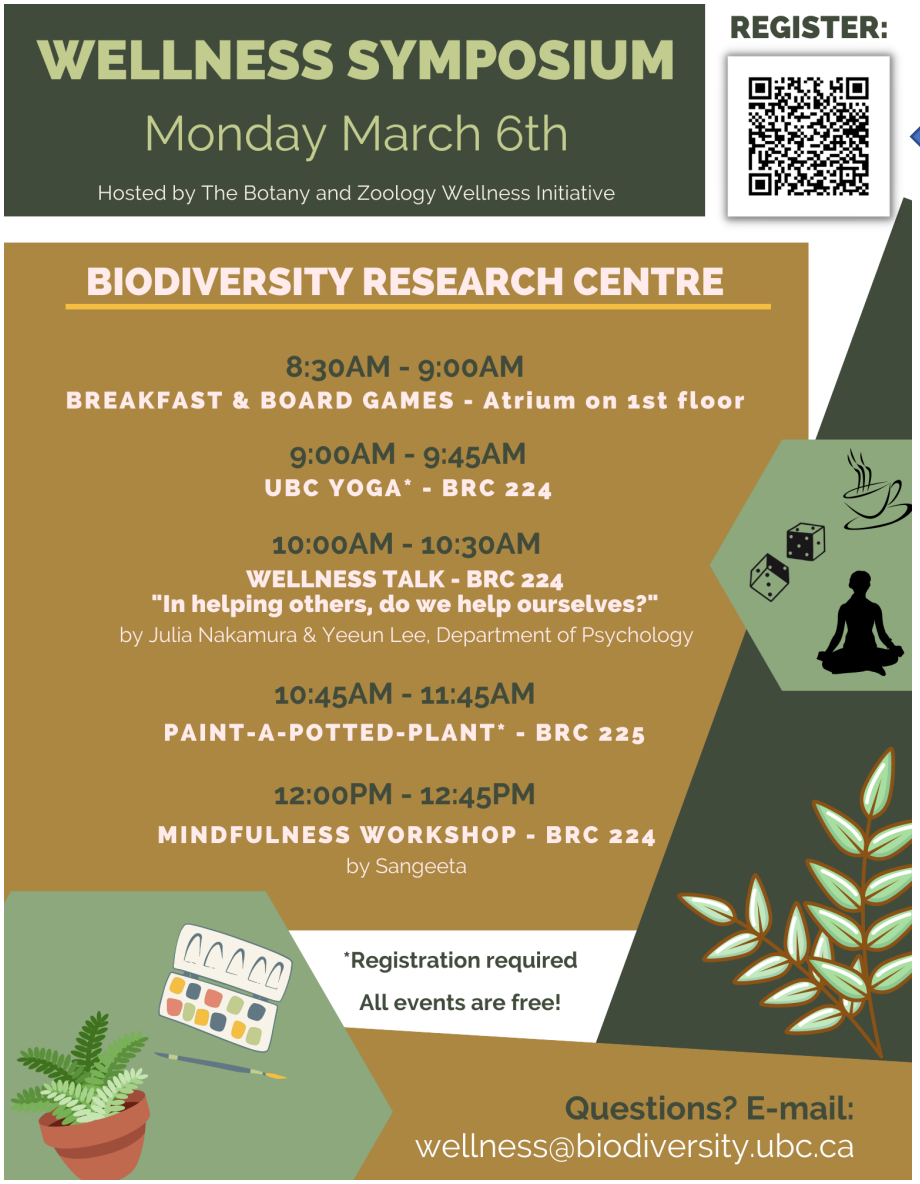
Class moved to Zoom due to snow—See Canvas Announcement

BIOL 501: Generalized linear model (GLM)

Peer-Feedback Survey

28 Feb. 2023 Dr. Beth Volpov, b.volpov@oceans.ubc.ca; UBC 2022 Term 2



Register here-free!

March 6th
8:30am- 1pm
Biodiversity Research Centre

Free and open to everyone. Ok to pop in and out during the day.

