The VGAM Package

by Thomas W. Yee

Introduction

The **VGAM** package implements several large classes of regression models of which *vector generalized linear and additive models* (VGLMs/VGAMs) are most commonly used (Table 1). The primary key words are maximum likelihood estimation, iteratively reweighted least squares (IRLS), Fisher scoring and additive models. Other subsidiary concepts are reduced rank regression, constrained ordination and vector smoothing. Altogether the package represents a broad and unified framework.

Written in S4 (Chambers, 1998), the **VGAM** modelling functions are used in a similar manner as `glm()` and Trevor Hastie's `gam()` (in **gam**, which is very similar to his S-PLUS version). Given a `vglm()/vgam()` object, standard generic functions such as `coef()`, `fitted()`, `predict()`, `summary()`, `vcov()` are available. The typical usage is like

```
vglm(yvector ~ x2 + x3 + x4,
     family = VGAMfamilyFunction,
     data = mydata)
vgam(ymatrix ~ s(x2) + x3,
     family = VGAMfamilyFunction,
     data = mydata)
```

Many models have a multivariate response, and therefore the LHS of the formula is often a matrix (otherwise a vector). The function assigned to the `family` argument is known as a **VGAM** *family function*. Table 2 lists some widely used ones.

The scope of **VGAM** is very broad; it potentially covers a wide range of multivariate response types and models, including univariate and multivariate distributions, categorical data analysis, quantile and expectile regression, time series, survival analysis, extreme value analysis, mixture models, correlated binary data, bioassay data and nonlinear least-squares problems. Consequently there is overlap with many other R packages. **VGAM** is designed to be as general as possible; currently over 100 **VGAM** family functions are available and new ones are being written all the time.

Basically, VGLMs model each parameter, transformed if necessary, as a linear combination of the explanatory variables. That is,

$$g_j(\theta_j) \;=\; \eta_j \;=\; \boldsymbol{\beta}_j^T \boldsymbol{x} \tag{1}$$

where g_j is a parameter link function. This idea emerges in the next section. VGAMs extend (1) to

$$g_j(\theta_j) \;=\; \eta_j \;=\; \sum_{k=1}^{p} f_{(j)k}(x_k), \tag{2}$$

i.e., an additive model for each parameter.

Two examples

To motivate **VGAM** let's consider two specific examples: the negative binomial distribution and the proportional odds model. These are often fitted using `glm.nb()` and `polr()` respectively, both of which happen to reside in the **MASS** package. We now attempt to highlight some advantages of using **VGAM** to fit these models—that is, to describe some of the VGLM/VGAM framework.

Negative binomial regression

A negative binomial random variable Y has a probability function that can be written as

$$P(Y=y) \;=\; \binom{y+k-1}{y} \left(\frac{\mu}{\mu+k}\right)^y \left(\frac{k}{k+\mu}\right)^k$$

where $y = 0, 1, 2, \ldots$, and parameters μ $(= E(Y))$ and k are positive. The quantity $1/k$ is known as the dispersion parameter, and the Poisson distribution is the limit as k approaches infinity. The negative binomial is commonly used to handle overdispersion with respect to the Poisson distribution because $\mathrm{Var}(Y) = \mu + \mu^2/k > \mu$.

The **MASS** implementation is restricted to a intercept-only estimate of k. For example, one cannot fit $\log k = \beta_{(2)1} + \beta_{(2)2}x_2$, say. In contrast, **VGAM** can fit

$$\begin{aligned} \log \mu &= \eta_1 = \boldsymbol{\beta}_1^T \boldsymbol{x}, \\ \log k &= \eta_2 = \boldsymbol{\beta}_2^T \boldsymbol{x}. \end{aligned}$$

by using `negbinomial(zero=NULL)` in the call to `vglm()`. Note that it is natural for any positive parameter to have the log link as the default.

There are also variants of the negative binomial in the **VGAM** package, e.g., the zero-inflated and zero-altered versions; see Table 2.

Proportional odds model

The proportional odds model (POM) for an ordered factor Y taking levels $\{1, 2, \ldots, M+1\}$ may be written

$$\mathrm{logit}\, P(Y \le j | \boldsymbol{x}) \;=\; \eta_j(\boldsymbol{x}) \tag{3}$$

where $\boldsymbol{x} = (x_1, \ldots, x_p)^T$ with $x_1 = 1$ denoting an intercept, and

$$\eta_j(\boldsymbol{x}) \;=\; \beta_{(j)1} + \boldsymbol{\beta}_*^T \boldsymbol{x}_{(-1)}, \quad j = 1, \ldots, M. \tag{4}$$

Here, $\boldsymbol{x}_{(-1)}$ is $\boldsymbol{x}$ with the first element deleted. Standard references for the POM include McCullagh and Nelder (1989) and Agresti (2002).

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

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

The **VGAM** family function `cumulative()` may be used to fit this model. Additionally, in contrast to `polr()`, it can also fit some useful variants/extensions. Some of these are described as follows.

(i) Nonproportional odds model

The regression coefficients $\boldsymbol{\beta}_*$ in (4) are common for all j. This is known as the so-called *parallelism* or *proportional odds* assumption. This is a theoretically elegant idea because the η_j do not cross, and therefore there are no problems with negative probabilities etc. However, this assumption is a strong one and ought to be checked for each explanatory variable.

(ii) Selecting different link functions.

VGAM is purposely extensible. The `link` argument to `cumulative()` may be assigned *any* **VGAM** link function whereas `polr()` currently provides four fixed choices. Of course, the user is responsible for choosing appropriate links, e.g., the probit and complementary log-log. Users may write their own **VGAM** link function if they wish.

(iii) *Partial proportional odds model*

An intermediary between the proportional odds and nonproportional odds models is to have some explanatory variables parallel and others not. Some authors call this a *partial proportional odds model*. As an example, suppose $p = 4$, $M = 2$ and

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

Here, the parallelism assumption applies to x_2 and x_4 only. This can be achieved by

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

or equivalently,

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

There are several other extensions that can easily be handled by the constraint matrices idea described later.

(iv) Common **VGAM** family function arguments.

Many authors define the POM as

$$\operatorname{logit} P(Y \geq j+1 | \boldsymbol{x}) \;=\; \eta_j(\boldsymbol{x}) \qquad (5)$$

rather than (3) because $M = 1$ coincides with logistic regression. Many **VGAM** family functions share common arguments and one of them is `reverse`. Here, setting `reverse=TRUE` will fit (5). Other common arguments include `link` (usually one for each parameter), `zero`, `parallel` and initial values of parameters.

