Monday, March 24

Distributions for Over-dispersion

One way to model over-dispersion is to assume a model of the form

$$g[E(Y_i)] = \beta_0 + \beta_1 x_{i1} + \cdots + \beta_k x_{ik} + \zeta_i,$$

where ζ_i is an *unobserved* unit-specific random quantity that represents one or more unobserved explanatory variables that vary over units.

The Negative Binomial Distribution

Suppose that Y_i has a Poisson distribution *conditional* on ζ_i, and e^{ζ_i} has a *gamma* distribution such that $E(e^{\gamma_i}) = 1$ and $\text{Var}(e^{\gamma_i}) = \alpha > 0$. The *marginal* distribution of Y_i is then a *negative binomial distribution*, with mean structure

$$g[E(Y_i)] = \eta_i,$$

and variance structure

$$\text{Var}(Y_i) = E(Y_i) + \alpha E(Y_i)^2 \geq E(Y_i).$$

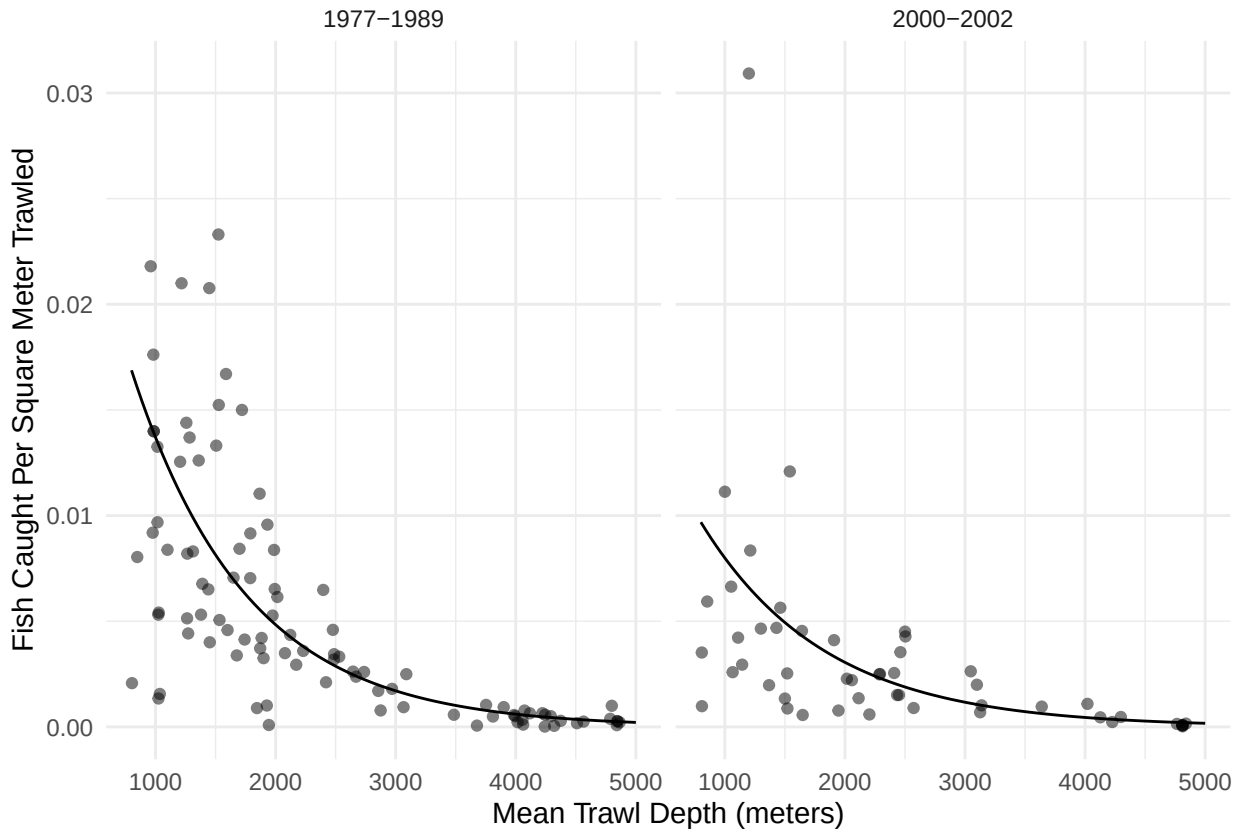
The Poisson distribution is a special case where $\alpha = 0$. This variance structure *does not* have the form

$$\text{Var}(Y_i) = \phi V[E(Y_i)]$$

unless α is known (which it normally is not), so this model is not a traditional GLM. But we can make inferences using maximum likelihood.

Example: Consider our model for the trawl fishing data. Here we will consider a negative binomial regression model.

