Regression Models in Systems Biology with R

Part III: Generalized Linear Model

Uwe Menzel 2014

www.matstat.org

Outline

1. **Simple Linear Regression**
    1. The statistics behind the output of "`lm`"
2. **General Linear Model**
    1. Continuous and categorical variables mixed, "`lm`"
    2. Interaction
3. **Generalized Linear Model**
    1. Logistic Regression - "`glm`"
    2. Multinomial Regression - "`multinom`"

3. Generalized Linear Models

- Generalization of (general) linear models
- Response variable is connected to the linear part via a link function
- Different error distributions of the response variable possible:
  - normal, binomial, Poisson, ...

$$f(y) = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \ldots + \beta_k \cdot x_k$$ (no error term!)

$$f(y) = \begin{cases} y & \textit{ordinary linear regression} \\ \ln(y) & \textit{log} \\ y^{-2} & \textit{inverse} \\ \sqrt{y} & \textit{inverse square} \\ \ln\left(\frac{y}{1-y}\right) & \textit{logit} \\ \ln\left(-\ln\left(1-y\right)\right) & \textit{log-log} \end{cases}$$

Assumptions:

- independence of errros
- absence of multicollinearity
- lack of strongly influential outliers

Logistic Regression

Fitting a regression curve $y = f(x)$ when the response y is binomial

Example: Th. Tarpey [http://www.wright.edu/~thaddeus.tarpey/]

Predictor: Dosage of carbon disulphide (CS_2, insecticide) - continuous
Response: binomial (dichotomous): killed / alive or 1/0
Response in $(0, 1)$ → ordinary linear regression is not possible:

$$\cancel{y = \beta_0 + \beta_1 \cdot x_1 + \beta_2 \cdot x_2 + \ldots + \beta_k \cdot x_k + \varepsilon}$$

Logistic Regression

Data can be given for each sample:

```
# Sample      Dose     Killed  # Killed (=1) or not (=2) for each beetle
# 1           49.1          0
# 2           49.1          0
# 3           49.1          1
# 4           49.1          0
# ...          ...        ...
# ...          ...        ...
# 478         76.5          1
# 479         76.5          1
# 480         76.5          1
# 481         76.5          1
```

... or (somewhat pre-processed) as proportions:

```
#    Dose     Exposed  Killed  # Number Exposed & Killed for each Dose
# 1  49.1          59       6
# 2  53.0          60      13
# 3  56.9          62      18
# 4  60.8          56      28
# 5  64.8          63      52
# 6  68.7          59      53
# 7  72.6          62      61
# 8  76.5          60      60
```

