The theory underlying the software can be found in Yee and Wild (1996), Yee (1998), Yee and Mackenzie (2002), Yee and Hastie (2003), Yee (2004a), Yee (2004b), Yee (2006), Yee and Stephenson (2007), Yee (2008), Yee and Dirnböck (2009), Yee (2010a), Yee (2010b), Yee et al. (2015), Yee and Hadi (2014), Yee (2015). These references contain additional examples etc. Please cite the appropriate references if you use the software!

The theory underlying the software can be found in Yee and Wild (1996), Yee (1998), Yee and Mackenzie (2002), Yee and Hastie (2003), Yee (2004a), Yee (2004b), Yee (2006), Yee and Stephenson (2007), Yee (2008), Yee and Dirnböck (2009), Yee (2010a), Yee (2010b), Yee et al. (2015), Yee and Hadi (2014), Yee (2015). These references contain additional examples etc. Please cite the appropriate references if you use the software!


DISCLAIMER: VGAM is still in the development stage, meaning that new features are still being added, and bugs are still being found on a regular basis. Function and argument names may change at any time, as well as default values of arguments; see the NEWS file. The VGAM package is available on a use-at-your-own-risk basis: the Author assumes no liability for loss or damage of any kind resulting from its use.

Note that, in VGAM 0.9-5, numerous argument names, their order, and function names are changed. This was due to inconsistencies I detected while writing my book, and it was deemed necessary to standardize things. This particularly relates to family functions implementing discrete and continuous distributions. I have tried to summarize all changes in the NEWS file, however, a few changes might have gone undocumented. Users of previous versions of VGAM are cautioned to check their code. My apologies for this inconvenience.

Modelling functions
The following functions form the heart of the VGAM package and use formula and family arguments, e.g., vglm(y ~ 1, family = maxwell, data = mdata).

- `vglm()` Vector generalized linear models.
- `vgam()` Vector generalized additive models.
- `rrvglm()` Reduced-rank vector generalized linear models (same as constrained linear ordination, or CLO).
- `cqo()` Constrained quadratic (Gaussian) ordination (QRR-VGLM).
- `cao()` Constrained additive ordination (RR-VGAM).
- `rcim()` Row–column interaction models (developmental stage only).

The following functions do not use the formula, family = arguments.

- `grc()` Goodman’s RC model.
- `vsmooth.spline()` Vector smoothing spline.

Extractor functions

- `AIC()` computes the Akaike information criterion or AIC.
- `BIC()` computes the Bayesian (Schwarz’s) information criterion or BIC.
- `class()` The object’s class.
- `coef()` Regression coefficients (the $\beta_i$ in (1) but enumerated in a different order).
- `Coef()` Regression coefficients, especially if the formula comprises of intercept only, i.e., ~ 1.
- `constraints()` Constraint matrices, $H_k$.
- `depvar()` Dependent variable (response), $Y$.
- `deviance()` Deviance, $D$.
- `df.residual()` Residual degrees of freedom.
- `fitted()` Fitted values, usually $\hat{\mu}_i$.
- `hatvalues()` Hat (or projection) matrix, $H$.
- `is.bell()` Are the response curve of a CQO bell-shaped?
- `is.parallel()` Are the $H_k = I_M$?
- `is.zero()` Are the $\eta_j$ intercept-only?
- `logLik()` Log-likelihood, $\ell$.
- `lrtest()` Likelihood ratio test.
- `linkfun()` Parameter link functions.
- `model.matrix()` The big model matrix $X_{VLM}$. This is never smaller than the lm-type model matrix.
- `nobs()` Number of observations, $n_{VLM}$ or $n_{LM}$.
- `npred()` Number of linear/additive predictors $\eta_j, M$.
- `QR.Q()` The Q matrix in the QR decomposition of the relevant model matrix used in the IRLS algorithm.
- `QR.R()` The R matrix in the QR decomposition of the relevant model matrix used in the IRLS algorithm.
- `predict()` Prediction of $\mu$ or $\eta$ from x.
- `print()` Print.
- `Rank()` Rank of a reduced-rank (ordination) model, R.
- `resid()` Residuals (working, Pearson, deviance, response).
- `show()` Show.
- `summary()` Summary.
- `terms()` Terms.
- `vcov()` Variance-covariance matrix, $\text{Var}(\hat{\beta})$.
- `weights()` Prior weights $w_i$, and working weights ($w_iW_i$), in matrix-band format.