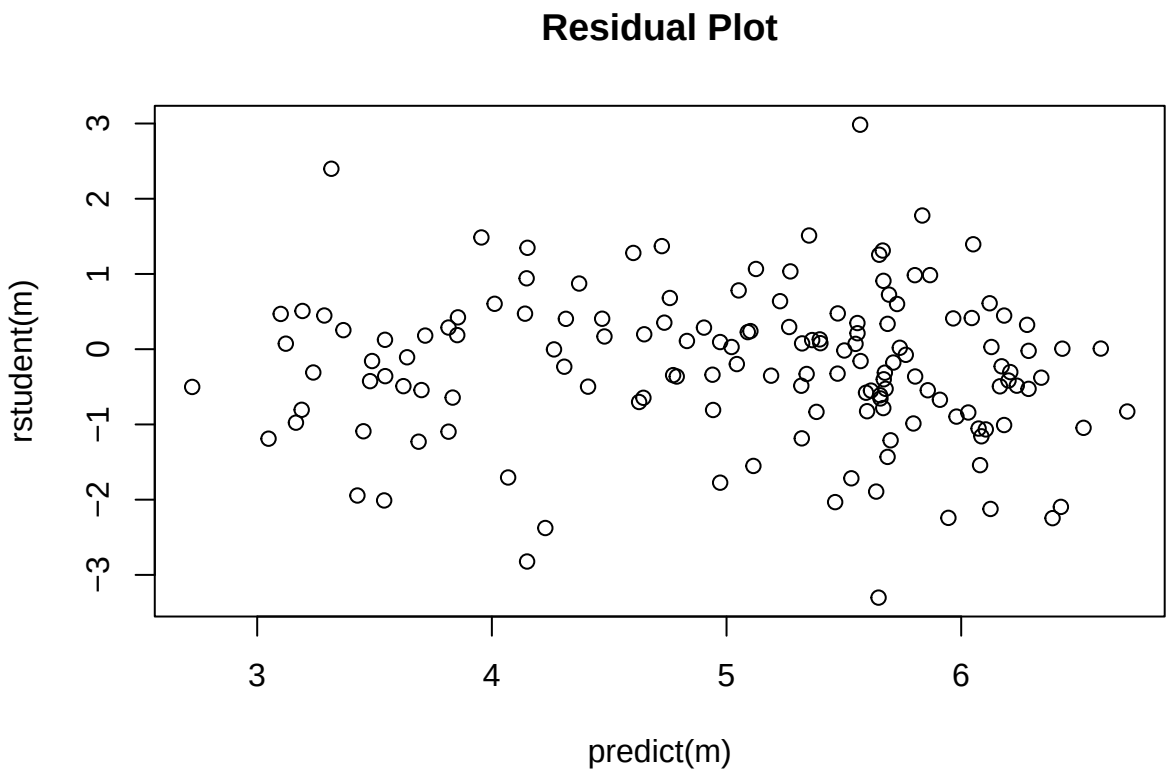
```
library(COUNT)
data(fishing)

library(MASS) # for the glm.nb function (note there is no family argument)

m <- glm.nb(totabund ~ period * meandepth + offset(log(sweptarea)),
  link = log, data = fishing)

d <- expand.grid(sweptarea = 1, period = levels(fishing$period),
  meandepth = seq(800, 5000, length = 100))
d$yhat <- predict(m, newdata = d, type = "response")

p <- ggplot(fishing, aes(x = meandepth, y = totabund/sweptarea)) +
  geom_point(alpha = 0.5) + facet_wrap(~ period) + theme_minimal() +
  labs(x = "Mean Trawl Depth (meters)",
    y = "Fish Caught Per Square Meter Trawled") +
  geom_line(aes(y = yhat), data = d)
plot(p)
```




```
summary(m) # note that what glm.nb calls theta equals 1/alpha
```

```
Call:
glm.nb(formula = totabund ~ period * meandepth + offset(log(sweptarea)),
    data = fishing, link = log, init.theta = 1.961162176)

Coefficients:
                          Estimate Std. Error z value Pr(>|z|)
(Intercept)               -3.25e+00   1.59e-01  -20.40   <2e-16 ***
period2000-2002           -6.19e-01   2.73e-01   -2.27    0.023 *
meandepth                 -1.04e-03   5.92e-05  -17.58   <2e-16 ***
period2000-2002:meandepth  7.95e-05   1.01e-04    0.79    0.432
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(1.96) family taken to be 1)

    Null deviance: 471.79  on 146  degrees of freedom
Residual deviance: 159.31  on 143  degrees of freedom
AIC: 1763

Number of Fisher Scoring iterations: 1

              Theta:  1.961
          Std. Err.:  0.219

 2 x log-likelihood:  -1752.713
```

```
plot(predict(m), rstudent(m), main = "Residual Plot")
```



Interestingly inferences based on the negative binomial model are very similar to those obtained using quasi-likelihood assuming the variance structure $V(Y_i) = \phi E(Y_i)^2$. Here are the parameter estimates, standard errors, and confidence intervals.

```
m.negbn <- glm.nb(totabund ~ period * meandepth + offset(log(sweptarea)),
  link = log, data = fishing)
m.quasi <- glm(totabund ~ period * meandepth + offset(log(sweptarea)),
  family = quasi(link = "log", variance = "mu^2"), data = fishing)
cbind(summary(m.negbn)$coefficients, confint(m.negbn))
```

```
                          Estimate Std. Error z value Pr(>|z|)     2.5 %    97.5 %
(Intercept)              -3.25e+00   1.59e-01 -20.404 1.53e-92 -3.560398 -2.927410
period2000-2002          -6.19e-01   2.73e-01  -2.269 2.32e-02 -1.181541 -0.043616
meandepth                -1.04e-03   5.92e-05 -17.584 3.25e-69 -0.001158 -0.000921
period2000-2002:meandepth  7.95e-05   1.01e-04   0.785 4.32e-01 -0.000133  0.000295
```

```
cbind(summary(m.quasi)$coefficients, confint(m.quasi))
```

```
                          Estimate Std. Error t value Pr(>|t|)     2.5 %    97.5 %
(Intercept)              -3.25e+00   1.59e-01 -20.418 3.19e-44 -3.560966 -2.929477
period2000-2002          -6.04e-01   2.72e-01  -2.221 2.79e-02 -1.167273 -0.028795
meandepth                -1.04e-03   5.87e-05 -17.740 5.99e-38 -0.001155 -0.000922
period2000-2002:meandepth  7.27e-05   9.99e-05   0.728 4.68e-01 -0.000138  0.000287
```

Here are the estimates of the rate ratios for period at several different depths.

```
library(trtools)
trtools::contrast(m.negbn,
  a = list(meandepth = c(1000,2000,3000,4000,5000), period = "2000-2002", sweptarea = 1),
  b = list(meandepth = c(1000,2000,3000,4000,5000), period = "1977-1989", sweptarea = 1),
```

```
  cnames = c("1000m","2000m","3000m","4000m","5000m"), tf = exp)
```

```
      estimate lower upper
1000m    0.583 0.402 0.847
2000m    0.632 0.487 0.819
3000m    0.684 0.518 0.902
4000m    0.740 0.493 1.112
5000m    0.802 0.449 1.430
```

```
trtools::contrast(m.quasi,
  a = list(meandepth = c(1000,2000,3000,4000,5000), period = "2000-2002", sweptarea = 1),
  b = list(meandepth = c(1000,2000,3000,4000,5000), period = "1977-1989", sweptarea = 1),
  cnames = c("1000m","2000m","3000m","4000m","5000m"), tf = exp)
```

```
      estimate lower upper
1000m    0.588 0.405 0.854
2000m    0.632 0.487 0.821
3000m    0.680 0.517 0.893
4000m    0.731 0.491 1.090
5000m    0.786 0.446 1.387
```