Modelling ordinal responses in **MASS** is serviced by a "one-off" function whereas **VGAM** naturally supports several related models such as the adjacent categories and continuation/stopping ratio models; see Table 2.

General framework

In this section we describe the general framework and notation. Fuller details can be found in the references of Table 1. Although this section may be skipped on first reading, an understanding of these details is necessary to realize its full potential.

Vector generalized linear models

Suppose the observed response $\boldsymbol{y}$ is a q-dimensional vector. VGLMs are defined as a model for which the conditional distribution of $\boldsymbol{Y}$ given explanatory $\boldsymbol{x}$ is of the form

$$f(\boldsymbol{y}|\boldsymbol{x};\mathbf{B}) \;=\; h(\boldsymbol{y},\eta_1,\ldots,\eta_M) \qquad (6)$$

for some known function $h(\cdot)$, where $\mathbf{B} = (\boldsymbol{\beta}_1\,\boldsymbol{\beta}_2\,\cdots\,\boldsymbol{\beta}_M)$ is a $p\times M$ matrix of unknown regression coefficients, and the jth linear predictor is

$$\eta_j \;=\; \boldsymbol{\beta}_j^T\boldsymbol{x} \;=\; \sum_{k=1}^{p}\beta_{(j)k}\,x_k, \quad j=1,\ldots,M, \qquad (7)$$

where $\boldsymbol{x} = (x_1,\ldots,x_p)^T$ with $x_1 = 1$ if there is an intercept. VGLMs are thus like GLMs but allow for multiple linear predictors, and they encompass models outside the limited confines of the classical exponential family.

The η_j of VGLMs may be applied directly to parameters of a distribution rather than just to means as for GLMs. A simple example is a univariate distribution with a location parameter ξ and a scale parameter $\sigma > 0$, where we may take $\eta_1 = \xi$ and $\eta_2 = \log\,\sigma$. In general, $\eta_j = g_j(\theta_j)$ for some parameter link function g_j and parameter θ_j. In **VGAM**, there are currently over a dozen links to choose from (Table 3), of which any can be assigned to any parameter, ensuring maximum flexibility.

There is no relationship between q and M in general: it depends specifically on the model or distribution to be fitted. For example, the mixture of two normal distributions has $q = 1$ and $M = 5$.

VGLMs are estimated by IRLS. Most models that can be fitted have a log-likelihood

$$\ell \;=\; \sum_{i=1}^{n} w_i\,\ell_i \qquad (8)$$

and this will be assumed here. The w_i are known positive prior weights. Let $\boldsymbol{x}_i$ denote the explanatory vector for the ith observation, for $i = 1,\ldots,n$. Then one can write

$$\boldsymbol{\eta}_i \;=\; \begin{pmatrix}\eta_1(\boldsymbol{x}_i)\\ \vdots \\ \eta_M(\boldsymbol{x}_i)\end{pmatrix} \;=\; \mathbf{B}^T\boldsymbol{x}_i \;=\; \begin{pmatrix}\boldsymbol{\beta}_1^T\boldsymbol{x}_i\\ \vdots \\ \boldsymbol{\beta}_M^T\boldsymbol{x}_i\end{pmatrix}. \qquad (9)$$

In IRLS, an adjusted dependent vector $\boldsymbol{z}_i = \boldsymbol{\eta}_i + \mathbf{W}_i^{-1}\boldsymbol{d}_i$ is regressed upon a large (VLM) design matrix, with $\boldsymbol{d}_i = w_i\,\partial\ell_i/\partial\boldsymbol{\eta}_i$. The working weights $\mathbf{W}_i$ here are $w_i\mathrm{Var}(\partial\ell_i/\partial\boldsymbol{\eta}_i)$ (which, under regularity conditions, is equal to $-w_i\,E[\partial^2\ell_i/(\partial\boldsymbol{\eta}_i\,\partial\boldsymbol{\eta}_i^T)]$, called the expected information matrix or EIM), giving rise to the Fisher scoring algorithm. Fisher scoring usually has good numerical stability because the $\mathbf{W}_i$ are positive-definite over a larger region of parameter space. The price to pay for this stability is typically a slower convergence rate and less accurate standard errors (Efron and Hinkley, 1978) compared to the Newton-Raphson algorithm.

Vector generalized additive models

VGAMs provide additive-model extensions to VGLMs, that is, (7) is generalized to

$$\eta_j(\boldsymbol{x}) \;=\; \beta_{(j)1} + \sum_{k=2}^{p} f_{(j)k}(x_k), \quad j=1,\ldots,M,$$

a sum of smooth functions of the individual covariates, just as with ordinary GAMs (Hastie and Tibshirani, 1990). The $\boldsymbol{f}_k = (f_{(1)k}(x_k),\ldots,f_{(M)k}(x_k))^T$ are centered for uniqueness, and are estimated *simultaneously* using *vector smoothers*. VGAMs are thus a visual data-driven method that is well suited to exploring data, and they retain the simplicity of interpretation that GAMs possess.

In practice we may wish to constrain the effect of a covariate to be the same for some of the η_j and to have no effect for others. For example, for VGAMs, we may wish to take

$$\begin{aligned}\eta_1 &= \beta_{(1)1} + f_{(1)2}(x_2) + f_{(1)3}(x_3),\\ \eta_2 &= \beta_{(2)1} + f_{(1)2}(x_2),\end{aligned}$$

so that $f_{(1)2} \equiv f_{(2)2}$ and $f_{(2)3} \equiv 0$. For VGAMs, we can represent these models using

$$\begin{aligned}\boldsymbol{\eta}(\boldsymbol{x}) &= \boldsymbol{\beta}_{(1)} + \sum_{k=2}^{p}\boldsymbol{f}_k(x_k)\\ &= \mathbf{H}_1\,\boldsymbol{\beta}_{(1)}^{*} + \sum_{k=2}^{p}\mathbf{H}_k\,\boldsymbol{f}_k^{*}(x_k) \qquad (10)\end{aligned}$$

where $\mathbf{H}_1,\mathbf{H}_2,\ldots,\mathbf{H}_p$ are known full-column rank *constraint matrices*, $\boldsymbol{f}_k^*$ is a vector containing a possibly reduced set of component functions and $\boldsymbol{\beta}_{(1)}^*$ is a vector of unknown intercepts. With no constraints at all, $\mathbf{H}_1 = \mathbf{H}_2 = \cdots = \mathbf{H}_p = \mathbf{I}_M$ and $\boldsymbol{\beta}_{(1)}^* = \boldsymbol{\beta}_{(1)}$. Like the $\boldsymbol{f}_k$, the $\boldsymbol{f}_k^*$ are centered for uniqueness. For VGLMs, the $\boldsymbol{f}_k$ are linear so that

$$\mathbf{B}^T \;=\; \left(\mathbf{H}_1\boldsymbol{\beta}_{(1)}^{*} \quad \mathbf{H}_2\boldsymbol{\beta}_{(2)}^{*} \quad \cdots \quad \mathbf{H}_p\boldsymbol{\beta}_{(p)}^{*}\right). \qquad (11)$$