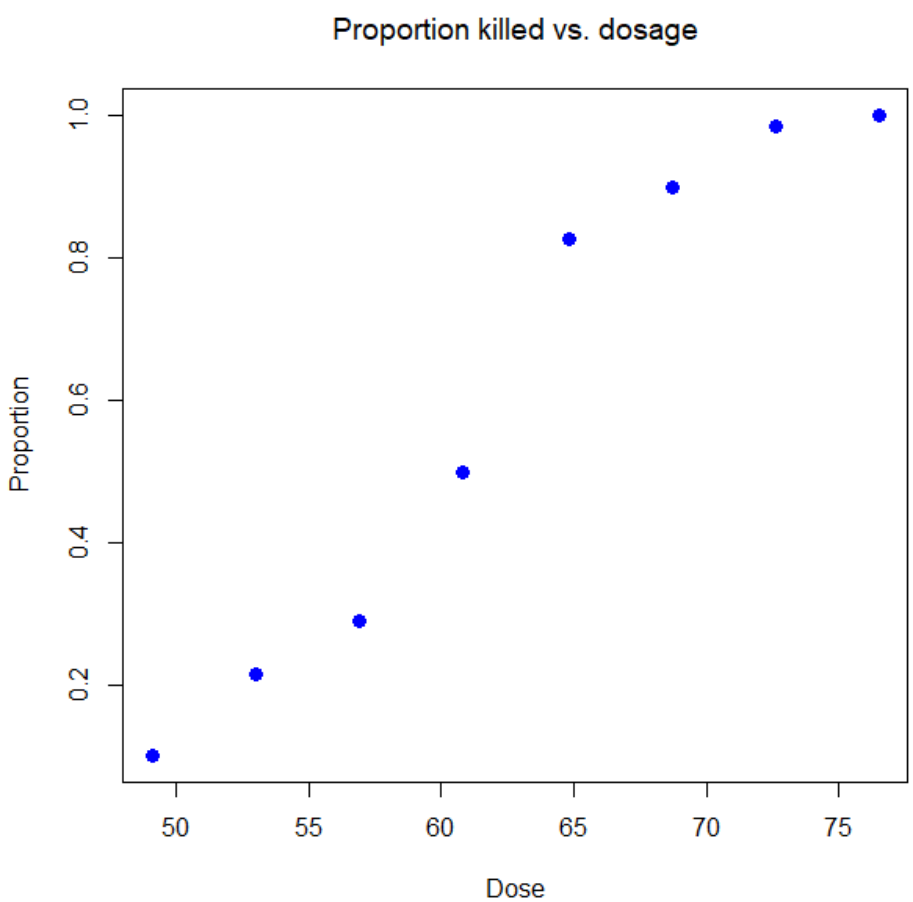
Logistic Regression

Plot proportion vs. dosage:

```
Proportion = beetles$Killed / beetles$Exposed

beetles = cbind(beetles, Proportion)
beetles

#   Dose Exposed  Killed     Proportion
# 1 49.1     59        6       0.1016949
# 2 53.0     60       13       0.2166667
# 3 56.9     62       18       0.2903226
# 4 60.8     56       28       0.5000000
# 5 64.8     63       52       0.8253968
# 6 68.7     59       53       0.8983051
# 7 72.6     62       61       0.9838710
# 8 76.5     60       60       1.0000000

plot(Proportion ~ Dose, data = beetles, main = …
```

Logistic Regression

```
plot(Proportion ~ Dose, data = beetles, main = "Proportion killed vs. dosage")
```



S-shaped between 0 and 1

$$y = \frac{e^t}{1+e^t} = \frac{1}{1+e^{-t}}$$

- logistic function
- t can be any real number
- ... but y is confined to the interval $(0, 1)$

is equivalent to:

$$\ln\left(\frac{y}{1-y}\right) = t$$

Logistic Regression

The response (proportions) is between 0 and 1, and the curve looks S-shaped. Hence, to model the relationship between dosage (x) and proportion killed (p), a logistic function could work, for instance:

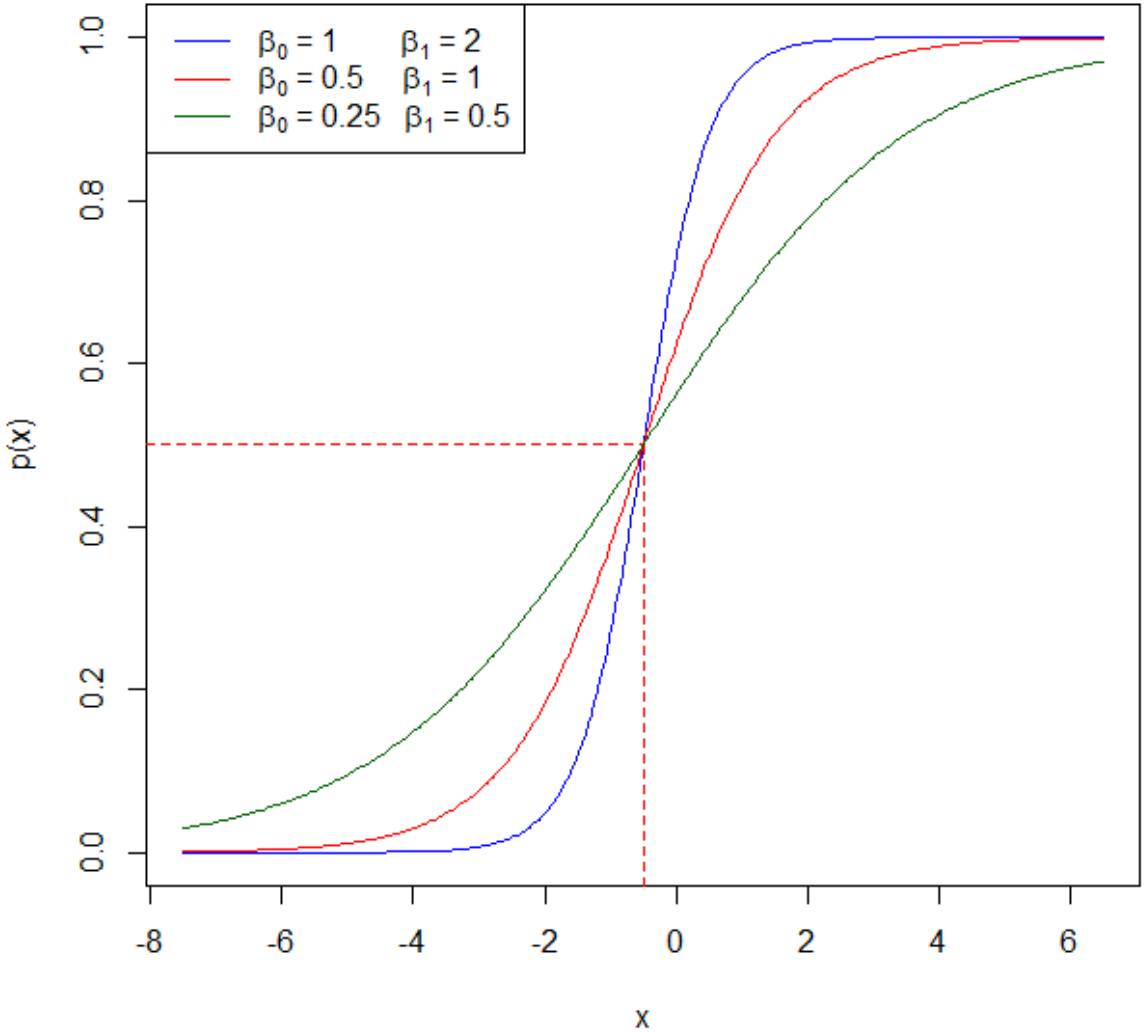
$$p(x) = \frac{e^{\beta_0 + \beta_1 x}}{1 + e^{\beta_0 + \beta_1 x}}$$

By using two parameters (β_0, β_1) as argument of the exponential function, we are able to independently adjust the center $x_{0.5}$ of the curve (where $y = 0.5$) and the slope $p'(x)$ at this center, since we have:

$$x_{0.5} = -\frac{\beta_0}{\beta_1} \qquad p'(x)|_{x=x_{0.5}} = -\frac{\beta_1}{4}$$

The next pages show how $p(x)$ changes with β_0 and β_1.