Here are the tests (likelihood ratio and F) for the "effect" of period. The null model assumes that expected abundance per unit area trawled is the same each period at a given depth. Put another way, the null model assumes that the rate ratio for period is one for all depths.

```
m.negbn.null <- glm.nb(totabund ~ meandepth + offset(log(sweptarea)),
  link = log, data = fishing)
anova(m.negbn.null, m.negbn)
```

```
Likelihood ratio tests of Negative Binomial Models

Response: totabund
                                        Model theta Resid. df    2 x log-lik.   Test    df LR stat.
1          meandepth + offset(log(sweptarea))  1.83       145           -1764                      
2 period * meandepth + offset(log(sweptarea))  1.96       143           -1753 1 vs 2     2     11.1
  Pr(Chi)
1        
2 0.00387
```

```
m.quasi.null <- glm(totabund ~ meandepth + offset(log(sweptarea)),
  family = quasi(link = "log", variance = "mu^2"), data = fishing)
anova(m.quasi.null, m.quasi, test = "F")
```

```
Analysis of Deviance Table

Model 1: totabund ~ meandepth + offset(log(sweptarea))
Model 2: totabund ~ period * meandepth + offset(log(sweptarea))
  Resid. Df Resid. Dev Df Deviance    F Pr(>F)
1       145       90.5                       
2       143       84.5  2     5.94 5.74  0.004 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Note: When using `anova` for a negative binomial model (estimated using the `glm.nb` function) we omit the `test = "LRT"` option which we use for generalized linear models. Somewhat confusingly, the `anova` function will do a likelihood ratio test for a `glm.nb` object, but will throw an error if we try to change the test type (even if we ask for a likelihood ratio test).

Heteroscedastic Consistent (Robust) Standard Errors

An alternative is to accept that the specified variance structure is incorrect and estimate standard errors in a way that provides *consistent* estimates despite the misspecification of the variance structure.[1]

Note: I needed to specify the data set as `trtools::rotifer` below as there is a data set of the same name in another package that was loaded earlier. It's actually the same data but in a different format from the data frame in the **trtools** package.

Example: Consider the logistic regression model for the `rotifer` data from the **trtools** package.

```
m <- glm(cbind(y, total - y) ~ species + density + species:density,
  family = binomial, data = trtools::rotifer)
```

Here are the parameter estimates and standard errors, with and without using the robust standard error estimates.

```
library(sandwich) # for the vcovHC function
library(lmtest)   # for coeftest and coefci functions
cbind(summary(m)$coefficients, confint(m))
```

```
                 Estimate Std. Error z value  Pr(>|z|)   2.5 %  97.5 %
(Intercept)       -114.35       4.03 -28.345 9.53e-177 -122.42 -106.60
speciespm            4.63       6.60   0.702  4.83e-01   -8.46   17.43
density            108.75       3.86  28.191 7.53e-175  101.33  116.46
speciespm:density   -3.08       6.33  -0.486  6.27e-01  -15.35    9.49
```

```
cbind(coeftest(m, vcov = vcovHC), coefci(m, vcov = vcovHC))
```

```
                 Estimate Std. Error z value Pr(>|z|)  2.5 % 97.5 %
(Intercept)       -114.35       18.3  -6.245 4.24e-10 -150.2  -78.5
speciespm            4.63       29.9   0.155 8.77e-01  -54.0   63.2
density            108.75       17.5   6.214 5.18e-10   74.4  143.0
speciespm:density   -3.08       28.8  -0.107 9.15e-01  -59.6   53.4
```

An alternative to using `coeftest` and `coefci` is `lincon(m, fcov = vcovHC)`. Now compare our inferences for the odds ratios for the effect of a 0.01 increase in density.

```
trtools::contrast(m,
  a = list(density = 0.02, species = c("kc","pm")),
  b = list(density = 0.01, species = c("kc","pm")),
  cnames = c("kc","pm"), tf = exp)
```

```
   estimate lower upper
kc     2.97  2.75  3.20
pm     2.88  2.61  3.17
```

```
trtools::contrast(m,
  a = list(density = 0.02, species = c("kc","pm")),
  b = list(density = 0.01, species = c("kc","pm")),
  cnames = c("kc","pm"), tf = exp, fcov = vcovHC)
```

```
   estimate lower upper
kc     2.97  2.11  4.18
pm     2.88  1.84  4.51
```

Using the **emmeans** package.

[1] Consistency is a rather technical condition, but roughly speaking a *consistent estimator* is one such that its sampling distribution becomes increasingly concentrated around the value being estimated as n increases.

```
library(emmeans)
pairs(emmeans(m, ~density|species, at = list(density = c(0.02,0.01))),
  type = "response", infer = TRUE)
```

```
species = kc:
 contrast                  odds.ratio    SE  df asymp.LCL asymp.UCL null z.ratio p.value
 density0.02 / density0.01       2.97 0.114 Inf      2.75      3.20    1  28.190  <.0001

species = pm:
 contrast                  odds.ratio    SE  df asymp.LCL asymp.UCL null z.ratio p.value
 density0.02 / density0.01       2.88 0.144 Inf      2.61      3.17    1  21.060  <.0001

Confidence level used: 0.95
Intervals are back-transformed from the log odds ratio scale
Tests are performed on the log odds ratio scale
```

```
pairs(emmeans(m, ~density|species, at = list(density = c(0.02,0.01))),
  type = "response", infer = TRUE, vcov = vcovHC)
```

```
species = kc:
 contrast                  odds.ratio    SE  df asymp.LCL asymp.UCL null z.ratio p.value
 density0.02 / density0.01       2.97 0.114 Inf      2.75      3.20    1  28.190  <.0001

species = pm:
 contrast                  odds.ratio    SE  df asymp.LCL asymp.UCL null z.ratio p.value
 density0.02 / density0.01       2.88 0.144 Inf      2.61      3.17    1  21.060  <.0001

Confidence level used: 0.95
Intervals are back-transformed from the log odds ratio scale
Tests are performed on the log odds ratio scale
```

For comparison consider also the results when using quasi-likelihood.

```
m <- glm(cbind(y, total - y) ~ species + density + species:density,
  family = quasibinomial, data = trtools::rotifer)
cbind(summary(m)$coefficients, confint(m))
```

```
                 Estimate Std. Error t value Pr(>|t|)  2.5 % 97.5 %
(Intercept)        -114.35       15.0  -7.647 4.74e-09 -146.0  -87.0
speciespm             4.63       24.5   0.189 8.51e-01  -46.2   51.3
density             108.75       14.3   7.606 5.36e-09   82.6  139.0
speciespm:density    -3.08       23.5  -0.131 8.96e-01  -47.8   45.7
```

```
trtools::contrast(m,
  a = list(density = 0.02, species = c("kc","pm")),
  b = list(density = 0.01, species = c("kc","pm")),
  cnames = c("kc","pm"), tf = exp)
```