Type	Name	Description
Categorical	`acat`	Adjacent categories model
	`cratio`	Continuation ratio model
	`cumulative`	Proportional and nonproportional odds model
	`multinomial`	Multinomial logit model
	`sratio`	Stopping ratio model
	`brat`	Bradley Terry model
	`bratt`	Bradley Terry model with ties
	`ABO`	ABO blood group system
Quantile regession	`lms.bcn`	Box-Cox transformation to normality
	`alsqreg`	Asymmetric least squares (expectiles on normal distribution)
	`amlbinomial`	Asymmetric maximum likelihood—for binomial
	`amlpoisson`	Asymmetric maximum likelihood—for Poisson
	`alaplace1`	Asymmetric Laplace distribution (1-parameter)
Counts	`negbinomial`	Negative binomial
	`zipoisson`	Zero-inflated Poisson
	`zinegbinomial`	Zero-inflated negative binomial
	`zibinomial`	Zero-inflated binomial
	`zapoisson`	Zero-altered Poisson (a hurdle model)
	`zanegbinomial`	Zero-altered negative binomial (a hurdle model)
	`mix2poisson`	Mixture of two Poissons
	`pospoisson`	Positive Poisson
Discrete distributions	`logarithmic`	logarithmic
	`skellam`	Skellam (Pois(λ_1)$-$ Pois(λ_2))
	`yulesimon`	Yule-Simon
	`zetaff`	Zeta
	`zipf`	Zipf
Continuous distributions	`betaff`	Beta (2-parameter)
	`betabinomial`	Beta-binomial
	`betaprime`	Beta-prime
	`dirichlet`	Dirichlet
	`gamma2`	Gamma (2-parameter)
	`gev`	Generalized extreme value
	`gpd`	Generalized Pareto
	`mix2normal1`	Mixture of two univariate normals
	`skewnormal1`	Skew-normal
	`vonmises`	Von Mises
	`weibull`	Weibull
Bivariate distributions	`amh`	Ali-Mikhail-Haq
	`fgm`	Farlie-Gumbel-Morgenstern
	`frank`	Frank
	`morgenstern`	Morgenstern
Bivariate binary responses	`binom2.or`	Bivariate logistic odds-ratio model
	`binom2.rho`	Bivariate probit model
	`loglinb2`	Loglinear model for 2 binary responses

Table 2: Some commonly used **VGAM** family functions. They have been grouped into types. Families in the classical exponential family are omitted here.

Vector splines and penalized likelihood

VGAMs employ *vector* smoothers in their estimation. One type of vector smoother is the *vector spline* which minimizes the quantity

$$\sum_{i=1}^{n} \{\boldsymbol{y}_i - \boldsymbol{f}(x_i)\}^T \boldsymbol{\Sigma}_i^{-1} \{\boldsymbol{y}_i - \boldsymbol{f}(x_i)\} \; + \sum_{j=1}^{M} \lambda_j \int_a^b \{f_j''(t)\}^2 \, dt \tag{12}$$

to scatterplot data $(x_i, \boldsymbol{y}_i, \boldsymbol{\Sigma}_i)$, $i = 1, \ldots, n$. Here, $\boldsymbol{\Sigma}_i$ are known symmetric and positive-definite error covariances, and $a < \min(x_1, \ldots, x_n)$ and $b > \max(x_1, \ldots, x_n)$. The first term of (12) measures lack of fit while the second term is a smoothing penalty. Equation (12) simplifies to an ordinary cubic smoothing spline when $M = 1$, (see, e.g., Green and Silverman, 1994). Each *component function* f_j has a non-negative smoothing parameter λ_j which operates as with an ordinary cubic spline.

Now consider the penalized likelihood

$$\ell - \frac{1}{2} \sum_{k=1}^{p} \sum_{j=1}^{M} \lambda_{(j)k} \int_{a_k}^{b_k} \{f_{(j)k}''(x_k)\}^2 \, dx_k. \tag{13}$$

Special cases of this quantity have appeared in a wide range of situations in the statistical literature to introduce cubic spline smoothing into a model's parameter estimation (for a basic overview of this roughness penalty approach see Green and Silverman, 1994). It transpires that the VGAM local scoring algorithm can be justified via maximizing the penalized likelihood (13) by vector spline smoothing (12) with $\mathbf{W}_i = \boldsymbol{\Sigma}_i^{-1}$.

From a practitioner's viewpoint the smoothness of a smoothing spline can be more conveniently controlled by specifying the degrees of freedom of the smooth rather than λ_j. Given a value of the degrees of freedom, the software can search for the λ_j producing this value. In general, the higher the degrees of freedom, the more wiggly the curve.

RR-VGLMs

Partition $\boldsymbol{x}$ into $(\boldsymbol{x}_1^T, \boldsymbol{x}_2^T)^T$ and $\mathbf{B} = (\mathbf{B}_1^T \;\; \mathbf{B}_2^T)^T$. In general, $\mathbf{B}$ is a dense matrix of full rank, i.e., rank $= \min(M, p)$. Thus there are $M \times p$ regression coefficients to estimate, and for some models and data sets this is "too" large.

One solution is based on a simple and elegant idea: replace $\mathbf{B}_2$ by a reduced-rank regression. This will cut down the number of regression coefficients enormously if the rank R is kept low. Ideally, the problem can be reduced down to one or two dimensions—a successful application of dimension reduction—and therefore can be plotted. The reduced-rank regression is applied to $\mathbf{B}_2$ because we want to make provision for some variables $\boldsymbol{x}_1$ that we want to leave alone, e.g., the intercept.

It transpires that *Reduced Rank VGLMs* (RR-VGLMs) are simply VGLMs where the constraint matrices are estimated. The modelling function `rrvglm()` calls an alternating algorithm which toggles between estimating two thin matrices $\mathbf{A}$ and $\mathbf{C}$, where $\mathbf{B}_2 = \mathbf{A}\,\mathbf{C}^T$.

Incidentally, special cases of RR-VGLMs have appeared in the literature. For example, a RR-multinomial logit model, or RR-MLM, is known as the *stereotype* model (Anderson, 1984). Another is Goodman (1981)'s RC model which is reduced-rank multivariate Poisson model. Note that the parallelism assumption of the proportional odds model (McCullagh and Nelder, 1989) can be thought of as a type of reduced rank regression where the constraint matrices are thin and known.