Plotting functions

Not all of the following apply to a fitted model.

- `biplot()` Biplot for RR-VGLMs.
- `deplot()` Density plot, e.g., for quantile regression.
lvplot() Latent variable plot (ordination diagram).
persp() Perspective (3-D; sometimes 2-D) plot.
plot() General plotting function.
guplot() Gumbel plot, e.g., for extreme values regression.
meplot() Mean excess plot, e.g., for extreme values regression.
qtplot() Quantile plot, e.g., for quantile regression.
rlplot() Return level plot, e.g., for extreme values regression.
trplot() Trajectory plot, e.g., for constrained ordination.

**Link functions**
Most parameters \( \theta_j \) are transformed into a linear/additive predictor \( \eta_j = \beta_j^T x \) or \( \eta_j = \sum_{k=1}^B f_{ijk}(x_k) \). All logarithms are to base \( e \) unless specified otherwise.

Usage: use full link function name (in quotes is ok), e.g., `vglm(y ~ 1, family = maxwell(link = "identitylink"))`.

- `log()` Log, \( \log(\theta) > 0 \).
- `loglog()` Log-log link function, \( \log(\log(\theta)) \), \( 0 < \theta < 1 \).
- `negreciprocal()` Negative-reciprocal, \( -1/\theta, \theta \neq 0 \).
- `fisherz()` Fisher’s Z-transformation, \( 1/2 \log \left( 1 + \frac{1}{\theta - 1} \right) \).
- `rhobit()` Twice Fisher’s Z-transformation, \( 1 + \theta \).
- `cauchit()` Cauchit link function, \( \frac{1}{1 + \theta} \).
- `logit()` Logit link function, \( \log \left( \frac{\theta}{1 - \theta} \right) \).
- `loglog() Log-log link function, \( \log(\log(\theta)) \).
- `negidentity()` Negative-identity, \( -\theta \).
- `reciprocal()` Reciprocal, \( 1/\theta, \theta \neq 0 \).
- `m2default()` Conversion from weight matrices (matrix-band format) to array format.
- `select()` Selects columns from a data frame, can output formulas. A little similar to `subset()`.
- `simulate()` Simulates new observations coming from the fitted model.

**Utility functions**
- `eta2theta()` \( \eta_j = g^{-1}(\theta_j) \).
- `theta2eta()` \( \theta_j = g(\eta_j) \).
- `m2a.default()` Conversion from weight matrices (matrix-band format) to array format.
- `select()` Selects columns from a data frame, can output formulas. A little similar to `subset()`.
- `simulate()` Simulates new observations coming from the fitted model.

**Common arguments in family functions**
- `zero` allows for \( \eta_j = \beta_{j1} x \), i.e., intercept-only. Can be assigned a vector with values from the set \{1,2,⋯,M\}. Negative values allowed for multiple responses. A `NULL` means all the \( \eta_j \) are modelled as functions of the co-variates.
- `exchangeable` For some \( s \) and \( t \), \( \eta_s - \eta_t = 0 \) or some constant. That is, \( \eta_s \) and \( \eta_t \) differ at most by an intercept.
- `parallel` If `TRUE`, for all \( k \), \( \beta_{ijk} = \beta_{ij} \) for some \( s \) and \( t \); i.e., the slope for \( X_k \) of some of the linear/additive predictors are parallel or differ by a constant.
- `parallel = FALSE \sim X1 + X2 + 1 means all terms, except for \( X_1 \) and \( X_2 \), have parallel slopes.

Next is a listing of VGAM family functions. These are assigned to the `family =` argument to the modelling functions, e.g., `vglm(y ~ x2 + x3, family = multinomial)`. The characters “dpqr” are placed within the parentheses to denote whether `dpqr`-type functions are available. Those `dpqr`-type functions from base R are wrapped in brackets `[]`.

**Generalized linear models**
- `binomialff()` Binomial. For multivariate responses use `binomialff(mv = TRUE)`.
- `quasibinomialff()` Binomial with dispersion parameter to be estimated.
- `gaussianff([dpqr])` Gaussian or normal.
- `inverse.gaussianff([dpqr])` Inverse Gaussian.
- `poissonff([dpqr])` Poisson.
- `quasipoissonff()` Poisson with dispersion parameter to be estimated.
- `quasif()` Quasi- family (not working yet).