```
   estimate lower upper
kc     2.97  2.22  3.96
pm     2.88  1.97  4.20
```

Recall that heteroscedastic consistent standard errors are best used with generous sample sizes. For modest sample sizes (such as this experiment) quasi-likelihood is probably better.

Generalized Linear Models Revisited

Recall that a generalized linear model (GLM) has the form

$$g[E(Y_i)] = \underbrace{\beta_0 + \beta_1 x_{i1} + \beta_1 x_{i2} + \cdots + \beta_k x_{ik}}_{\eta_i},$$

where g is the *link function* and η_i is the *linear predictor* or *systematic component*. This is the *mean structure* of the model.

The *variance structure* of a GLM is

$$\text{Var}(Y_i) = \phi V[E(Y_i)],$$

where ϕ is a *dispersion parameter* and V is the *variance function*.

If we define $h = g^{-1}$ so that $E(Y_i) = h(\eta_i)$ we can write a GLM concisely as

$$E(Y_i) = h(\eta_i) \tag{1}$$
$$\text{Var}(Y_i) = \phi V[h(\eta_i)] \tag{2}$$

to define the *mean structure* and a *variance structure* for Y_i, respectively, by specifying the mean and variance of Y_i to be functions of $x_{i1}, x_{i2}, \ldots, x_{ik}$.

The specification of a generalized linear model therefore requires three components.

1. The *systematic component* $\eta_i = \beta_0 + \beta_1 x_{i1} + \beta_1 x_{i2} + \cdots + \beta_k x_{ik}$.
2. The *link function* g for the mean structure $g[E(Y_i)] = \eta_i$.
3. The *distribution* of the response variable Y_i, which implies the variance structure $\text{Var}(Y_i) = \phi V[E(Y_i)]$, or we can specify the variance structure *directly*.

Four common distributions from the exponential family of distributions (normal/Gaussian, Poisson, gamma, and inverse-Gaussian) imply variance structures of the form

$$\text{Var}(Y_i) = \phi E(Y_i)^p$$

The values of p are $p = 0$ (normal/Gaussian), $p = 1$ (Poisson if $\phi = 1$), $p = 2$ (gamma), and $p = 3$ (inverse-Gaussian). Also note that when using quasi-likelihood we can use other values of p via the `tweedie` function from the **statmod** package.

GLMs for Gamma-Distributed Response Variables

If Y_i has a *gamma* distribution then Y_i is a positive and continuous random variable, and $\text{Var}(Y_i) = \phi E(Y_i)^2$. Such models are sometimes suitable for response variables that are bounded below by zero and right-skewed. Common link functions include the *log* and *inverse* functions. With a log link function we have a mean structure like that for Poisson regression where

$$\log E(Y_i) = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik},$$

or

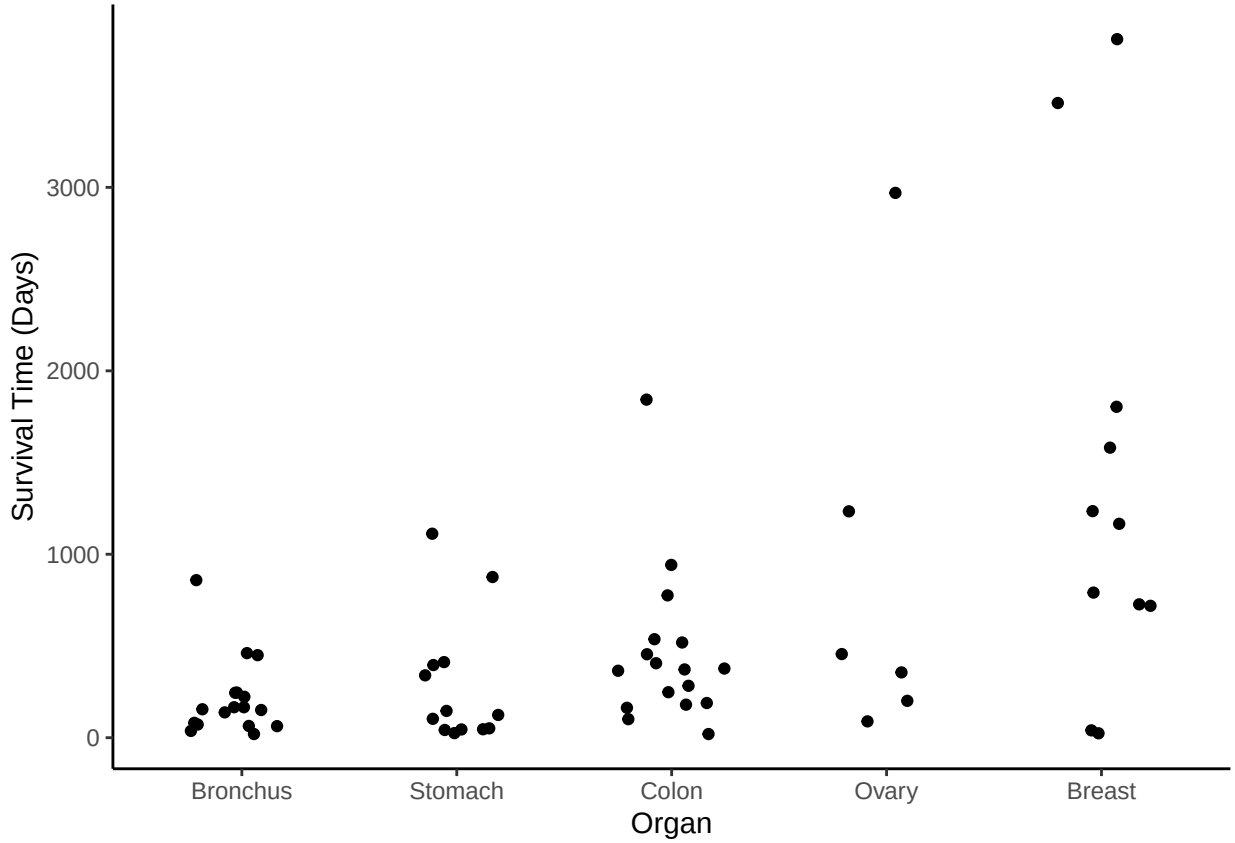
$$E(Y_i) = \exp(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_k x_{ik}),$$

so the effects of explanatory variables and contrasts can be interpreted by applying the exponential function e^x and interpreting the effects as multiplicative factors or percent increase/decrease or percent larger/smaller.

Example: Consider again the cancer survival time data.

```
library(Stat2Data)
data(CancerSurvival)
CancerSurvival$Organ <- with(CancerSurvival, reorder(Organ, Survival, mean))
```

```
p <- ggplot(CancerSurvival, aes(x = Organ, y = Survival)) +
  geom_jitter(height = 0, width = 0.25) +
  labs(y = "Survival Time (Days)") + theme_classic()
plot(p)
```



A gamma model might be appropriate here. First consider a model with a log link function.

