, especially separation/divorce.

Y = marital status, with levels written as

- 1 = single,
- 2 = separated or divorced,
- 3 = widower, and
- 4 = married or living with a partner (the reference group). Why?

There were 648, 209, 22 and 3226 such men respectively. We have $M = 3$.

```
fit1 <-
  rrvglm(marital ~ age30 + logedu1 + binge + smokenow +
                   sun + nerves + nervous + hurt + tense +
                   miserable + fedup + worry + worrier + moody,
         family = multinomial, data = workforce,
         noRRR = ~ age30 + logedu1, Rank = 1, Index.corner = 2)
```

Table : Variables used in the workforce study. All except the first three are binary, denoted by 1 and 0 for "Yes" and 'No" respectively. The second column are the questionnaire wordings.

`marital`	Y = marital status
`age30`	Age $-$ 30, in years
`logedu1`	log(1+ Years of education at secondary or high school)
`binge`	In the last three months what is the largest number of drinks that you had on any one day? (1= 20 or more, 0 = less than 20)
`smokenow`	Current smoker?
`sun`	Does not usually wear a hat, shirt or suntan lotion when outside during summer
`nerves`	Do you suffer from "nerves"?
`nervous`	Would you call yourself a "nervous" person?
`hurt`	Are your feelings easily hurt?
`tense`	Would you call yourself tense or "highly strung"?
`miserable`	Do you feel "just miserable" for no reason?
`fedup`	Do you often feel "fed-up"?
`worry`	Do you worry about awful things that might happen?
`worrier`	Are you a worrier?
`moody`	Does your mood often go up and down?

Table : Estimated MLM regression coefficients for the workforce data. An asterisk denotes a Wald statistic for that coefficient is greater than 2 in absolute value. Nb. $Y = 1 =$ single, $2 =$ separated or divorced, $3 =$ widower, and $4 =$ married or living with a partner.

Variable	$\log(p_1/p_4)$	$\log(p_2/p_4)$	$\log(p_3/p_4)$
Intercept	-1.573^*	-2.921^*	-6.123^*
age30	-0.189^*	0.012	0.077^*
logedu1	0.254	-0.316	-0.198
binge	0.801^*	0.319	1.127
smokenow	0.022	0.501^*	0.654
sun	-0.066	0.120	-0.088
nerves	-0.102	0.123	-1.457
nervous	0.298	0.354	1.007
hurt	0.184	0.210	0.483
tense	0.166	0.484^*	1.108
miserable	-0.050	0.128	-0.093
fedup	0.112	0.249	-0.214
worry	0.113	-0.102	-0.549
worrier	-0.028	-0.243	-0.345
moody	-0.111	0.092	-0.194

Table : Partial rank-1 RR-MLM fitted to the workforce data: the tables are $\widehat{\mathbf{B}}_1$, $\widehat{\mathbf{C}}$, and $\widehat{\mathbf{A}}$ respectively.

Variable	$\log(p_1/p_4)$	$\log(p_2/p_4)$	$\log(p_3/p_4)$
Intercept	-1.762^*	-2.699^*	-6.711^*
age30	-0.191^*	0.012	0.086^*
logedu1	0.338	-0.365	-0.089

Variable	$\widehat{\mathbf{C}}$	RR-MLM SE	MC SE	MLM SE
binge	0.786^*	0.164	0.190	0.153
smokenow	0.306^*	0.132	0.124	0.116
sun	0.015	0.117	0.121	0.116
nerves	-0.074	0.141	0.145	0.138
nervous	0.430^*	0.168	0.171	0.162
hurt	0.248^*	0.122	0.127	0.119
tense	0.416^*	0.170	0.174	0.160
miserable	0.027	0.132	0.132	0.130
fedup	0.162	0.123	0.120	0.119
worry	0.004	0.145	0.152	0.144
worrier	-0.160	0.130	0.127	0.126
moody	-0.052	0.120	0.128	0.120

$\widehat{\mathbf{A}}$	RR-MLM SE	MC SE
0.725^*	0.169	0.181
1.000	—	—
1.418^*	0.529	0.615

Some Rank-1 Interpretations

1. $\widehat{\nu} = \widehat{\mathbf{c}}^T \mathbf{x}_2$ is a latent variable measuring lack of general health and well-being,
2. Five of the unhealthy variables are significant (cf. 3 for MLM),
3. $\widehat{\nu}$ has a greater effect on widowhood and less on singleness compared to separation/divorce,
4. As a specific example, consider the 'effect' of smoking versus nonsmoking. Then keeping all other variables fixed, $\widehat{\nu}(\text{smoking}) - \widehat{\nu}(\text{nonsmoking}) = 0.306$ so that

$$\frac{\widehat{p}_{\text{separated/divorced}}}{\widehat{p}_{\text{married}}} \approx \exp(1 \times 0.306) \approx 1.36$$

which compares to

$$\frac{\widehat{p}_{\text{widowed}}}{\widehat{p}_{\text{married}}} \approx \exp(1.418 \times 0.306) \approx 1.54,$$

5. Using `bs(age30, df = 3)` perturbs the results only slightly,
6. $\widehat{\phi}^{1/2} \approx 1.2$ for all rank models, indicating only a small amount of over-dispersion.