**Nonlinear regression models**
- `micmen()` Michaelis-Menten model, \( \mu_i = \theta_1 x_i / (\theta_2 + x_i) \).

**Categorical data**
In the following, \( g \) is a link function, \( \eta_j \) the linear/additive predictors, and \( Y \in \{1,\ldots,M+1\} \) is a categorical response.
- `acat()` Adjacent categories model, \( \eta_j = g(\sum_{k=j}^M P[Y = k]) \).
- `cumulative()` Cumulative categories model, \( \eta_j = g(\sum_{k=1}^j P[Y = k]) \). Includes the proportional odds model. Derivatives available through `margeff()`.
- `propodds()` Proportional odds model \( \eta_j = \logit(P[Y > j]) \).
- `cratio()` Continuation ratio model, \( \eta_j = g(\sum_{k=j}^M P[Y \geq k]) \).
- `sratio()` Stopping ratio model, \( \eta_j = g(\sum_{k=1}^j P[Y = j | Y \geq k]) \).
- `multinomial()` Multinomial logit model, \( \eta_j = \log(P[Y = j] / P[Y = M + 1]) \). Derivatives available through `margeff()`.
- `brat()` Bradley Terry model (without ties).
- `bratt()` Bradley Terry model (with ties).
- `ordpoisson()` Ordinal Poisson model.

The argument `reverse` reverses the direction of many of the above probabilities. See documentation on the `xij` argument at the package’s website: this handles covariates that have different values for differing linear/additive predictors \( \eta_j \). This will handle consumer choice or discrete choice models.

**Bivariate responses**
- `binom2.or(dr)` Bivariate logistic/probit/… odds ratio model for two binary responses.
- `binom2.rho(dr)` Bivariate probit model for two binary responses. Based on a standard \( N_2 \) with correlation parameter \( \rho \).
Bivariate distributions

bilogistic4(dpr) 4-parameter bivariate logistic distribution.

amh(dpr) Ali-Mikhail-Haq’s bivariate distribution.
biclaytoncop(dr) Bivariate Clayton copula distribution.
bifrankcop(dpr) Bivariate Frank’s copula distribution.
bistudentt(d) Bivariate Student- t distribution.

loglinb2() loglinb3() Loglinear model for two/three binary responses.

Quantile and expectile regression

I) LMS methods
lms.bcn() Box-Cox transformation to normality.
lms.bcg() Box-Cox transformation to the gamma distribution.
lms.yjn() Yeo-Johnson transformation to normality.

Special methods functions for these models are:
qplot() Quantile plot.
deplot() Density plot.
cdf() Cumulative distribution function.

II) Asymmetric Laplace distribution (ALD) methods
alaplace1(dpqr) 1-parameter ALD.
alaplace2(dpqr) 2-parameter ALD.
alaplace3(dpqr) 3-parameter ALD.

III) Asymmetric Maximum Likelihood (AML) estimation methods
aminormal() Asymmetric least squares (expectile) regression (Efron, 1991).
ambinomial() Logistic (expectile) regression.
amlexponential() Exponential (expectile) regression.
ampoisson() Poisson (expectile) regression (Efron, 1992).

Genetical data

AA.Aa.aa() AA-Aa-aa blood group system.
AB.Ab.aB.ab2() AB-Ab-aB-ab2 blood group system.
AB.Ab.AB.ab() AB-Ab-AB-ab blood group system.
ABO() ABO blood group system.
G1G2G3() G1G2G3 blood group system.
MNSs() MNSs blood group system.

Positive and zero-inflated or zero-altered distributions

Here, positive distributions have the zero probability that the response is 0. Zero-altered models are also called hurdle models.

posbernoulli.[b,t,] (dr) Positive Bernoulli distribution for closed-population capture–recapture experiments based on the conditional likelihood.