```
m <- glm(Survival ~ Organ, family = Gamma(link = log), data = CancerSurvival)
cbind(summary(m)$coefficients, confint(m))
```

```
             Estimate Std. Error t value Pr(>|t|)   2.5 % 97.5 %
(Intercept)     5.355      0.250  21.385 1.77e-29  4.9010   5.89
OrganStomach    0.301      0.380   0.792 4.31e-01 -0.4404   1.07
OrganColon      0.771      0.354   2.177 3.35e-02  0.0699   1.47
OrganOvary      1.430      0.490   2.917 4.99e-03  0.5246   2.49
OrganBreast     1.887      0.399   4.723 1.48e-05  1.1157   2.70
```

We might compare the survival times to the type of cancer with lowest expected survival time.

```
trtools::contrast(m, tf = exp,
  a = list(Organ = c("Stomach","Colon","Ovary","Breast")),
  b = list(Organ = "Bronchus"),
  cnames = paste(c("Stomach","Colon","Ovary","Breast"), "/", "Bronchus", sep = ""))
```

```
                estimate lower upper
Stomach/Bronchus     1.35 0.631  2.89
Colon/Bronchus       2.16 1.064  4.39
Ovary/Bronchus       4.18 1.567 11.15
Breast/Bronchus      6.60 2.966 14.67
```

Now suppose we specify the same variance structure directly. Note that the results are *identical*.

```
m <- glm(Survival ~ Organ, family = quasi(link = log, variance = "mu^2"), data = CancerSurvival)
cbind(summary(m)$coefficients, confint(m))
```

```
             Estimate Std. Error t value Pr(>|t|)   2.5 % 97.5 %
(Intercept)     5.355      0.250  21.385 1.77e-29  4.9010   5.89
OrganStomach    0.301      0.380   0.792 4.31e-01 -0.4404   1.07
OrganColon      0.771      0.354   2.177 3.35e-02  0.0699   1.47
OrganOvary      1.430      0.490   2.917 4.99e-03  0.5246   2.49
OrganBreast     1.887      0.399   4.723 1.48e-05  1.1157   2.70
```

```
trtools::contrast(m, tf = exp,
  a = list(Organ = c("Stomach","Colon","Ovary","Breast")),
  b = list(Organ = "Bronchus"),
  cnames = paste(c("Stomach","Colon","Ovary","Breast"), "/", "Bronchus", sep = ""))
```

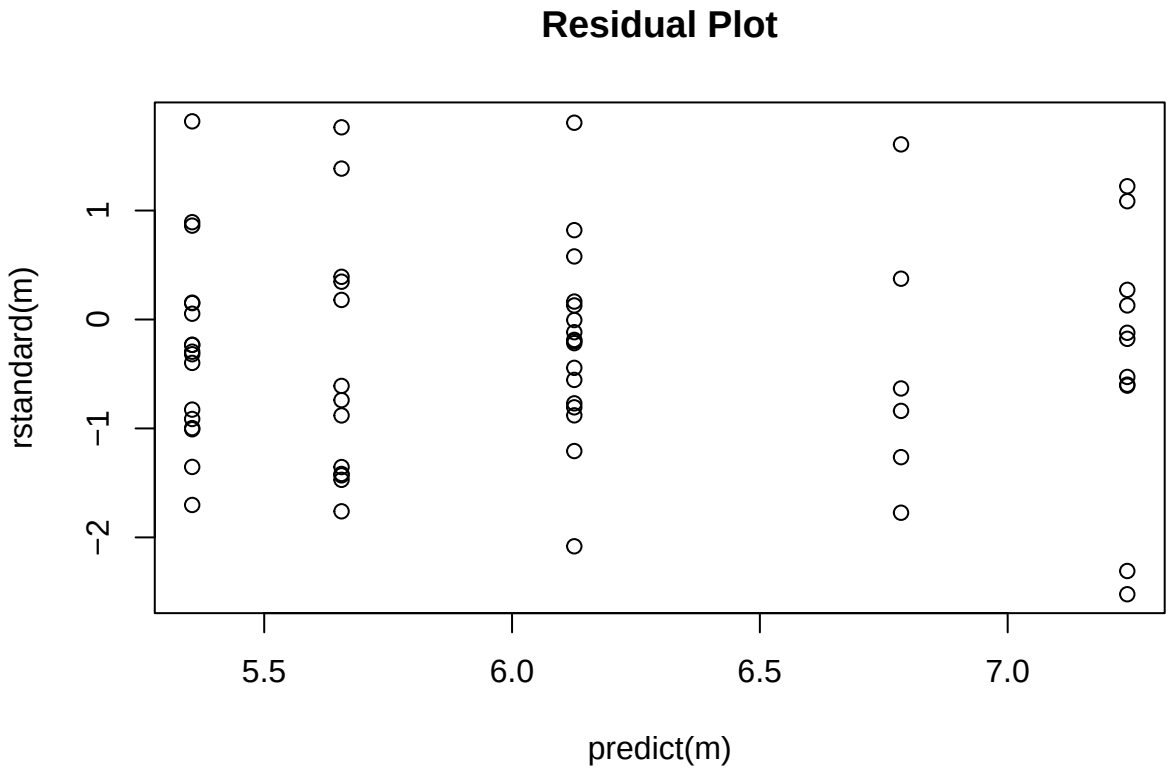
```
                estimate lower upper
Stomach/Bronchus     1.35 0.631  2.89
Colon/Bronchus       2.16 1.064  4.39
Ovary/Bronchus       4.18 1.567 11.15
Breast/Bronchus      6.60 2.966 14.67
```

```
emmeans::contrast(emmeans(m, ~Organ, type = "response"),
  method = "trt.vs.ctrl", ref = 1, infer = TRUE,
  adjust = "none", df = m$df.residual)
```

```
 contrast           ratio    SE df lower.CL upper.CL null t.ratio p.value
 Stomach / Bronchus  1.35 0.514 59    0.631     2.89    1   0.790  0.4310
 Colon / Bronchus    2.16 0.765 59    1.064     4.39    1   2.180  0.0330
 Ovary / Bronchus    4.18 2.050 59    1.567    11.15    1   2.920  0.0050
 Breast / Bronchus   6.60 2.640 59    2.966    14.67    1   4.720  <.0001

Degrees-of-freedom method: user-specified
Confidence level used: 0.95
Intervals are back-transformed from the log scale
Tests are performed on the log scale
```

Naturally we should check the residuals to see if the variance structure is reasonable.