QRR-VGLMs and constrained ordination

RR-VGLMs form an optimal linear combination of the explanatory variables $\boldsymbol{x}_2$ and then fit a VGLM to these. Thus, in terms of (ecological) ordination, it performs a constrained linear ordination or CLO. Biotic responses are usually unimodal, and therefore this suggest fitting a quadratic on the $\boldsymbol{\eta}$ scale. This gives rise to the class of Quadratic RR-VGLMs, or QRR-VGLMs, so as to have constrained quadratic ordination or CQO. The result are bell-shaped curves/surfaces on axes defined in terms of gradients. The CQO approach here is statistically more rigorous than the popular canonical correspondence analysis (CCA) method. For more details see the references in Table 1.

Some user-oriented topics

Making the most of **VGAM** requires an understanding of the general framework described above plus a few other notions. Here are some of them.

Common arguments

VGAM family functions share a pool of common types of arguments, e.g., `exchangeable`, `nsimEIM`, `parallel`, `zero`. These are more convenient shortcuts for the argument `constraints` which accepts a named list of constraint matrices $\mathbf{H}_k$. For example, setting `parallel = TRUE` would constrain the coefficients $\beta_{(j)k}$ in (7) to be equal for all $j = 1, \ldots, M$, each separately for $k = 2, \ldots, p$. For some models the $\beta_{(j)1}$ are equal too. Another example is the `zero` argument which accepts a vector specifying which η_j is to be modelled as an intercept-only. Assigning it `NULL` means none. A last example is `nsimEIM`: some **VGAM** family functions use simulation to estimate the EIM and for these this argument controls the number of random variates generated.

Link functions

Most **VGAM** family functions allow a flexible choice of link functions to be assigned to each η_j. Table 3 lists some of them. Most **VGAM** family functions offer a link argument for each parameter plus an extra argument to pass in any value associated with the link function, e.g., an offset or power.

Table 3: Some **VGAM** link functions currently available.

Link	$g(\theta)$	**Range of** θ
`cauchit`	$\tan(\pi(\theta - \frac{1}{2}))$	$(0,1)$
`cloglog`	$\log_e\{-\log_e(1-\theta)\}$	$(0,1)$
`fisherz`	$\frac{1}{2}\log_e\{(1+\theta)/(1-\theta)\}$	$(-1,1)$
`fsqrt`	$\sqrt{2\theta} - \sqrt{2(1-\theta)}$	$(0,1)$
`identity`	θ	$(-\infty,\infty)$
`loge`	$\log_e(\theta)$	$(0,\infty)$
`logc`	$\log_e(1-\theta)$	$(-\infty,1)$
`logit`	$\log_e(\theta/(1-\theta))$	$(0,1)$
`logoff`	$\log_e(\theta + A)$	$(-A,\infty)$
`probit`	$\Phi^{-1}(\theta)$	$(0,1)$
`powl`	θ^p	$(0,\infty)$
`rhobit`	$\log_e\{(1+\theta)/(1-\theta)\}$	$(-1,1)$

Random variates and EIMs

Many `dpqr`-functions exist to return the density, distribution function, quantile function and random generation of their respective **VGAM** family function. For example, there are the `[dpqr]sinmad()` functions corresponding to the Singh-Maddala distribution **VGAM** family function `sinmad()`.

Incidentally, if you are a developer or writer of models/distributions where scoring is natural, then it helps writers of future **VGAM** family functions to have expressions for the EIM. A good example is Kleiber and Kotz (2003) whose many distributions have been implemented in **VGAM**. If EIMs are untractable then algorithms for generating random variates and an expression for the score vector is the next best choice.

Smart prediction (for interest only)

It is well known among seasoned R and S-PLUS users that `lm()` and `glm()`-type models used to have prediction problems. **VGAM** currently implements "smart prediction" (written by T. Yee and T. Hastie) whereby the parameters of parameter dependent functions are saved on the object. This means that functions such as `scale(x)`, `poly(x)`, `bs(x)` and `ns(x)` will *always* correctly predict. However, most R users are unaware that other R modelling functions will not handle pathological cases such as `I(poly(x))` and `bs(scale(x))`. In S-PLUS even terms such as `poly(x)` and `scale(x)` will not predict properly without special measures.

Examples

Here are some examples giving a flavour of **VGAM** that I hope convey some of its breadth (at the expense of depth); `http://www.stat.auckland.ac.nz/~yee/VGAM` has more examples as well as some fledging documentation and the latest prerelease version of **VGAM**.

Example 1: Negative binomial regression

Consider a simple example involving simulated data where the dispersion parameter is a function of x. Two independent responses are constructed. Note that the k parameter corresponds to the `size` argument.

```
set.seed(123)
x = runif(n <- 500)
y1 = rnbinom(n, mu=exp(3+x), size=exp(1+x))
y2 = rnbinom(n, mu=exp(2-x), size=exp(2*x))
fit = vglm(cbind(y1,y2) ~ x,
           fam = negbinomial(zero=NULL))
```

Then

```
> coef(fit, matrix=TRUE)
            log(mu1) log(k1) log(mu2) log(k2)
(Intercept)   2.96    0.741     2.10  -0.119
x             1.03    1.436    -1.24   1.865
```

An edited summary shows

```
> summary(fit)
...
Coefficients:
               Value Std. Error t value
(Intercept):1  2.958     0.0534   55.39
(Intercept):2  0.741     0.1412    5.25
```

```
(Intercept):3  2.097      0.0851    24.66
(Intercept):4 -0.119      0.1748    -0.68
x:1            1.027      0.0802    12.81
x:2            1.436      0.2508     5.73
x:3           -1.235      0.1380    -8.95
x:4            1.865      0.4159     4.48
```

so that all estimates are within two standard errors of their true values.