Outline for today
GLMs

- What is a generalized linear model
- Advantages and assumptions of GLMs
- Linear predictors and link functions
  - Example: fit a constant (the proportion)
- Analysis of deviance table
  - Example: fit dose-response data using logistic regression
  - Example: fit count data using a log-linear model
- Modeling overdispersion (excessive variance)
  - Example: Modeling contingency tables

Review linear model

Review: fitting a linear model in R

- Y=mx+b
- Y is response variable
- X is explanatory variable
- Errors normally distributed with **equal variance** at all values of the X variables
- Uses least squares to fit model to data and to estimate parameters
- **R code:** uses lm() for fixed effects only or lme() if mixed-effects linear model

Review: fitting a linear model in R

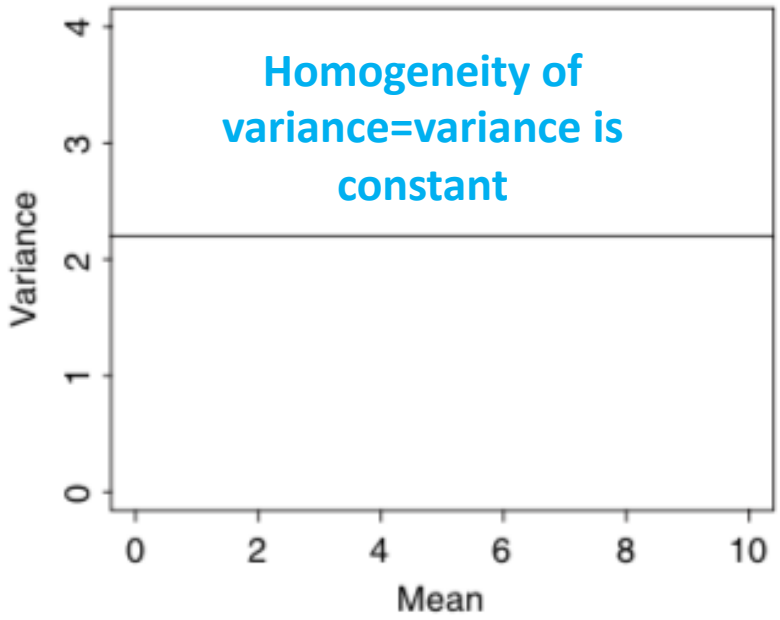
- Can x **predict** y with a linear equation?
  - Linear regression—**predictive with numeric x**
  - z <- lm(y ~ x)
  - Scatter plot with a line
- Does y differ among x categories?
  - Single factor ANOVA —**not predictive , categorical X**
  - z <- lm(y ~ x)
  - Box-plot

Predicted Y-values are modelled directly in a linear model (same units, same scale)

What is a GLM and why use it?

Why use GLMs

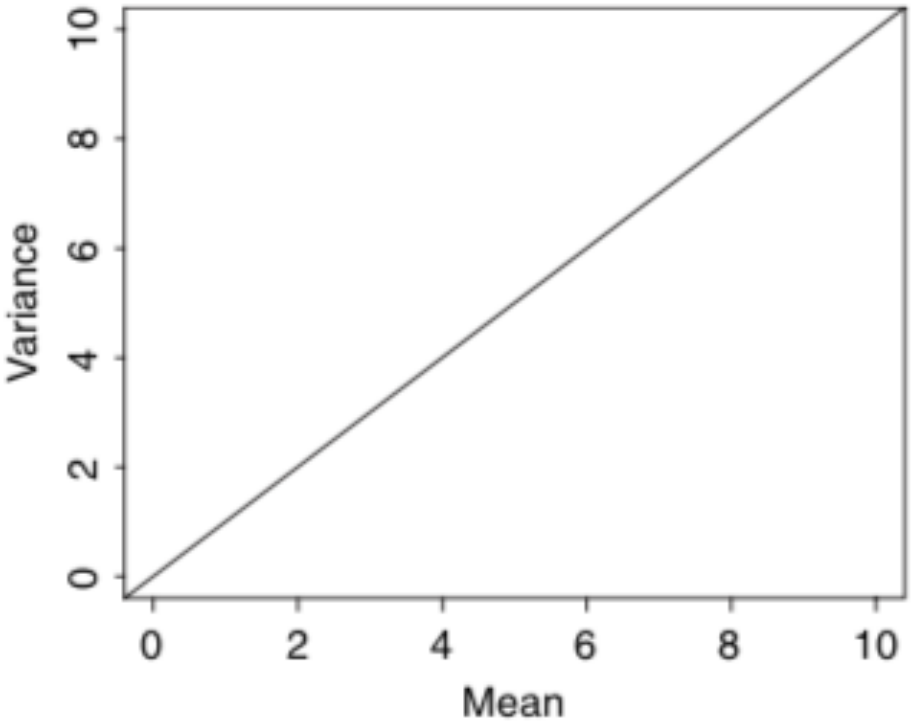
With linear modelling (lm or lme), central assumption is that variance is constant (flat line),