```
plot(predict(m), rstandard(m), main = "Residual Plot")
```

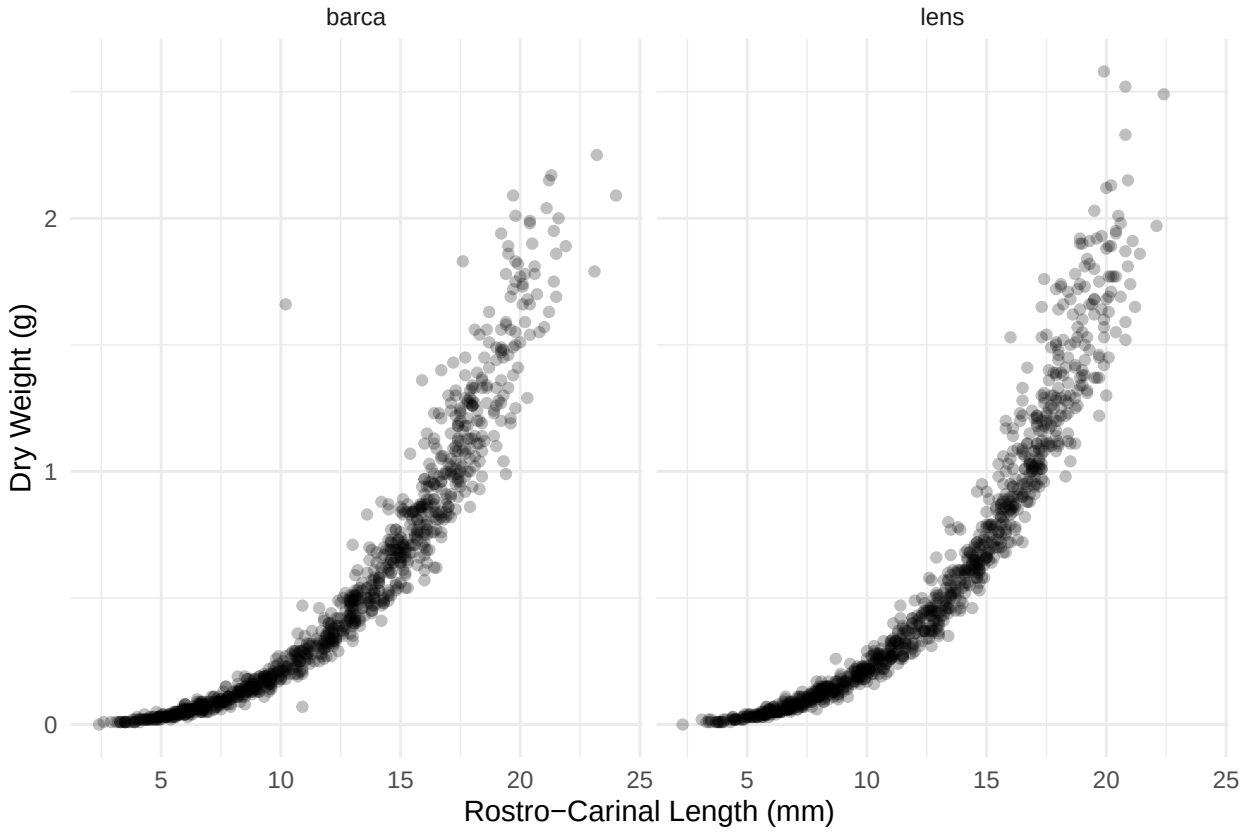


Example: Consider the following observations of dry weight (in grams) and rostro-carinal length (in mm) of a species of barnacles sampled from the inter-tidal zones near Punta Lens and Punta de la Barca along the Atlantic coast of Spain.

```
library(npregfast)
head(barnacle)
```

```
    DW   RC     F
1 0.14  9.5 barca
2 0.00  2.4 barca
3 0.42 13.1 barca
4 0.01  3.7 barca
5 0.03  5.6 barca
6 1.56 18.6 barca
```

```
p <- ggplot(barnacle, aes(x = RC, y = DW)) + theme_minimal() +
  geom_point(alpha = 0.25) + facet_wrap(~ F) +
  labs(x = "Rostro-Carinal Length (mm)", y = "Dry Weight (g)")
plot(p)
```



A common allometric regression model would have the form

$$E(Y_i) = ax_i^b$$

where Y_i is the dry weight for the i-th observation, and x_i is the rostro-carinal length for the i-th observation. We can also write this as

$$\log E(Y_i) = \log a + b \log x_i$$

or, equivalently,

$$E(Y_i) = \exp(\log a + b \log x_i)$$

or

$$E(Y_i) = \exp(\beta_0 + \beta_1 \log x_i)$$

where $\beta_0 = \log a$ and $\beta_1 = b$. This is basically a log-linear model since we can write

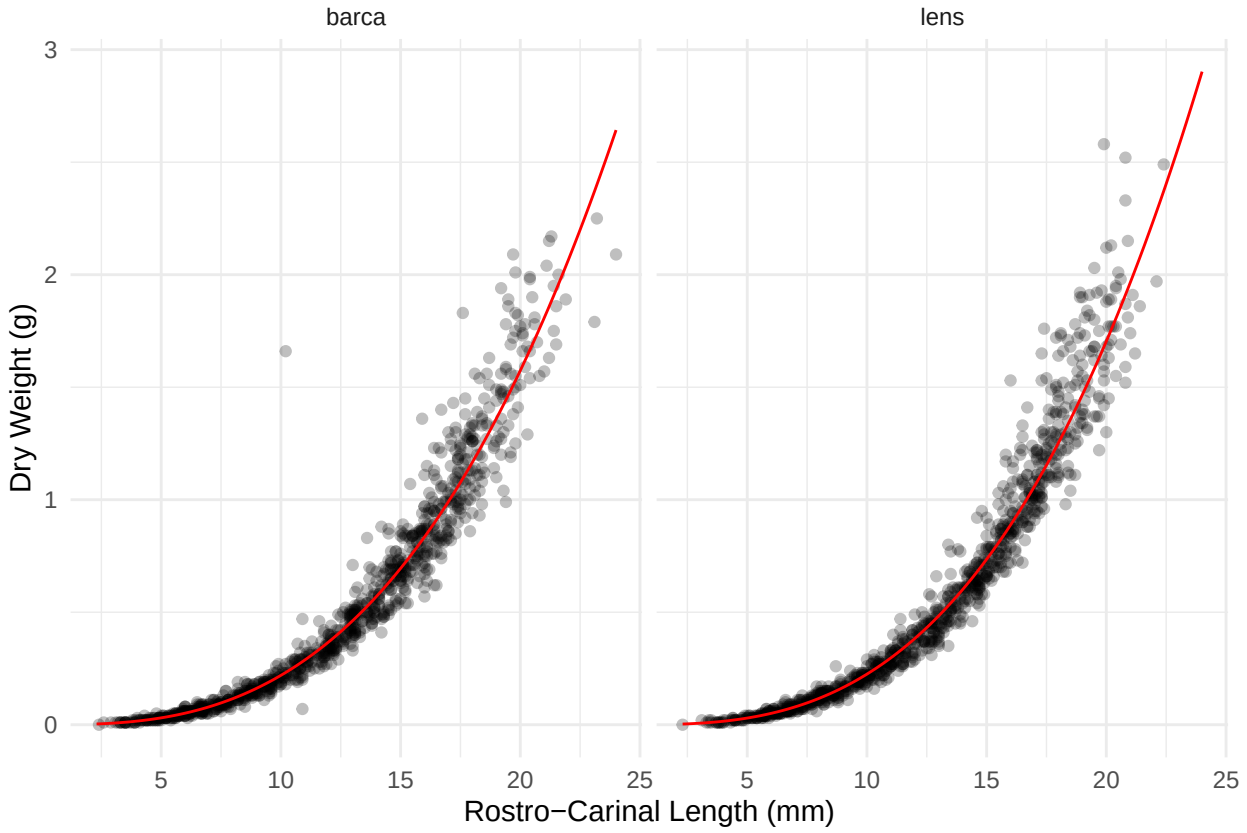
$$\log E(Y_i) = \beta_0 + \beta_1 \log x_i.$$

Because dry weight is continuous and positive, with the variability appearing to increase with the expected dry weight, we might specify a gamma distribution for dry weight.