Example 2: Proportional odds model

Here we fit the POM on data from Section 5.6.2 of McCullagh and Nelder (1989).

```
data(pneumo)
pneumo = transform(pneumo,
                   let=log(exposure.time))
fit = vglm(cbind(normal,mild,severe) ~ let,
           cumulative(par=TRUE, rev=TRUE),
           data = pneumo)
```

Then

```
> coef(fit, matrix=TRUE)
            logit(P[Y>=2]) logit(P[Y>=3])
(Intercept)        -9.6761       -10.5817
let                 2.5968         2.5968
> constraints(fit)   # Constraint matrices
$‘(Intercept)‘
     [,1] [,2]
[1,]    1    0
[2,]    0    1

$let
     [,1]
[1,]    1
[2,]    1
```

The constraint matrices are the $\mathbf{H}_k$ in (10) and (11). The estimated variance-covariance matrix, and some of the fitted values and prior weights, are

```
>  vcov(fit)
          (Intercept):1 (Intercept):2    let
(Intercept):1     1.753         1.772 -0.502
(Intercept):2     1.772         1.810 -0.509
let              -0.502        -0.509  0.145
> cbind(fitted(fit),
        weights(fit, type="prior"))[1:4,]
  normal    mild  severe
1  0.994 0.00356 0.00243 98
2  0.934 0.03843 0.02794 54
3  0.847 0.08509 0.06821 43
4  0.745 0.13364 0.12181 48
```

Example 3: constraint matrices

How can we estimate θ given a random sample of n observations from a $N(\mu = \theta, \sigma = \theta)$ distribution? One way is to use the **VGAM** family function `normal1()` with the constraint that the mean and standard deviation are equal. Its default is $\eta_1 = \mu$ and $\eta_2 = \log\sigma$ but we can use the identity link function for σ and set $\mathbf{H}_1 = (1,1)^T$. Suppose $n = 100$ and $\theta = 10$. Then we can use

```
set.seed(123)
theta = 10
y = rnorm(n <- 100, mean = theta, sd = theta)
clist = list("(Intercept)" = rbind(1, 1))
fit = vglm(y ~ 1, normal1(lsd="identity"),
           constraints = clist)
```

Then we get $\widehat{\theta} = 9.7504$ from

```
> coef(fit, matrix = TRUE)
              mean     sd
(Intercept) 9.7504 9.7504
```

Consider a similar problem but from $N(\mu = \theta, \sigma^2 = \theta)$. To estimate θ make use of $\log\mu = \log\theta = 2\log\sigma$ so that

```
set.seed(123)
y2 = rnorm(n, theta, sd = sqrt(theta))
clist2 = list("(Intercept)" = rbind(1, 0.5))
fit2 = vglm(y2 ~ 1, normal1(lmean="loge"),
            constraints = clist2)
```

Then we get $\widehat{\theta} = 10.191$ from

```
> (cfit2 <- coef(fit2, matrix = TRUE))
            log(mean) log(sd)
(Intercept)    2.3215  1.1608
> exp(cfit2[1,"log(mean)"])
[1] 10.191
```

It is left to the reader as an exercise to figure out how to estimate θ from a random sample from $N(\mu = \theta, \sigma = 1 + e^{-\theta})$, for $\theta = 1$, say.

Example 4: mixture models

The 2008 World Fly Fishing Championships (WFFC) was held in New Zealand a few months ago. Permission to access and distribute the data was kindly given, and here we briefly look at the length of fish caught in Lake Rotoaira during the competition. A histogram of the data is given in Figure 1(a). The data looks possibly bimodal, so let's see how well a mixture of two normal distributions fits the data. This distribution is usually estimated by the EM algorithm but `mix2normal1()` uses simulated Fisher scoring.

```
data(wffc)
fit.roto = vglm(length/10 ~ 1, data=wffc,
    mix2normal1(imu2=45, imu1=25, ESD=FALSE),
    subset = water=="Rotoaira")
```

Here, we convert mm to cm. The estimation was helped by passing in some initial values, and we could have constrained the two standard deviations to be equal for parsimony. Now

```
> coef(fit.roto, matrix = TRUE)
            logit(phi)    mu1 log(sd1)
(Intercept)    -1.8990 25.671   1.4676
               mu2 log(sd2)
(Intercept) 46.473   1.6473
> Coef(fit.roto)
    phi     mu1     sd1     mu2     sd2
 0.1302 25.6706  4.3388 46.4733  5.1928
```

The resulting density is overlaid on the histogram in Figure 1(b). The LHS normal distribution doesn't look particularly convincing and would benefit from further investigation.

Finally, for illustration's sake, let's try the `untransform` argument of `vcov()`. Under limited conditions it applies the delta method to get at the untransformed parameters. The use of `vcov()` here is dubious (why?); its (purported?) standard errors are

```
> round(sqrt(diag(vcov(fit.roto,
        untransform=TRUE))), dig=3)
  phi   mu1   sd1   mu2   sd2
0.026 1.071 0.839 0.422 0.319
```

A more detailed analysis of the data is given in Yee (2008). The 2008 WFFC was a successful and enjoyable event, albeit we got pipped by the Czechs.

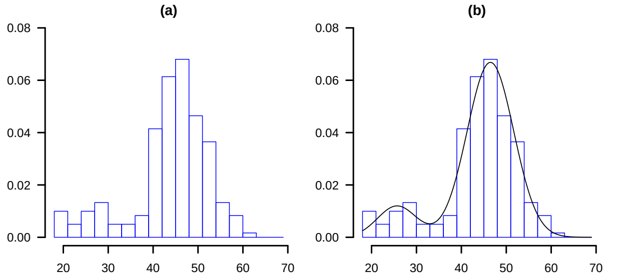

Figure 1: (a) Histogram of fish lengths (cm) at Lake Rotoaira. (b) Overlaid with the estimated mixture of two normal density functions. The sample size is 201.

Example 5: quantile & expectile regression

Over recent years quantile regression has become an important tool with a wide range of applications. The breadth of the VGLM/VGAM framework is illustrated by **VGAM** implementing *three* subclasses of quantile regression methods. We will only mention the two most popular subclasses here.

The first is of the LMS-type where a transformation of the response (e.g., Box-Cox) to some parametric distribution (e.g., standard normal) enables quantiles to be estimated on the transformed scale and then back-transformed to the original scale. In particular the popular Cole and Green (1992) method falls in this subclass. More details are in Yee (2004b).

The second is via expectile regression. Expectiles are almost as interpretable as quantiles because, given $X = x$, the percentile $\xi_\tau(x)$ specifies the position below which 100τ% of the (probability) mass of Y lies; while the expectile $\mu_\omega(x)$ determines, again given $X = x$, the point such that 100ω% of the mean distance between it and Y comes from the mass below it. The 0.5-expectile is the mean while the 0.5-quantile is the median. Expectile regression can be used to perform quantile regression by choosing ω that gives the desired τ. There are theoretical reasons why this is justified.