Logistic Function

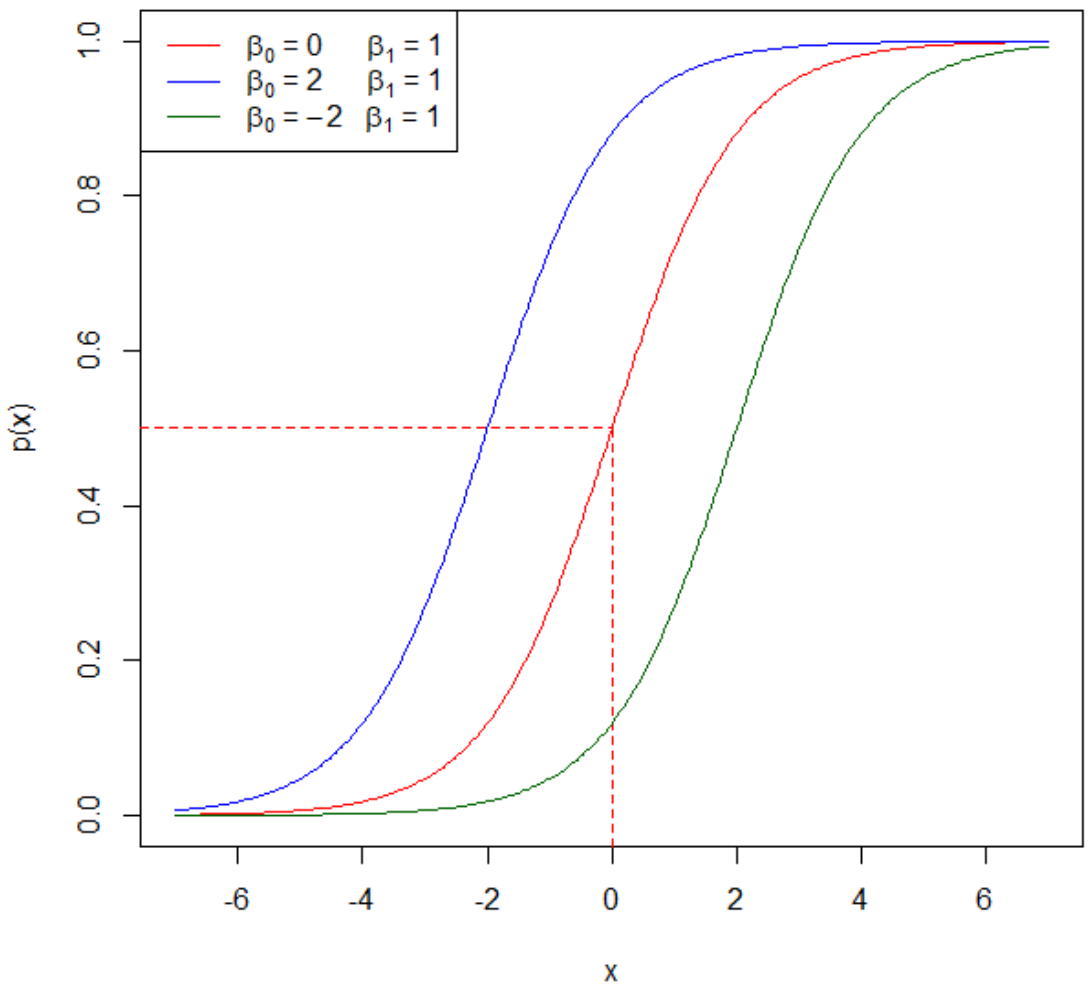


$$x_{0.5} = -\frac{\beta_0}{\beta_1}$$

$$p'(x)|_{x=x_{0.5}} = -\frac{\beta_1}{4}$$

Changing β_1 but keeping the ratio β_0/β_1 constant does not change the point on the x-axis where $p(x) = 1/2$, but it changes the slope in this point.

Logistic Function



$$x_{0.5} = -\frac{\beta_0}{\beta_1}$$

$$p'(x)|_{x=x_{0.5}} = -\frac{\beta_1}{4}$$

Keeping β_1 constant and changing β_0 shifts the point where $p(x) = 1/2$ but does not changes the slope in this point.

Logistic Regression

The formula $p(x) = \dfrac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}}$ is equivalent to:

$$\ln\left[\frac{p(x)}{1-p(x)}\right] = \beta_0 + \beta_1 x$$

... where the left side is called logit.

The right-hand term can be extended when the model includes multiple predictors:

$$\ln\left[\frac{p(\mathbf{x})}{1-p(\mathbf{x})}\right] = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots$$

Logistic Regression
- Terms -

$$\ln\left[\frac{p(x)}{1-p(x)}\right] = \beta_0 + \beta_1 x$$

- If p is interpreted as a probability, then $p/(1-p)$ is called odds:
  - $odds = p(succes)\ /\ p(failure)$
- The natural logarithm of the odds is called "log odds" or "logit".
- The logit is the link function for logistic regression.
- The odds can take arbitrary positiv real values,
  - e.g. $p = 0.8 \rightarrow \text{odds} = 4$
- The log-odds can take arbitrary real values,
  - e.g. $p = 0.2 \rightarrow \text{odds} = 1/4 \rightarrow ln(1/4) = -1.386$
  - e.g. $p = 0.5 \rightarrow \text{odds} = 1 \rightarrow ln(1) = 0$

Probability vs. Odds

Throw a die:

Random variable X: number on the die

$$P\,(\text{throw a } 1) \quad = \quad P(X=1) = 1/6$$

$$Odds\,(\text{throw a } 1) \quad = \quad \frac{P(X=1)}{1-P(X=1)} = \frac{1/6}{5/6} = \frac{1}{5}$$

Logistic Regression: Parameter Estimation*

$$\ln\left[\frac{p(x)}{1-p(x)}\right] = \beta_0 + \beta_1 x$$

How to find $\boldsymbol{\beta_0}$ and $\boldsymbol{\beta_1}$?