Univariate continuous distributions

Additionally, some of these distributions have random variates generation, density functions, cumulative distribution functions and quantile functions associated with them. For example, 
rsinmad(). dlomax(). pdagum(). qfisk(). Those distribution with d, p, q, r functions have characters inside their arguments, e.g., sinmad(r).

alaplace[123](dpqr) Asymmetric Laplace distribution.
benini(dpqr) 1-parameter Benini distribution.
betaR([dpqr]) 2-parameter beta distribution.
betaff([dpqr]) 2-parameter beta distribution.
betaprime() 2-parameter beta-prime distribution.
betaII() 3-parameter beta II distribution.
bisap(dpr) Birnbaum-Saunders distribution.
cardiod(dpqr) Cardiod distribution.
caucchy(dpqr) 2-parameter Cauchy distribution.
caucyh1(dpqr) 1-parameter Cauchy distribution.
chisq(dpqr) Chi-squared distribution.
cens.gumbel() Censored Gumbel distribution.
cens.normal() Censored univariate normal distribution.
cens.rayleigh() Censored Rayleigh distribution.
dagum(dpqr) 3-parameter Dagum distribution.
double.cens.normal() Double censored 2-parameter univariate normal distribution.
erlang() Erlang distribution.
exepxp() 2-parameter Exponentiated exponential distribution.
exepxp1() 2-parameter Exponentiated exponential distribution (using a profile (concentrated) likelihood).
explogff(dpqr) Exponential logarithmic distribution.
exponential(dpqr) Exponential distribution.
fff(dpqr) F-distribution.
fisk(dpqr) 2-parameter Fisk distribution.
foldnormal(dpqr) Folded normal distribution (univariate and generalized).
frechct(dpqr) 2-parameter Fréchet distribution.
gamma1(dpqr) 1-parameter gamma distribution.
gamma2(dpqr), gammar(dpqr) 2-parameter gamma distribution.
genbetaII() 4-parameter generalized beta II distribution.
gengamma.stacy(dpqr) Generalized gamma distribution.
gompertz(dpqr) 2-parameter Gompertz distribution.
hypersecant(), hypersecant01() Hyperbolic secant distribution.
inv.gaussianff(dpr) 2-parameter inverse Gaussian distribution.
inv.lomax(dpqr) 2-parameter inverse Lomax distribution.
inv.paralogistic(dpqr) 2-parameter inverse paralogistic distribution.
kumar(dpqr) Kumaraswamy distribution.
laplace(dpqr) Laplace distribution.
leipnik() Leipnik distribution.
levy(dpqr) Lévy distribution.
lgamma1(dpqr) 1-parameter log-gamma distribution.
lgamma3(dpqr) 3-parameter log-gamma distribution.
lindley(dpr) Lindley distribution.
linon(dpqr) 3-parameter generalized beta distribution (Libby and Novick, 1982).
logistic1(dpqr) 1-parameter logistic distribution.
logistic(dpqr) 2-parameter logistic distribution.
lognormal1(dpqr) 2-parameter lognormal distribution.
lognormal3(dpqr) 3-parameter lognormal distribution.
lomax(dpqr) Lomax distribution.
makeham(dpqr) Makeham distribution.
maxwell(dpqr) Maxwell distribution.
mccullagh89() McCullagh’s (1989) distribution.
nakagami(dpqr) Nakagami distribution.
normal.vcm() Linear model with varying-coefficients.
paralogistic(dpqr) 2-parameter paralogistic distribution.
paretoff(dpqr) Pareto distribution (Pareto(I)).
paretoIIV(dpqr) Pareto(IV) distribution.
paretoIII(dpqr) Pareto(III) distribution.
paretoII(dpqr) Pareto(II) distribution.
perks(dpqr) Perks’ distribution.
poisson.points(d) Distances to a fixed point, in a Poisson plane or volume.
rayleigh(dpqr) Rayleigh distribution.
riceff(dpqr) Rice distribution.
rigff() Reciprocal inverse Gaussian distribution.
sc.studentt2(dpqr) Scaled Student’s t2 distribution.
simplex(dr) 2-parameter simplex distribution.
simmad(dpqr) 3-parameter Singh-Maddala distribution.
skewnormal(dr) 1-parameter univariate skew-normal distribution.
studentt((dpqr)), studentt2((dpqr)), studentt3((dpqr)) Student t distribution.
tikuv(dpqr) Short-tailed symmetric distribution of Tikue and Vaughan (1999).
tobit() Tobit model.
triangle(dpqr) Triangle distribution.
truncpareto(dpqr) Truncated (upper) Pareto distribution (Pareto(I)).
truncweibull() Truncated Weibull distribution.
uninormal((dpqr)) 2-parameter univariate normal distribution.
vomises() von Mises distribution.
waldff() Standard Wald distribution.
weibull() 2-parameter Weibull distribution.