For normally distributed responses, expectile regression is based on *asymmetric least squares* (ALS) estimation, a variant of ordinary LS estimation. ALS estimation was generalized to *asymmetric maximum likelihood* (AML) estimation for members in the exponential family by Efron (1992). An example of this is the AML Poisson family which is illustrated in Figure 2. The data were generated by

```
set.seed(123)
alldat = data.frame(x=sort(runif(n<-500))+.2)
mymu = function(x)
    exp(-2 + 6*sin(2*x-0.2) / (x+0.5)^2)
alldat = transform(alldat,
                   y = rpois(n,mymu(x)))
```

Through trial and error we use

```
fit = vgam(y ~ s(x), data = alldat,
           amlpoisson(w=c(0.11, 0.9, 5.5)))
```

because the desired $\boldsymbol{\tau} = (0.25, 0.5, 0.75)^T$ obtained by

```
>  fit@extra[["percentile"]]
w.aml=0.11  w.aml=0.9  w.aml=5.5
      24.6       50.2       75.0
```

is to sufficient accuracy. Then Figure 2 was obtained by

```
with(alldat, plot(x, y, col="darkgreen"))
with(alldat, matlines(x, fitted(fit),
                      col="blue", lty=1))
```

If `parallel=TRUE` was set then this would avoid the embarrasing crossing quantile problem.

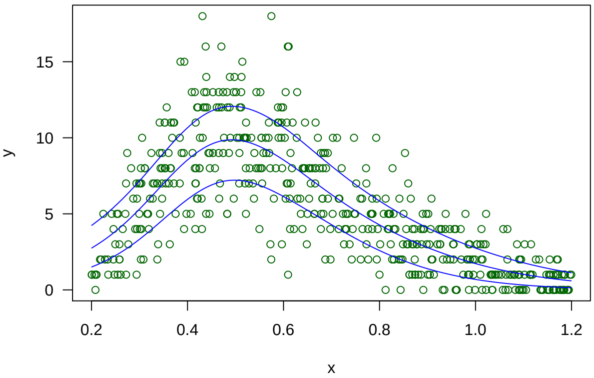

Figure 2: AML Poisson model fitted to the simulated Poisson data. The fitted expectiles correspond approximately to $\boldsymbol{\tau} = (0.25, 0.5, 0.75)^T$ and were fitted by smoothing splines.

Example 6: the bivariate logistic odds-ratio model

We consider the coalminers data of Table 6.6 of McCullagh and Nelder (1989). Briefly, 18282 working coalminers were classified by the presence of $Y_1 =$ breathlessness and $Y_2 =$ wheeze. The explanatory variable $x =$ age was available. We fit a nonparametric bivariate logistic odds-ratio model

$$\begin{aligned}\eta_1 &= \operatorname{logit} P(Y_1 = 1|x) = \beta_{(1)1} + f_{(1)1}(x), && (14)\\ \eta_2 &= \operatorname{logit} P(Y_2 = 1|x) = \beta_{(2)1} + f_{(2)1}(x), && (15)\\ \eta_3 &= \log \psi = \beta_{(3)1} + f_{(3)1}(x), && (16)\end{aligned}$$

where $\psi = p_{11}p_{00}/(p_{10}p_{01})$ is the odds ratio, $p_{jk} = P(Y_1 = j, Y_2 = k|x)$. It is a good idea to afford $f_{(3)1}(x)$ less flexibility, and so we use

```
data(coalminers)
fit.coal = vgam(cbind(nBnW,nBW,BnW,BW) ~
      s(age, df=c(4,4,3)),
      binom2.or(zero=NULL), coalminers)
mycols = c("blue","darkgreen","purple")
plot(fit.coal, se=TRUE, lcol=mycols,
     scol=mycols, overlay=TRUE)
```

to give Figure 3. It appears that the log odds ratio could be modelled linearly.

Were an exchangeable error structure true we would expect both functions $\widehat{f}_{(1)1}(x)$ and $\widehat{f}_{(2)1}(x)$ to be on top of each other. However it is evident that this is not so, relative to the standard error bands. Possibly a quadratic function for each is suitable.

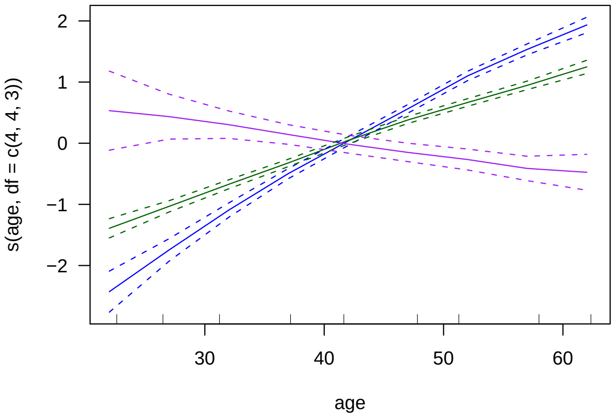

Figure 3: Bivariate logistic odds-ratio model fitted to the coalminers data. This is the VGAM (14)–(16) with centered components overlaid: $\widehat{f}_{(1)1}(x)$ in blue (steepest curve), $\widehat{f}_{(2)1}(x)$ in green, $\widehat{f}_{(3)1}(x)$ in purple (downward curve). The dashed curves are pointwise ±2 standard error bands

Example 7: the Gumbel model

Central to classical extreme value theory is the generalized extreme value (GEV) distribution of which the Gumbel is a special case when $\xi = 0$. Suppose the maxima is Gumbel and let $Y_{(1)}, \ldots, Y_{(r)}$ be the r largest observations such that $Y_{(1)} \geq \cdots \geq Y_{(r)}$. The joint distribution of

$$\left(\frac{Y_{(1)} - b_n}{a_n}, \ldots, \frac{Y_{(r)} - b_n}{a_n}\right)^T$$

has, for large n, a limiting distribution with density $f(y_{(1)}, \ldots, y_{(r)}; \mu, \sigma) =$

$$\sigma^{-r} \exp\left\{-\exp\left(-\frac{y_{(r)} - \mu}{\sigma}\right) - \sum_{j=1}^{r}\left(\frac{y_{(j)} - \mu}{\sigma}\right)\right\},$$

for $y_{(1)} \geq \cdots \geq y_{(r)}$. Upon taking logarithms, one can treat this as an approximate log-likelihood.

The **VGAM** family function `gumbel()` fits this model. Its default is $\boldsymbol{\eta}(x) = (\mu(x), \log \sigma(x))^T$. We apply the block-Gumbel model to the well-known Venice sea levels data where the 10 highest sea levels (in cm) for each year for the years $x = 1931$ to 1981 were recorded. Note that only the top 6 values are available in 1935. For this reason let's try using the top 5 values. We have

```
data(venice)
fit=vglm(cbind(r1,r2,r3,r4,r5)~I(year-1930),
         gumbel(R=365, mpv=TRUE, zero=2,
                lscale="identity"),
         data = venice)
```

giving

```
> coef(fit, mat=TRUE)
               location scale
(Intercept)     104.246  12.8
I(year - 1930)    0.458   0.0
```

This agrees with Smith (1986).