- determine β_0 and β_1 in such a way that the fitted $p(x)$ matches as good as possible the observed points $\{x_i, p(x_i)\}$
- least-squares regression is **not** possible:
  - neither the normality nor the equal variance assumption are met for 0/1-values !
- Maximum-Likelihood (**ML**) estimation is more promising: choose those parameters (β_0, β_1) that make the observation most likely

Logistic Regression: Parameter Estimation*

The response variable in the above experiment is binomial for a given dosage (two outcomes: killed / alive) →

- Taking n independent samples, each with success probability p, gives the following probability that k of them succeed:
- (random variable X = number of successes, success = beetle killed [Sorry!]):

$$P(X = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

probability mass function for the binomial distribution

The first observation of the beetle experiment was:

```
#    Dose    Exposed    Killed
# 1  49.1        59         6
# 2  ...
```

For the dosage 49.1, we had 6 successes in 59 trials. This is the experimental outcome. in ML, we search the p which makes that outcome most likely.

ML seeks the parameter p that makes the observed outcome most likely. Hence, if we had this row only, we would try to find the p that maximizes $L(p)$ in the following expression:

$$L(p) = \binom{59}{6} p^6 (1-p)^{53} \Longrightarrow \text{maximize by adjusting } p$$

Logistic Regression: Parameter Estimation*

However, because we have multiple observations (at different dosages x), we have to maximize the joint likelihood covering all observed outcomes:

$$L(p) = P\left(X_1 = k_1, X_2 = k_2, \ldots \mid p(x)\right)$$

Note that there is no single probability p for all X_i because p depends on the dosage. We presumed that the measurements are independent, so that the joint likelihood simplifies to:

$$L(p) = P\left(X_1 = k_1 \mid p_1\right) \cdot P\left(X_2 = k_2 \mid p_2\right) \cdot \ldots$$

or in more detail:

$$L(p) = \binom{n_1}{k_1} p_1^{k_1} \left(1 - p_1\right)^{n_1 - k_1} \cdot \binom{n_2}{k_2} p_2^{k_2} \left(1 - p_2\right)^{n_2 - k_2} \cdot \ldots$$

We want to find p as a function of the dosage, i.e. we propose there exists a function $p(x)$, so that we can write $p_1 = p(x_1), p_2 = p(x_2)$, etc. , which allows the following replacement:

Logistic Regression: Parameter Estimation*

$$L = \binom{n_1}{k_1} p_1^{k_1} (1-p_1)^{n_1-k_1} \cdot \binom{n_2}{k_2} p_2^{k_2} (1-p_2)^{n_2-k_2} \cdot \ldots$$

consider p as a function of x

$$L = \binom{n_1}{k_1} p(x_1)^{k_1} (1-p(x_1))^{n_1-k_1} \times \binom{n_2}{k_2} p(x_2)^{k_2} (1-p(x_2))^{n_2-k_2} \times \ldots$$

As shown above, it might be convenient to think of $p(x)$ as a logistic function:

$$p(x_1) = \frac{e^{\beta_0+\beta_1 x_1}}{1+e^{\beta_0+\beta_1 x_1}}$$

$$p(x_2) = \frac{e^{\beta_0+\beta_1 x_2}}{1+e^{\beta_0+\beta_1 x_2}}$$

$$p(x_3) = \ldots$$

L is a function of the $p(x_1)$, $p(x_2)$, etc. which in turn are functions of β_0 and β_1, so that L is a function of β_0 and β_1. In ML, we have to choose β_0 and β_1 in such a way that $L(\beta_0, \beta_1)$ becomes maximal.

Logistic Regression with "glm"

Unfortunately, there is no closed solution for the ML expression, so that numerical solutions are needed (Newton-Raphson iteration or similar). The good news is that we don't have to care about all that, because we use R:

a) Using the "**glm**" function when the results are stored as proportions:

```
#    Dose Exposed Killed  Proportion
# 1 49.1      59      6   0.1016949
# 2 53.0      60     13   0.2166667
# 3 56.9      62     18   0.2903226
# 4 60.8      56     28   0.5000000
# 5 64.8      63     52   0.8253968
# 6 68.7      59     53   0.8983051
# 7 72.6      62     61   0.9838710
# 8 76.5      60     60   1.0000000

survived = beetles$Exposed - beetles$Killed
killed = beetles$Killed
exposure = beetles$Dose

logfit = glm(cbind(killed, survived) ~ Dose, family = binomial) # logit link!
summary(logfit)
```