Univariate discrete distributions

[dpqr]benf() 0-parameter Benford distribution.
betabinomial(dpr) Beta-binomial distribution.
betabinomialff(dpr) Beta-binomial distribution.
betageometric(dpqr) Beta-geometric distribution.
borel.tanner(dr) Borel-Tanner distribution.
double.expbinomial() Double exponential binomial distribution.
genpoisson() Generalized Poisson distribution.
geometric(dpqr) Geometric distribution.
hzeta(dpqr) Haight’s zeta function.
logff(dpqr) Logarithmic distribution.
inv.binomial() Inverse binomial distribution.
negbinomial(dpqr) Negative binomial distribution with parameters µ and k.
negbinomial.size(dpqr) Negative binomial distribution with parameter µ and known k.
–polono(dpr) Poisson-lognormal distribution (no family function available yet.)
polya(dpqr) Pólya (negative binomial) distribution with parameters p and k.
seq2binomial() 2-stage sequential binomial distribution.
truncgeometric() Truncated geometric distribution.
yulesimon(dpr) Yule-Simon distribution.
zettaff() Zeta distribution.
zipf(dp) Zipf distribution.

Finite mixture models

mix2exp() Two exponential distributions.
mix2normal() Two univariate normals.
mix2poisson() Two Poisson distributions.

Miscellaneous models and distributions

DeLury() De Lury’s model for fish depletion analysis [in VGAMdata].
rrar() Reduced-rank autoregressive model for multiple time series.
dirichlet() Dirichlet distribution.
dirmultinomial(), dirmul.old() Dirichlet-multinomial distribution.
huber(dpqr) Huber’s robust regression method.
SUR() Seemingly unrelated regressions.
Miscellaneous mathematical functions

erf()  Error function, \( \text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-t^2) \, dt \).
erfc()  Complementary error function, \( 1 - \text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty \exp(-t^2) \, dt \).
expint()  Exponential integral, \( \int_0^x \exp(-t) \, dt, x > 0 \).
exppexpint()  Exponential integral, \( \exp(-x) \text{Ei}(x), x > 0 \).
expint.E1()  Exponential integral, \( \int_0^x \exp(-t^2) \, dt, x > 0 \).
lambertW()  Lambert’s \( W \) function for \( W(z) \exp(W(z)) = z \).
lerch()  Lerch’s \( \Phi(x,s,v) \) function.
pgamma.derr()  First 2 derivatives of the incomplete gamma.
pgamma.derr.unscaled()  First 2 derivatives of the incomplete gamma integral.
pnorm2()  CDF of the unscaled incomplete gamma integral.
pgamma.deriv()  First 2 derivatives of the incomplete gamma integral.
pgamma.deriv.unscaled()  First 2 derivatives of the unscaled incomplete gamma integral.
poissonff()  Fully finished yet.
binomialff()  VGLM).

Quadratic and additive ordination

cqo()  Canonical quadratic (Gaussian) ordination (QRR-VGAM).
cao()  Constrained additive ordination (RR-VGAM). Not fully finished yet.
The fast algorithm currently works for families
poissonff()  and \( \text{binomialff} \) (logit and cloglog links available).

Special methods functions for these models are:

\[ \text{Coef()} \]  \( \hat{A}, \hat{B}, \hat{C}, \hat{D}, \hat{u}_i, \hat{T}_i, \hat{v}_i, \) etc.
\[ \text{concoef()} \]  Constrained (canonical) coefficients \( \hat{C} \)
latvar()  Latent variables \( \hat{v}_i = \hat{C} \cdot \hat{x}_{2i} \) (site scores)
Max()  Maxima \( E[Y|\hat{u}_i] = g^{-1}(\hat{\alpha}_i) \)
Opt()  Optima \( \hat{\alpha}_i \) (species scores)
Tol()  Tolerances \( \hat{T}_i \)
lvplot()  Latent variable plot (ordination diagram; for rank \( R = 1 \) or 2)
persp()  Perspective (3-D; sometimes 2-D) plot
calibrate()  Calibration: estimate \( \nu \) from \( y \)
trplot()  Trajectory plot (for \( R = 1 \) only)

Miscellaneous

\[ \text{methods}(\text{class} = \text{class}(\text{fit})) \]  lists all the methods to handle objects of class \( \text{fit} \).

\[ \text{slotNames}(\text{fit}) \]  lists the slots of the object \( \text{fit} \), but it is best to use extractor functions where possible.