Now let's look at the first 10 order statistics and introduce some splines such as Rosen and Cohen (1996). As they do, let's assume no serial correlation.

```
y=as.matrix(venice[,paste("r",1:10,sep="")])
fit1 = vgam(y ~ s(year, df=3),
            gumbel(R=365, mpv=TRUE),
            data=venice, na.action=na.pass)
```

A plot of the fitted quantiles

```
mycols = 1:3
qtplot(fit1, mpv=TRUE, lcol=mycols,
       tcol=mycols, las=1, llwd=2,
       ylab="Venice sea levels (cm)",
       pcol="blue", tadj=0.1, bty="l")
```

produces Figure 4. The curves suggest a possible inflection point in the mid-1960s. The *median predicted value* (MPV) for a particular year is the value that the maximum of the year has an even chance of exceeding. As a check, the plot shows about 26 values exceeding the MPV curve—this is about half of the 51 year values.

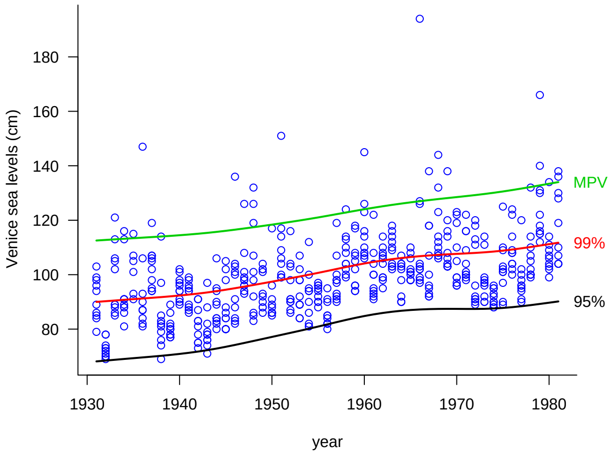

Figure 4: Block-Gumbel model fitted to the Venice sea levels data, smoothing splines are used.

More details about the application of VGLMs and VGAMs to extreme value data analysis are given in Yee and Stephenson (2007).

Example 8: nonlinear regression

The VGLM framework enables a Gauss-Newton-like algorithm to be performed in order to solve the nonlinear regression model

$$Y_i \quad = \quad f(\boldsymbol{u}_i; \boldsymbol{\theta}) + \varepsilon_i, \qquad i = 1, \ldots, n, \tag{17}$$

where $\boldsymbol{\theta}$ is an M-vector of unknown parameters and the ε_i are assumed to be i.i.d. $N(0, \sigma^2)$. Here, it is more convenient to use $\boldsymbol{u}$ to denote the regressors rather than the usual $\boldsymbol{x}$. An example of (17) is the Michaelis-Menten model

$$Y \;=\; \frac{\theta_1 \, u}{\theta_2 + u} + \varepsilon$$

and this is implemented in the **VGAM** family function `micmen()`. One advantage **VGAM** has over `nls()` for fitting this model is that `micmen()` is self-starting, i.e., initial values are automatically chosen.

Here is the Michaelis-Menten model fitted to some enzyme velocity and substrate concentration data (see Figure 5).

```
data(enzyme)
fit = vglm(velocity ~ 1, micmen, enzyme,
           form2 = ~ conc - 1)
with(enzyme, plot(conc, velocity, las=1,
                  xlab="concentration",
                  ylim=c(0,max(velocity)),
                  xlim=c(0,max(conc))))
with(enzyme, points(conc, fitted(fit),
                    col="red", pch="+"))

U = with(enzyme, max(conc))
newenzyme = data.frame(conc=seq(0,U,len=200))
fv = predict(fit, newenzyme, type="response")
with(newenzyme, lines(conc, fv, col="blue"))
```

The result is Figure 5.

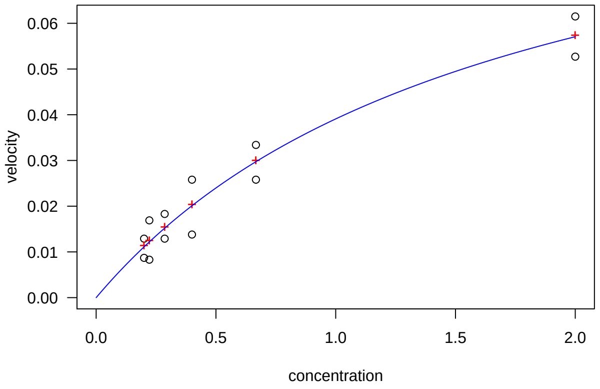

Figure 5: Michaelis-Menten model fitted to some enzyme data.

Example 9: constrained ordination

Ecologists frequently perform an ordination in order to see how multi-species and environmental data are related. An ordination enables one to display these data types simultaneously in an ordination plot.

Several methods have been developed of which one of the most common is canonical correspondence analysis (CCA). It was developed under the restrictive assumptions of a *species packing model* which stipulates that species have equal tolerances (a measure of niche width), equal maxima (a measure of abundance), and optima (optimal environments) that are uniformly distributed over the range of the gradient, which are linear combinations of the environmental variables (also known as latent variables).

CCA is a heuristic approximation to constrained quadratic ordination (CQO). The function `cqo()` implements CQO, meaning it assumes each species has a symmetric bell-shaped response with respect to underlying gradients. CQO does not necessarily make any of the species packing model assumptions.

A simple CQO model is where the responses are Poisson counts and there is one gradient. Then the model can be written like a Poisson regression

$$\begin{aligned} \log \mu_s \quad &= \quad \beta_{(s)1} + \beta_{(s)2}\, \nu + \beta_{(s)3}\, \nu^2 && (18)\\ &= \quad \alpha_s - \frac{1}{2}\left(\frac{\nu - u_s}{t_s}\right)^2. && (19)\end{aligned}$$

where $\nu = \boldsymbol{c}^T \boldsymbol{x}_2$ is the latent variable, and s denotes the species ($s = 1, \ldots, S$). The curves are unimodal provided $\beta_{(s)3}$ are negative. The second parameterization (19) has direct ecological interpretation: u_s is the species' optimum or *species score*, t_s is the tolerance, and $\exp(\alpha_s)$ is the maximum.