But in many practical applications, variance is **not** constant, so this assumption is invalid

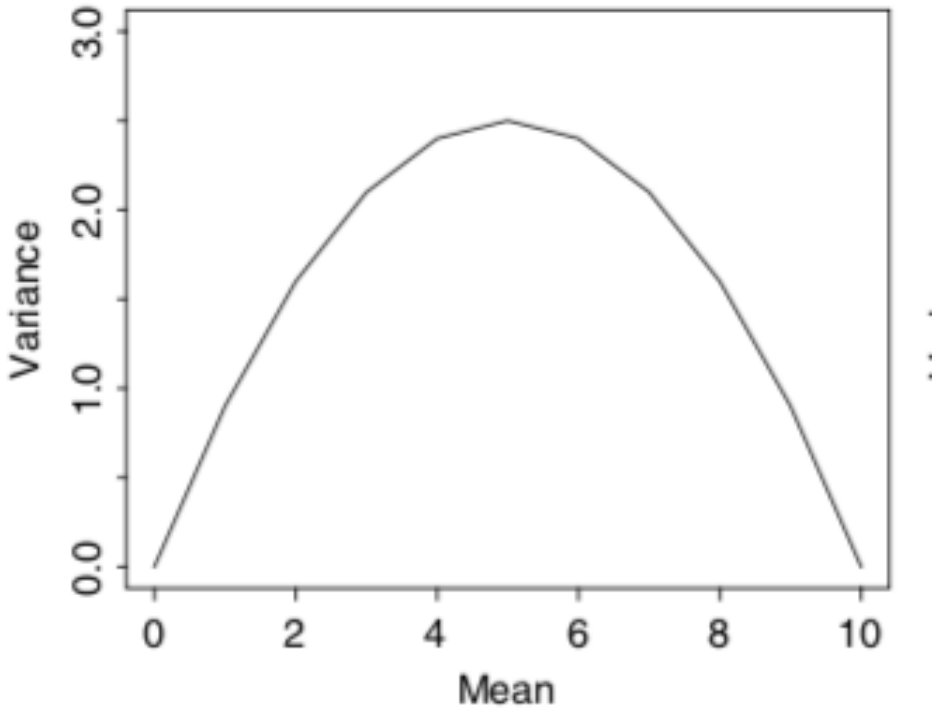
But...with **count data** (often zero-inflated)

Variance often increases
linearly with the mean



But…with **proportion data** (e.g. success vs failure),

variance can be an inverted U-shaped function of the mean (bottom left).



GLMs are an extension of regular linear modelling

- GLMs extend the linear modelling framework to variables that are not Normally distributed and don't meet homogeneity of variance
- **What does Generalized mean?**
  - A glm is a flexible generalization of ordinary linear regression
  - Still assuming a linear relationship
  - If not linear—try a GAM instead.

You have already done a GLM!