Table : Partial rank-2 RR-MLM fitted to the workforce data: the tables are $\widehat{\mathbf{B}}_1$, $\widehat{\mathbf{C}}$, and $\widehat{\mathbf{A}}$ respectively. An asterisk denotes a Wald statistic for that coefficient is greater than 2 in absolute value.

Variable	$\log(p_1/p_4)$	$\log(p_2/p_4)$	$\log(p_3/p_4)$
Intercept	-1.539^*	-2.908^*	-6.604^*
age30	-0.189^*	0.012	0.087^*
logedu1	0.257	-0.318	-0.111

binge	0.775^*	0.306
smokenow	0.071	0.513^*
sun	-0.068	0.120
nerves	-0.177	0.101
nervous	0.321^*	0.363
hurt	0.199	0.214
tense	0.225	0.503^*
miserable	-0.056	0.127
fedup	0.068	0.236
worry	0.068	-0.118
worrier	-0.051	-0.249
moody	-0.121	0.091

1.000	0.000
0.000	1.000
1.796	0.253

Some Rank-2 Interpretations

1. For the rank-2 model, the single and separated/divorce groups were chosen as baseline.
2. The fitted model suggests that

$$\widehat{\boldsymbol{\eta}} = \begin{pmatrix} \log\left(\frac{\widehat{p}_{\text{single}}}{\widehat{p}_{\text{married}}}\right) \\ \log\left(\frac{\widehat{p}_{\text{separated/divorced}}}{\widehat{p}_{\text{married}}}\right) \\ \log\left(\frac{\widehat{p}_{\text{widowed}}}{\widehat{p}_{\text{married}}}\right) \end{pmatrix} \approx \begin{pmatrix} \widehat{\nu}_1 \\ \widehat{\nu}_2 \\ 2\,\widehat{\nu}_1 \end{pmatrix},$$

where

$$\widehat{\nu}_1 \approx \frac{3}{4}\,\texttt{binge} + \frac{1}{3}\,\texttt{nervous}$$

and

$$\widehat{\nu}_2 \approx \frac{1}{2}\,\texttt{smokenow} + \frac{1}{2}\,\texttt{tense},$$

3. None of the heavily weighted variables in $\widehat{\nu}_j$ are in common,
4. The transition from singleness to marriage versus married to widowhood appears to be opposites and distinct from the process of separation/divorce.