Logistic Regression with "glm"

b) Using the "**glm**" function when the results are stored for each sample (using 0/1 coding):

```
# Sample Dose Killed
# 1      49.1       0
# 2      49.1       0
# 3      49.1       1
# 4      49.1       0
# ...     ...     ...
# ...     ...     ...
# 478    76.5       1
# 479    76.5       1
# 480    76.5       1
# 481    76.5       1
```

```
logfit = glm(killed ~ Dose, data=beetles.fr, family = binomial) # logit
summary(logfit)
```

Both methods a) and b) result in the same output:

Logistic Regression with "glm"

$$p(x) = \frac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}}$$

```
summary(logfit)

# .... clipped ....
#
# Coefficients:
#              Estimate Std. Error z value Pr(>|z|)
# (Intercept) -14.82300    1.28959  -11.49   <2e-16 ***
# Dose          0.24942    0.02139   11.66   <2e-16 ***
#
# (Dispersion parameter for binomial family taken to be 1)
#
# Null deviance: 284.2024 on 7 degrees of freedom
# Residual deviance: 7.3849 on 6 degrees of freedom
# AIC: 37.583
#
# Number of Fisher Scoring iterations: 4
```

β_0 β_1

Wald-Test

$$\frac{\hat{\beta}_1 - \beta_1}{se\left(\hat{\beta}_1\right)} \sim N(0,1)$$

- Fisher Scoring iteration is similar to Newton-Raphson algorithm
- Wald Test can be used because the ML-Estimator is normally distributed if the sample size is big enough

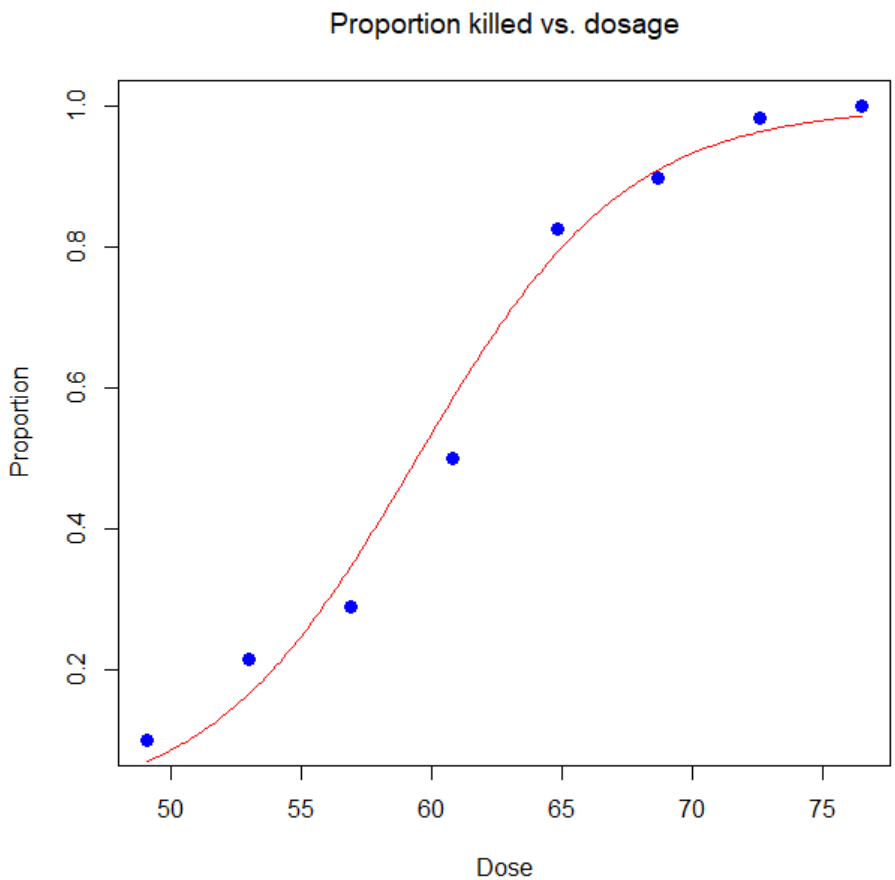
Logistic Regression with "glm"

```
beta.zero = coef(logfit)[1] # -14.823
beta.one  = coef(logfit)[2] # 0.2494156
```