- Linear regression is also a special case of a GLM
- **Link function:** identity
- **Probability distribution:** Gaussian (Normal).
- Linear predictor and the parameter for Gaussian distribution are identical (response variable is in same units)

When is a GLM useful

- Response variable has a distribution other than the normal (Gaussian) distribution, and transformation of the data is undesirable or impossible.
- Examples:
  - Binary response data (1 or 0, dead or alive)
  - Data that are counts (number of offspring, leaves, or tattoos).
  - Analysis of contingency tables (confusion matrices).

Advantages of GLMs vs transforming response variable

- More flexible than simply transforming variables.
- Yields more familiar measures of the response variable than data transformations.
- Avoids the problems associated with transforming 0's and 1's. For example, the logit transformation of 0 or 1 can't be computed.
- **Retains the same analysis framework as linear models.**
- glm() can handle data having other probability distributions than the ones used in these examples, including exponential and gamma distributions.

Generalized linear model

- The model still includes a *linear predictor* **But** the **predicted *Y*-values are not modelled directly**
- Non-normal distributions of errors and unequal error variances are ok because specified by link function
- Uses maximum likelihood to estimate parameters
- Uses log-likelihood ratio tests to test parameters
- **R Code:** fit models using **glm()**

Predicted Y-values are NOT modelled directly (*different* units, *different* scale)

Generalized linear model

- The model still includes a *linear predictor* **But** the **predicted *Y*-values are now transformed**

- **All glms have these basic parts**
  1. Error structure (probability distributions)
  2. Linear predictor
  3. Link function

Don’t worry→The R coding for GLMS is not a huge change from lm

- The **R Code to fit a model is similar to lm(),** except that now you have to also specify an **error distribution** and **link function** must be specified using the family argument.
- The outputs are a bit different since not modelling response variable directly (have to do inverse of link fx to get back to original response), but lots of overlap with lm()

Make sure you’re comfortable with lm before diving into glm

Assumptions of GLMS

Assumptions

- Statistical independence of data points
  - Use glm if only fixed effects
  - Use glm**m** if mixed-effects (e.g. repeated measures, random effects)
  - Assumes linear relationship→If not, do a GAM first
- Correct specification of the link function for the data.
- The variances of the residuals correspond to that assumed by the link function.

Evaluating assumptions of the glm() fit

- Do the variances of the residuals correspond to those assumed by the chosen link function?

- The log link function assumes that the *Y* values are Poisson distributed at each *X.*

- A key property of the Poisson distribution is that within each treatment group the variance and mean are equal (i.e., the glm() dispersion parameter = 1). **But real data rarely show this.**

Correcting for overdispersion

- Assume the variance of the error distribution is exactly specified by poisson distribution.
- Typically, however, the error variance for count data is greater than that specified by the poisson distribution termed “overdispersion”

If the variances of the errors in the data are not in agreement with the distributions, use the following instead.

Logistic regression example with binomial

family = quasibinomial(link = "logit")

Log-libear regression example with poisson

family = quasipoisson(link = "log"))

Using GLMS to model error structures

- Up to this point, we have dealt with the statistical analysis of data with normal errors
  - But in reality many kinds of data have non-normal errors
- Examples. (not exhaustive)
  - errors that are strictly bounded (as in proportions);
  - errors that cannot lead to negative fitted values (as in counts).
- Options to “fix” this
  1. Transformation of the response variable
  2. Non-parametric methods
  3. GLM allows the specification of a variety of different error distributions

Error distributions

- Poisson errors, useful with count data;
- binomial errors, useful with data on proportions;
- gamma errors, useful with data showing a constant coefficient of variation;
- exponential errors, useful with data on time to death (survival analysis).

Linear predictors and link functions

Link Functions

- One of the difficult things to grasp about GLMs is the relationship between the values of the response variable (as measured in the data and predicted by the model in fitted values) and the linear predictor.

The thing to remember is that the link function "links" or relates the mean value of y to its linear predictor.

Link function literally "links" the linear predictor and the parameter for probability distribution

Link function relates mean value of y to it's linear predictor

- The value of **the linear predictor**, is obtained by **transforming the value of y by the link function**

- The **predicted value of y** is obtained by applying the **inverse of the link function to the linear predictor**

Because you are going through a link function, the units are not the same as your response variable. You need to do the inverse of the link fx to report the values.