```
barnacle <- subset(barnacle, DW > 0) # remove observations of zero weight to avoid errors
m <- glm(DW ~ F + log(RC) + F:log(RC), family = Gamma(link = log), data = barnacle)
summary(m)$coefficients
```

```
              Estimate Std. Error t value Pr(>|t|)
(Intercept)    -8.0613     0.0390 -206.79 0.000000
Flens          -0.1569     0.0572   -2.74 0.006184
log(RC)         2.8423     0.0159  179.30 0.000000
Flens:log(RC)   0.0788     0.0232    3.41 0.000674
```

```
d <- expand.grid(F = c("barca","lens"), RC = seq(2.3, 24, length = 100))
d$yhat <- predict(m, newdata = d, type = "response")
p <- p + geom_line(aes(y = yhat), color = "red", data = d)
plot(p)
```




```
# effect of a 20% increase in RC
trtools::contrast(m, tf = exp,
    a = list(F = c("barca","lens"), RC = 6),
    b = list(F = c("barca","lens"), RC = 5),
    cnames = c("barca","lens"))
```

```
      estimate lower upper
barca     1.68  1.67  1.69
lens      1.70  1.69  1.71
```

```
pairs(emmeans(m, ~RC|F, at = list(RC = c(6,5)),
  type = "response"), infer = TRUE)
```

```
F = barca:
 contrast  ratio      SE   df lower.CL upper.CL null t.ratio p.value
 RC6 / RC5  1.68 0.00485 1994     1.67     1.69    1 179.300  <.0001

F = lens:
 contrast  ratio      SE   df lower.CL upper.CL null t.ratio p.value
 RC6 / RC5  1.70 0.00524 1994     1.69     1.71    1 173.100  <.0001

Confidence level used: 0.95
Intervals are back-transformed from the log scale
```

```
Tests are performed on the log scale
```

```
# comparing the two locations at different values of RC
trtools::contrast(m, tf = exp,
    a = list(F = "lens", RC = c(10,15,20)),
    b = list(F = "barca", RC = c(10,15,20)),
    cnames = c("10mm","15mm","20mm"))
```

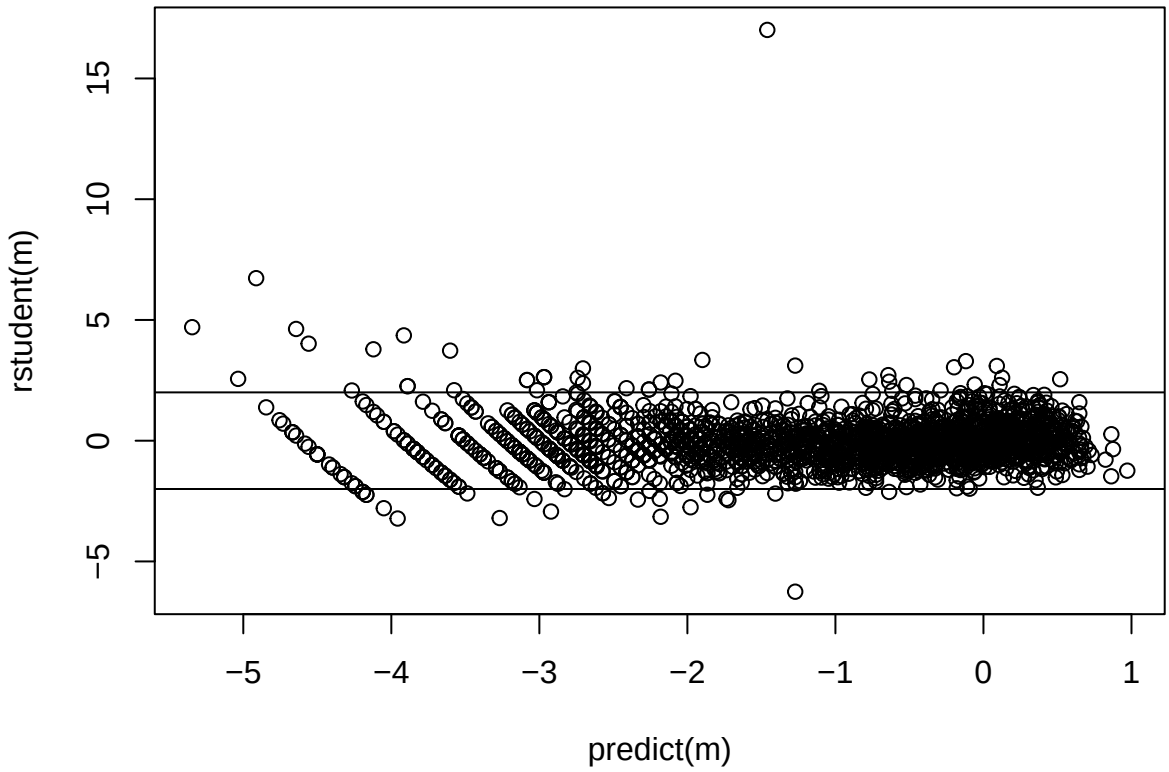
```
     estimate lower upper
10mm     1.02  1.00  1.05
15mm     1.06  1.03  1.08
20mm     1.08  1.05  1.12
```