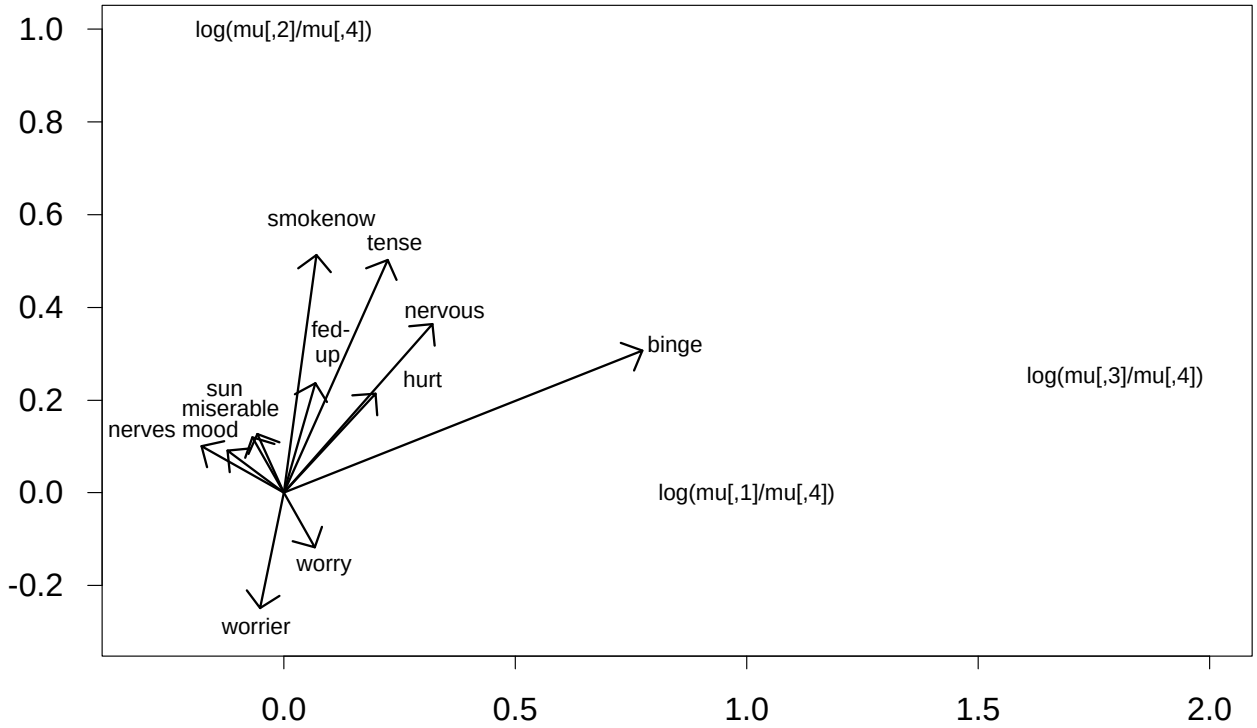


Figure : Biplot (output from `biplot.rrvglm()`) from the rank-2 partial RR-MLM fitted to the workforce data. The label "`mu[, 1]`" is p_1 etc. The biplot comes about by plotting the rows of **A** and **C**.

Table : Deviances and AIC.

Rank	Deviance	AIC	Degrees of Freedom
0	5421.919	4212.689	12312
1	4120.695	4166.695	12292
2	4102.293	4172.293	12280
3	4095.043	4185.043	12270

Conclusion

- Both the rank-1 and rank-2 partial RR-MLMs have successfully shed insights into the marital status of the workforce data.
- For the rank-1 model, reducing 36 regression coefficients down to $12 + 2 = 14$ has enabled two extra psychological variables to become statistically significant.
- An interpretation via latent variables has been enlightening for both RR-MLMs.

References


Anderson, J. A., 1984. Regression and ordered categorical variables. J. Roy. Statist. Soc. Ser. B 46 (1), 1–30, with discussion.


Chambers, J. M., 1998. Programming with Data: A Guide to the S Language. Springer, New York, USA.


Chambers, J. M., 2008. Software for Data Analysis: Programming with R. Statistics and Computing. Springer, New York, USA.


Hastie, T. J., Tibshirani, R. J., 1990. Generalized Additive Models. Chapman & Hall, London.


Wand, M. P., Ormerod, J. T., 2008. On semiparametric regression with O'Sullivan penalized splines. Australian & New Zealand Journal of Statistics 50 (2), 179–198.


Wilkinson, G. N., Rogers, C. E., 1973. Symbolic description of factorial models for analysis of variance. J. Roy. Statist. Soc. Ser. C 22 (3), 392–399.


Wood, S. N., 2006. Generalized Additive Models: An Introduction with R. Chapman and Hall, London.


Wood, S. N., 2017. Generalized Additive Models: An Introduction with R, 2nd Edition. Chapman & Hall/CRC, London.


Yee, T. W., 2004. A new technique for maximum-likelihood canonical Gaussian ordination. Ecol. Monogr. 74 (4), 685–701.


Yee, T. W., 2006. Constrained additive ordination. Ecology 87 (1), 203–213.


Yee, T. W., 2015. Vector Generalized Linear and Additive Models: With an Implementation in R. Springer, New York, USA.


Yee, T. W., 2016. Comment on: "Smoothing parameter and model selection for general smooth models" by Wood, S. N. and Pya, N. and Säfken, N. Journal of the American Statistical Association 111 (516), 1565–1568.


Yee, T. W., Hadi, A. F., 2014. Row-column interaction models, with an R implementation. Computational Statistics 29 (6), 1427–1445.


Yee, T. W., Hastie, T. J., 2003. Reduced-rank vector generalized linear models. Statistical Modelling 3 (1), 15–41.


Yee, T. W., Wild, C. J., 1996. Vector generalized additive models. J. Roy. Statist. Soc. Ser. B 58 (3), 481–493.

Concluding Remarks

1. LMs . . . GLMs . . . VGLMs . . . (*breadth*).
   VGLMs fit a very large class of regression models. They are model-driven. MLEs and classical inference are available.
2. AMs . . . GAMs . . . VGAMs . . . (*depth*).
   Smoothing forms the basis for a data-driven exploratory analysis, to improve model-driven analyses They are *recommended!!*
3. For VGLMs/VGAMs the central concepts are
   - $\eta_j = g_j(\theta_j)$ for $j = 1, \ldots, M$,
   - multiple responses,
   - constraint matrices $\mathbf{H}_k$,
   - `xij`.
4. VGAM extends the practical use of regression significantly. It is freely available on CRAN or at the author's web page at
   `https://www.stat.auckland.ac.nz/~yee`