Several link functions to choose from

Common probability distributions and their canonical link functions (common pairings, but not the only options).

Error	Canonical link
normal	*identity*
poisson	*log*
binomial	*logit*
Gamma	*reciprocal*

Specify error structure with family argument

- Normal
  - glm(y~x, family=identity)
  - identity link function
- Poisson errors, useful with count data
  - glm(y ~ x, family = poisson) or family=quasipoisson
  - log link function
- binomial errors, useful with data on proportions
  - glm(y ~ x, family = binomial) or family=quasibinomial
- exponential errors, useful with data on time to death (survival analysis)
  - glm(y ~ x, family = exponential)
- gamma errors, useful with data showing a constant coefficient of variation

2 most common link functions:**logistic (aka logit)**

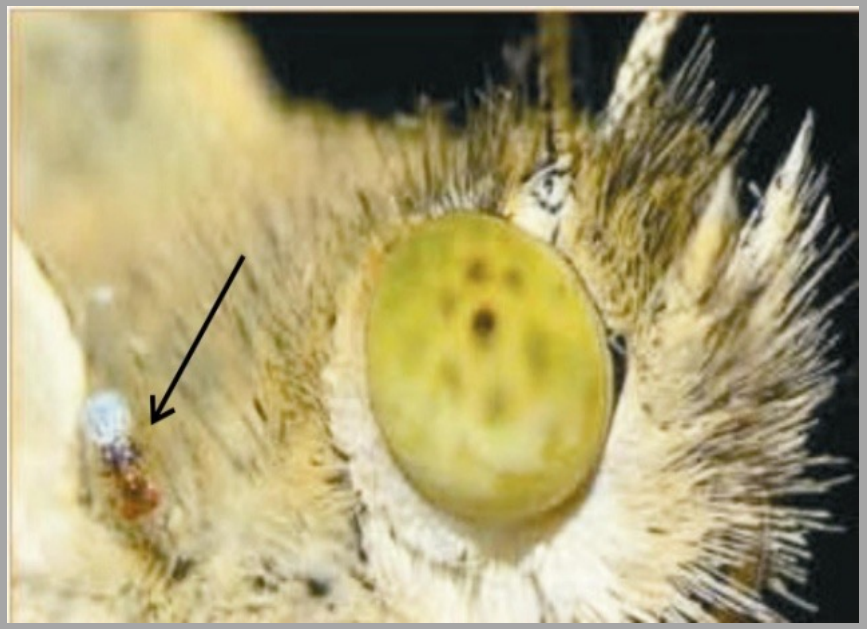
- Used to model binary data (e.g., survived vs died)
- The link function $\log \frac{\mu}{1-\mu}$ is also known as the log-odds

Common link functions:**log**

- Natural log (i.e., base *e*)
- log(*predicted y values*) = $\eta = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots$
- Greek mu Greek eta
- Usually used to model <u>count data</u> (e.g., number of mates, etc)
- **Link function:** log(predicted y values)
- **Inverse function to get predicted y values:***predicted Y values* = e^{η}