Here is an example using some hunting spiders data. The responses are 12 species' numbers trapped over a 60 week period and there are 6 environmental variables measured at 28 sites. The $\boldsymbol{x}$ are standardized prior to fitting.

```
data(hspider)
hspider[,1:6] = scale(hspider[,1:6])
p1 = cqo(cbind(Alopacce, Alopcune, Alopfabr,
               Arctlute, Arctperi, Auloalbi,
               Pardlugu, Pardmont, Pardnigr,
               Pardpull, Trocterr, Zoraspin)
         ~ WaterCon + BareSand + FallTwig +
         CoveMoss + CoveHerb + ReflLux,
         fam = poissonff, data = hspider,
         Crow1posit = FALSE, ITol = FALSE)
```

Then the fitted coefficients of (19) can be seen in

```
> coef(p1)

C matrix (constrained/canonical coefficients)
             lv
WaterCon -0.233
BareSand  0.513
FallTwig -0.603
CoveMoss  0.211
CoveHerb -0.340
ReflLux   0.798


...

Optima and maxima
         Optimum Maximum
Alopacce   1.679   19.29
Alopcune  -0.334   18.37
Alopfabr   2.844   13.03
Arctlute  -0.644    6.16
Arctperi   3.906   14.42
Auloalbi  -0.586   19.22
Pardlugu      NA      NA
Pardmont   0.714   48.60
Pardnigr  -0.530   87.80
Pardpull  -0.419  110.32
Trocterr  -0.683  102.27
Zoraspin  -0.742   27.24

Tolerance
            lv
Alopacce 1.000
Alopcune 0.849
Alopfabr 1.048
Arctlute 0.474
Arctperi 0.841
Auloalbi 0.719
Pardlugu    NA
Pardmont 0.945
Pardnigr 0.529
Pardpull 0.597
Trocterr 0.932
Zoraspin 0.708
```

All but one species have fitted bell-shaped curves. The $\hat{\nu}$ can be interpreted as a moisture gradient. The curves may be plotted with

```
S = ncol(p1@y) # Number of species
clr = (1:(S+1))[-7] # omits yellow
persp(p1, col=clr, llty=1:S, llwd=2, las=1)
legend("topright", legend=colnames(p1@y),
       col=clr, bty="n", lty=1:S, lwd=2)
```

and this results in Figure 6.

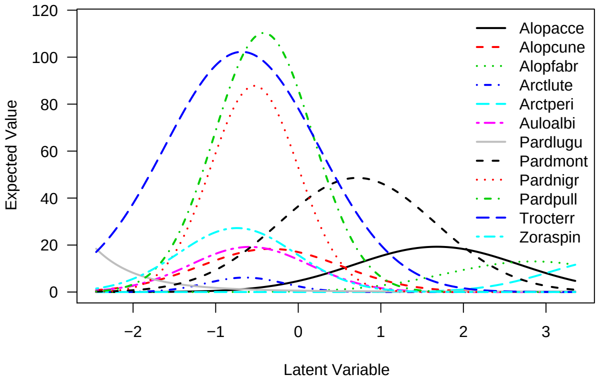

Figure 6: CQO Poisson model, with unequal tolerances, fitted to the hunting spiders data.

Currently `cqo()` will handle presence/absence data via `family = binomialff`. Altogether the present `cqo()` function is a first stab at ordination based on sound regression techniques. Since it fits by maximum likelihood estimation it is likely to be sensitive to outliers. The method requires very clear unimodal responses amongst the species, hence the x needs to be collected over a broad range of environments. All this means that it may be of limited use to 'real' biological data where messy data is common and robustness to outliers is needed.

Summary

Being a large project, **VGAM** is far from completion. Currently some of the internals are being rewritten and a monograph is in the making. Its continual development means changes to some details presented here may occur in the future. The usual maintenance, consisting of bug fixing and implementing improvements, applies.

In summary, the contributed R package **VGAM** is purposely general and computes the maximum likelihood estimates of many types of models and distributions. It comes not only with the capability to do a lot of things but with a large and unified framework that is flexible and easily understood. It is hoped that the package will be useful to many statistical practitioners.

Bibliography

A. Agresti. *Categorical Data Analysis*. Wiley, New York, second edition, 2002.

J. A. Anderson. Regression and ordered categorical variables (with discussion). *Journal of the Royal Sta-*

tistical Society, Series B, Methodological, 46(1):1–30, 1984.

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

T. J. Cole and P. J. Green. Smoothing reference centile curves: The LMS method and penalized likelihood. *Statistics in Medicine*, 11(10):1305–1319, 1992.

B. Efron. Poisson overdispersion estimates based on the method of asymmetric maximum likelihood. *Journal of the American Statistical Association*, 87(417):98–107, 1992.

B. Efron and D. V. Hinkley. Assessing the accuracy of the maximum likelihood estimator: Observed versus expected Fisher information. *Biometrika*, 65(3): 457–481, 1978.

L. A. Goodman. Association models and canonical correlation in the analysis of cross-classifications having ordered categories. *Journal of the American Statistical Association*, 76(374):320–334, 1981.

P. J. Green and B. W. Silverman. *Nonparametric Regression and Generalized Linear Models: A Roughness Penalty Approach*. Chapman & Hall, London, 1994.

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

C. Kleiber and S. Kotz. *Statistical Size Distributions in Economics and Actuarial Sciences*. Wiley-Interscience, Hoboken, NJ, USA, 2003.

P. McCullagh and J. A. Nelder. *Generalized Linear Models*. Chapman & Hall, London, second edition, 1989.

O. Rosen and A. Cohen. Extreme percentile regression. In W. Härdle and M. G. Schimek, editors, *Statistical Theory and Computational Aspects of Smoothing: Proceedings of the COMPSTAT '94 Satellite Meeting held in Semmering, Austria, 27–28 August 1994*, pages 200–214, Heidelberg, 1996. Physica-Verlag.

R. L. Smith. Extreme value theory based on the r largest annual events. *Journal of Hydrology*, 86(1–2):27–43, 1986.

T. W. Yee. A new technique for maximum-likelihood canonical Gaussian ordination. *Ecological Monographs*, 74(4):685–701, 2004a.

T. W. Yee. Quantile regression via vector generalized additive models. *Statistics in Medicine*, 23(14):2295–2315, 2004b.

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

T. W. Yee. Vector generalized linear and additive models, with applications to the 2008 World Fly Fishing Championships. *In preparation*, 2008.

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

T. W. Yee and A. G. Stephenson. Vector generalized linear and additive extreme value models. *Extremes*, 10(1–2):1–19, 2007.

T. W. Yee and C. J. Wild. Vector generalized additive models. *Journal of the Royal Statistical Society, Series B, Methodological*, 58(3):481–493, 1996.

Thomas W. Yee
Department of Statistics
University of Auckland
New Zealand
`t.yee@auckland.ac.nz`