```
pairs(emmeans(m, ~F|RC, at = list(RC = c(10,15,20)),
  type = "response"), infer = TRUE, reverse = TRUE)
```

```
RC = 10:
 contrast     ratio     SE   df lower.CL upper.CL null t.ratio p.value
 lens / barca  1.02 0.0107 1994     1.00     1.05    1   2.360  0.0184

RC = 15:
 contrast     ratio     SE   df lower.CL upper.CL null t.ratio p.value
 lens / barca  1.06 0.0125 1994     1.03     1.08    1   4.780  <.0001

RC = 20:
 contrast     ratio     SE   df lower.CL upper.CL null t.ratio p.value
 lens / barca  1.08 0.0178 1994     1.05     1.12    1   4.830  <.0001

Confidence level used: 0.95
Intervals are back-transformed from the log scale
Tests are performed on the log scale
```

Checking the residuals.

```
plot(predict(m), rstudent(m), main = "Residual Plot")
abline(-2,0)
abline(2,0)
```



Note: Eliminating a couple of observations due to having a zero dry weight is not of much consequence here since there are so many observations. But if there were fewer observations this would not be a good idea. A better approach would be to just specify the same model using `quasi`. Note that using `quasi` with `variance = "mu^2"` is effectively equivalent to using `family = gamma`.

```
m <- glm(DW ~ F + log(RC) + F:log(RC), data = barnacle,
  family = quasi(link = "log", variance = "mu^2"))
summary(m)$coefficients
```

```
             Estimate Std. Error t value Pr(>|t|)
(Intercept)   -8.0613     0.0390 -206.79 0.000000
Flens         -0.1569     0.0572   -2.74 0.006184
log(RC)        2.8423     0.0159  179.30 0.000000
Flens:log(RC)  0.0788     0.0232    3.41 0.000674
```

```
# effect of a 20% increase in RC
trtools::contrast(m, tf = exp,
    a = list(F = c("barca","lens"), RC = 6),
    b = list(F = c("barca","lens"), RC = 5),
    cnames = c("barca","lens"))
```

```
      estimate lower upper
barca     1.68  1.67  1.69
lens      1.70  1.69  1.71
```

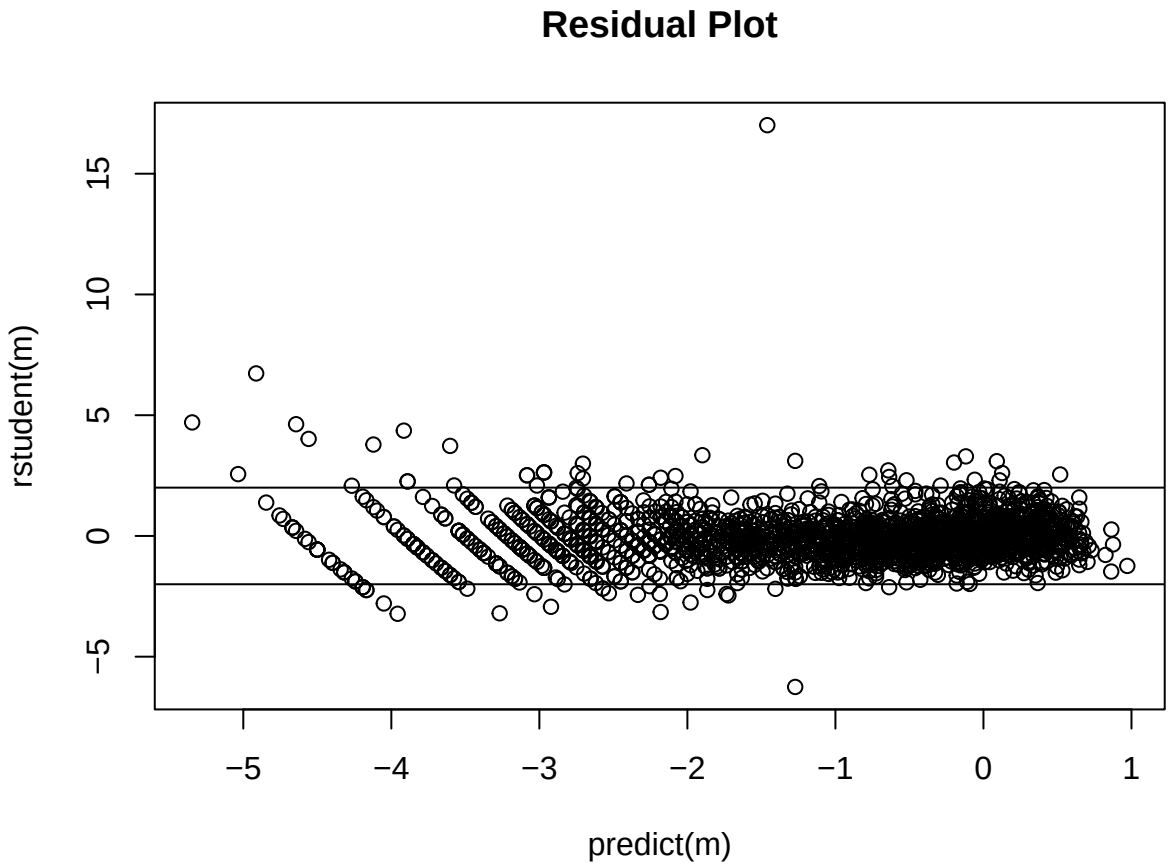
```
# comparing the two locations at different values of RC
trtools::contrast(m, tf = exp,
    a = list(F = "lens", RC = c(10,15,20)),
```

```
    b = list(F = "barca", RC = c(10,15,20)),
    cnames = c("10mm","15mm","20mm"))
```

```
     estimate lower upper
10mm     1.02  1.00  1.05
15mm     1.06  1.03  1.08
20mm     1.08  1.05  1.12
```

Checking the residuals.

```
plot(predict(m), rstudent(m), main = "Residual Plot")
abline(-2,0)
abline(2,0)
```



Inverse-gaussian GLMs are similar. There the variance increases a bit faster with the expected response. To estimate such a model use `family = inverse.gaussian`. An equivalent model is to use `quasi` with `variance = mu^3`.