Some formulae

For most VGAMs the log-likelihood \( \ell = \sum_{j=1}^n w_j \ell_j(\eta_1, \ldots, \eta_M) \) is maximized, where \( \eta_j = \beta_j^T x \). The prior weights \( w_i \) are inputted using \( \text{vglm(..., weights = ...)} \).

For VGAMs (Yee and Hastie, 2003):

\[ \eta(x) = H_1 \beta_1^T x + \cdots + H_p \beta_p^T x_p = B^T x \]  (1)

where \( H_1, \ldots, H_p \) are known full-column rank constraint matrices, and \( \beta^T_k \) is a vector containing a possibly reduced set of unknown regression coefficients. With no constraints at all, \( H_k = I_M \) for all \( k \). Usually \( x_1 = 1 \) (intercept term). In general,

\[ B^T = (H_1 \beta_1^T \cdots H_p \beta_p^T) . \]  (2)

Then \( \text{coef(fit, matrix = TRUE)} \) is the estimate of \( B \), and \( \text{constraints(fit)} \) are the \( H_k \). The \( H_k \) can be inputted with \( \text{vglm(..., constraints = \text{list("(Intercept)" = \ldots, x2 = \ldots)})} \), or with arguments such as \( \text{parallel} = \text{list("(Intercept)" = \ldots, x2 = \ldots)} \)
with \( M \) \( k \)-vectors of smooth functions of \( x_k \) (estimated by a vector smoothing spline). With no constraints, \( \eta_j = \sum_{k=1}^M f_{ij}^{(k)}(x_k) \).

For RR-VGLMs (Yee and Hastie, 2003; Yee, 2014a):

\[ \eta(x) = B_1^T x_1 + A \nu \]  (4)

where \( x = (x_1^T, x_2^T)^T, \nu = C^T x_2 \) is a vector of latent variables, \( A \) is \( M \times R \) and \( C \) is \( p_2 \times R \). Here, \( A \) and \( C \) are estimated, and \( B = (B_1^T, B_2^T)^T \) with \( B_2 = CA^T \), a reduced-rank approximation of a subset of \( B \) (cf. (1)). The \( rank \ R \) is often 1 or 2, maybe 3 . . . .

For QRR-VGLMs (Yee, 2004a):

\[ \eta(x) = B_1^T x_1 + A \nu + \sum_{j=1}^M (v^T D_j \nu) e_j \]  (5)

where \( e_j \) is a vector of zeros but with a one in the \( j \)-th position, and \( D_j \) is \( R \times R \) symmetric matrices. Then \( T_p = -\frac{1}{2}D_j^{-1} \) are tolerance matrices.

For RCIMs (Yee and Hadi, 2014): these are RR-VGLMs applied to \( Y \) (no \( X \)!), with

\[ g_1(\theta_1) = \eta_{1ij} = \mu + \alpha_i + \gamma_j + \sum_{r=1}^R c_{iu} a_r, \]  (6)

where \( R \leq \min(M, p_2) \). The other parameters \( \theta_2, \ldots \) are usual intercept-only.

For a rank-1 CAO (Yee, 2006): these are RR-VGAMs with

\[ g_1(\mu_{iq}) = \eta_{iq} = f_q^{(k)}(\nu_{ij}) \]  (7)

where \( q = 1, \ldots, Q; \eta_{ij} = (\eta_{i1}, \ldots, \eta_{iQ})^T, \nu_{ij} = c^T x_i \) is a latent variable or site score, and the \( f_q \) are estimated by a smoothing spline. Only Poisson and Bernoulli responses are handled currently, and for rank-1 only too.

References


URL http://www.jstatsoft.org/v32/i10/