Start the log link function example with wasps

Back to the wasp example form MLE



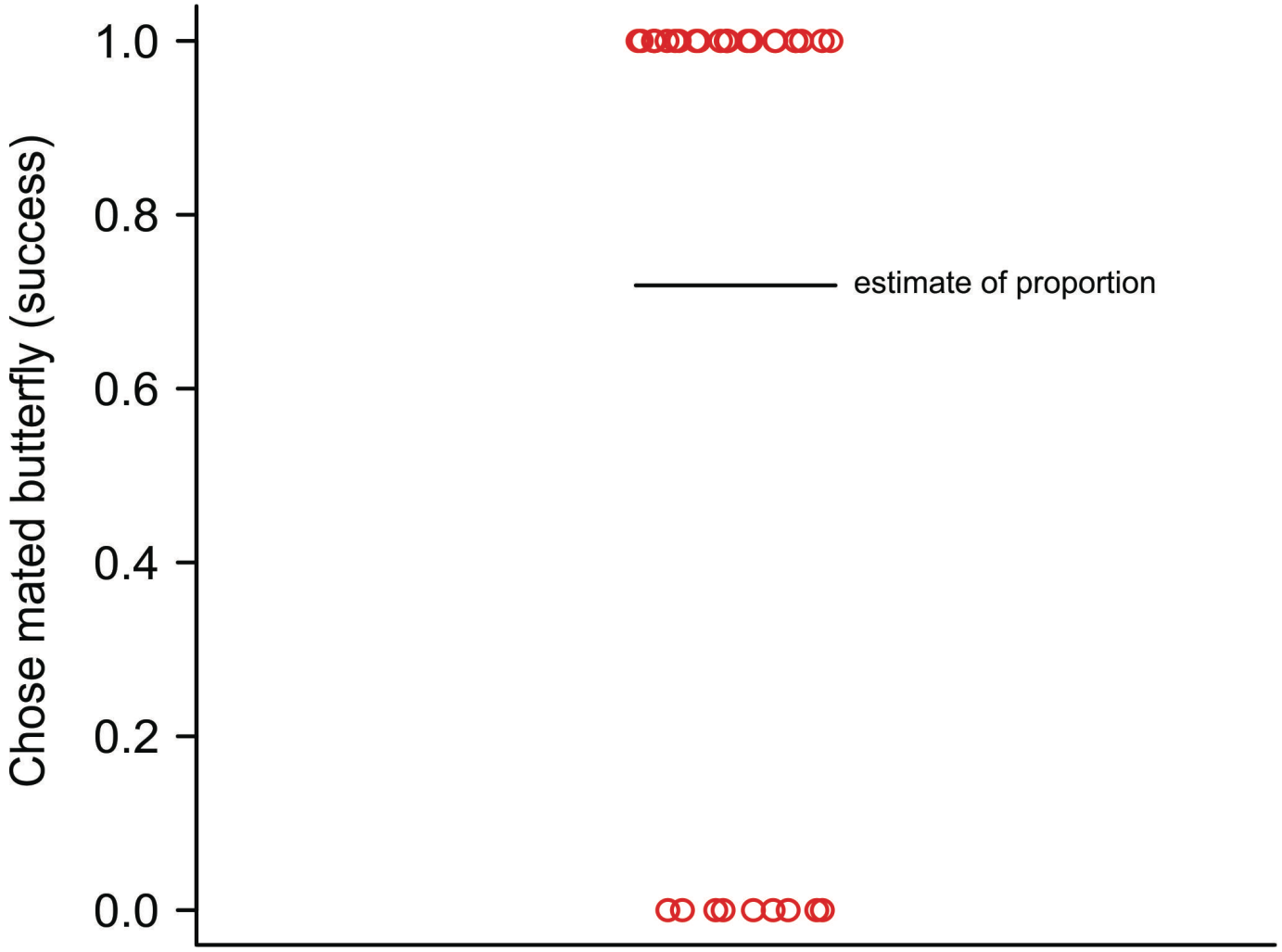
Wasp example

- This example was used previously in Likelihood lecture. My goal here is to connect what glm() does with what we did by brute force with likelihood previously.
- The wasp, *Trichogramma brassicae*, rides on female cabbage white butterflies, *Pieris brassicae*. When a butterfly lays her eggs on a cabbage, the wasp climbs down and parasitizes the freshly laid eggs.
- Fatouros et al. (2005) carried out trials to determine whether the wasps can distinguish mated female butterflies from unmated females. In each trial a single wasp was presented with two female cabbage white butterflies, one a virgin female, the other recently mated.
- Y = 23 of 32 wasps tested chose the mated female. What is the proportion p of wasps in the population choosing the mated female?

- **Number of wasps choosing the mated female fits a binomial distribution**
- Under random sampling, the number of “successes” in *n* trials has a binomial distribution, with *p* being the probability of “success” in any one trial.
- To model these data, let “success” be “wasp chose mated butterfly”
- Y=23 successes
- N=32 trials
- Goal is to estimate the probability of success “p”

Use glm() to fit a constant, and so obtain the ML estimate of *p*

- The data are binary. Each wasp has a measurement of 1 or 0 ("success" or "failure") for choice: 1 1 1 0 1 1 1 0 1 0 1 0 1 1 1 1 0 1 0 1 1 1 1 1 0 1 1 1 0 0 1 1
- z <- glm(choice ~ 1, family = binomial(link="logit"))
- Family specifies the error distribution (binomial) and the link function (logit)


Chose mated butterfly (success)
1.0
0.8
0.6
0.4
0.2
0.0
estimate of proportion

Use glm() to fit a constant, and so obtain the ML estimate of *p*

- Fits a model having only a constant. Use the link function appropriate for binary data:
- Fits a model having only a constant. Use the link function appropriate for binary data:
-
- $\log \frac{\mu}{1-\mu} = \beta$
-
- μ here refers to the population proportion (*p*) but let's stick with μ symbol here to use consistent notation for generalized linear models.
-
- Fitting will yield the estimate,$\hat{\beta}$.
-
- The estimate of proportion $\hat{\mu}$ is then obtained using the inverse function:

Use summary() for estimation

Use `summary()` for estimation

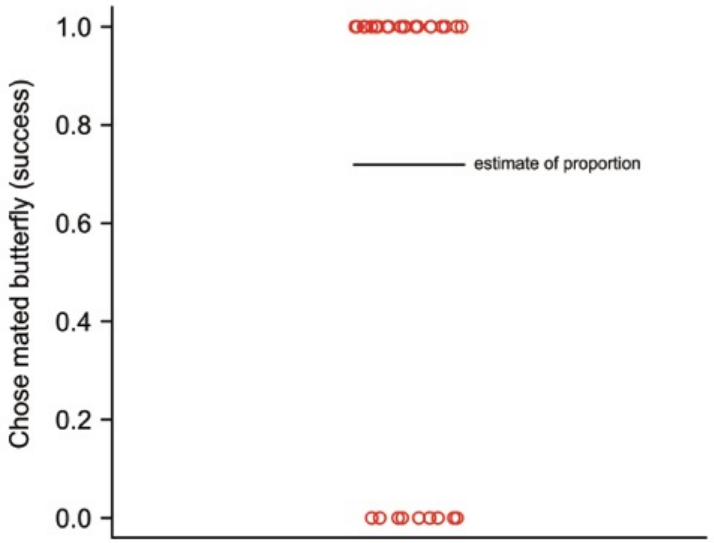
```
summary(z)
```

```
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.9383     0.3932   2.386    0.017 *
```

0.9383 is the estimate of β (the constant on the **logit** scale). Convert back to ordinary scale (plug into inverse equation) to get estimate of population proportion:

$$\hat{\mu} = \frac{e^{\hat{\beta}}}{1 + e^{\hat{\beta}}} = \frac{e^{0.9383}}{1 + e^{0.9383}} = 0.719$$

This is the ML estimate of the population proportion. This is identical to the estimate obtained last week using likelihood function.



Confidence intervals of GLM

Confidence intervals

```
summary(z)

             Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.9383     0.3932   2.386    0.017 *
```

95% confidence limits:

```
myCI <- confint(z)          # on logit scale
exp(myCI)/(1 + exp(myCI)) # inverse logit scale

    2.5 %     97.5 %
0.5501812 0.8535933
```

$0.550 \leq p \leq 0.853$ is the same result we obtained last week for likelihood based confidence intervals using likelihood function (more decimal places this week).

Avoid using `summary()` for hypothesis testing

```
summary(z)
```

```
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    0.9383     0.3932   2.386    0.017 *
```

The *z*-value (Wald statistic) and *P*-value test the null hypothesis that $\beta = 0$. This is the same as a test of the null hypothesis that the true (population) proportion $\mu = 0.5$, because

$$\frac{e^0}{1 + e^0} = 0.5$$

Agresti (2002, *Categorical data analysis*, 2nd ed., Wiley) says that for small to moderate sample size, the Wald test is less reliable than the log-likelihood ratio test.

Log-likelihood ratio test using glm

Use anova(null,model1) to compare models with LRT

- Last week we calculated the log-likelihood ratio test for these data “by hand”.
  Here we’ll use glm() to accomplish the same task.
- “Full” model (*b* estimated from data):
- z1 <- glm(y ~ 1, family = binomial(link=”logit”))
- “Reduced” model (*b* set to 0 by removing intercept from model):
- z0 <- glm(y ~ 0, family = binomial(link=”logit”))

Use anova() to test hypotheses

```
anova(z0, z1, test  = "Chi") # Analysis of deviance

 Model 1: y ~ 0     # Reduced model
 Model 2: y ~ 1     # Full model
```

Analysis of deviance table:

```
   Resid. Df Resid. Dev Df Deviance P(>|Chi|)
 1        32     44.361
 2        31     38.024  1    6.337   0.01182 *
```

The deviance is the log-likelihood ratio statistic (*G*-statistic). It has an approximate χ^2 distribution under the null hypothesis.

Residual deviance measures goodness of fit of the model to the data.

G = 6.337 is the identical result we obtained "by hand" using log likelihood ratio test last week.

Workshop

Workshop

1. Read and examine data
   - Do a quick plot of the data
2. Fit a GLM
   - What probability distribution is the Y variable?
   - What is the appropriate link function?
3. Visualize the model (plot it)
4. Extract useful information from model
5. Calculate likelihood-based 95% CI

Sections of the Workshop

- Natural selection in song sparrows
  - Binary response (dead or alive)
  - Logit link function
- Crab satellites
  - Poisson distribution
  - Log link function
- Prion resistance
  - Contingency table
  - Poisson distribution
  - Log link function

R Tips Page

R Tips Page-Fit a model **glm()**

- **Glm() is similar to lm() except, must specify**

1. Error distribution (e.g binomial=binary, poisson =count data)
2. link function specified using the family argument.
   - **Logit==logistic**

- Assume that the variance of the error distribution is exactly specified by the distribution
- **Fix:** If it's not, use the "quasi-" versions of each distribution
  - Output includes estimate of the dispersion parameter (a value greater than one indicates overdispersion, whereas a value less than 1 indicates underdispersion).

Fit a glm() withpoisson and quasipoisson error distributions on **count data**

- model1<- glm(response ~ explanatory, family = poisson(link="log"), data = mydata)

- **Fix: Use"quasi-"distribution instead if overdispersion**
- model1<- glm(response ~ explanatory, family = **quasi**poisson(link = "log"), data = mydata)

Fit a glm() with binomial and quasibinomial error distributions

- model1<- glm(Y ~ X, family = binomial(link="logit"), data = mydata)
- Assume that the variance of the error distribution is exactly specified by the binomial distribution
- If error variance overdispersed→**Fix by using 'quasi-" instead:** Output will include estimate of dispersion parameter (if > 1 indicates overdispersion; if < 1 indicates underdispersion).
- model1<- glm(Y ~ X, family = **quasi**binomial(link = "logit"), data = mydata)

Type"help(family)" to see other error distributions and link functions that can be modeled using glm().

R Tips Page: Useful glm commands on model1

- summary(model1) # parameter estimates, overall model fit
- coef(model1) # model coefficients
- resid(model1) # deviance residuals
- predict(model1) # predicted values on the transformed scale
- predict(model1, se.fit = TRUE). # Includes SE's of predicted values
- fitted(model1) # predicted values on original scale
- anova(model1, test = "Chisq") # Analysisa of deviance - sequential
- anova(model1, model2, test = "Chisq") # compare fits of 2 models, "reduced" vs "full"
- anova(model1, test = "F") # Use F test for gaussian, quasibinomial or quasipoisson