That means the fitted function $p(x)$ is: $$\hat{p}(x) = \frac{e^{-14.82+0.25x}}{1+e^{-14.82+0.25x}}$$

```
plot(Proportion ~ Dose, data=beetles) # observed

range(beetles$Dose) # 49.1 76.5
x.fit = seq(49.1, 76.5, len = 201)

# the formula above:
y.fit = exp(beta.zero + beta.one*xt)/(1 + exp(beta.zero + beta.one*xt))

points(x.fit, y.fit, col="red", type="l", lwd=1.5) # fitted
```

Logistic Regression with "`glm`"



exp. values (proportions) and fitted curve

$$\hat{p}(x) = \frac{e^{-14.82+0.25x}}{1+e^{-14.82+0.25x}}$$

plotting the (0,1) values for each sample does not look pretty

Logistic Regression*

- Interpretation of the Regression coefficients -

$$\ln\left[\frac{p(x)}{1-p(x)}\right] = \beta_0 + \beta_1 x \quad \Rightarrow \quad \frac{p(x)}{1-p(x)} = e^{\beta_0+\beta_1 x}$$

$$odds(x) = e^{\beta_0+\beta_1 x}$$

If the predictor x is incremented by 1 unit, i.e. $x \rightarrow x+1$

$$odds(x+1) = e^{\beta_0+\beta_1\cdot(x+1)} = e^{\beta_1}\cdot e^{\beta_0+\beta_1 x} = \underline{e^{\beta_1}}\cdot odds(x)$$

Every unit increase in x increases the odds by $exp(\beta_1)$.

$$\frac{odds(x+1)}{odds(x)} = e^{\beta_1}$$ odds-ratio: $OR = e^{\beta_1}$ for continuous predictors

$$\frac{odds(level1)}{odds(baselevel)} = e^{\beta_1}$$ odds-ratio: $OR = e^{\beta_1}$ for categorical predictors

Logistic Regression*
- Model search-

- Often, multiple potential predictors are available
- **model search**: similar to multiple linear regression
- R: `drop1, add1, step, update, anova`
- model search: http://data.princeton.edu/r/glms.html
- Validation of models (null and residual deviance) with chi-square test, e.g.:
  - `Residual deviance: 7.3849 on 6 degrees of freedom`
  - `pchisq(7.3849, df=6, lower.tail = F) # 0.286713`
  - model fits the data well (model is rejected if $p < 0.05$)

Multinomial Regression

- **Predictors**: continuous / categorical
- **Response:** categorical
- with more than 2 possible outcomes
  - phenotype A / phenotype B / phenotype C
  - splice variant A / splice variant B / splice variant C
  - ...

6 possible outcomes

Multinomial Regression

K outcomes $\rightarrow K-1$ independent binary logistic regressions:
(see http://en.wikipedia.org/wiki/Multinomial_logistic_regression)

$$\left.\begin{aligned}
\ln\frac{P(Y_i=1)}{P(Y_i=K)} &= \boldsymbol{\beta_1}\cdot\mathbf{X_i}\\
\ln\frac{P(Y_i=2)}{P(Y_i=K)} &= \boldsymbol{\beta_2}\cdot\mathbf{X_i}\\
\cdots &= \cdots\\
\ln\frac{P(Y_i=K-1)}{P(Y_i=K)} &= \boldsymbol{\beta_{K-1}}\cdot\mathbf{X_i}
\end{aligned}\right\}$$

$$\begin{aligned}
P(Y_i=1) &= P(Y_i=K)e^{\boldsymbol{\beta_1}\cdot\mathbf{X_i}}\\
P(Y_i=2) &= P(Y_i=K)e^{\boldsymbol{\beta_2}\cdot\mathbf{X_i}}\\
\cdots &= \cdots\\
P(Y_i=K-1) &= P(Y_i=K)e^{\boldsymbol{\beta_{K-1}}\cdot\mathbf{X_i}}
\end{aligned}$$

once $P(Y_i=K)$ is known, the other $P(\ldots)$ can be calculated recursively

We have $\sum_{all\ k} P(Y_i=k)=1$, and therefore:

$$\begin{aligned}
P(Y_i=K) &= 1-\sum_{k=1}^{K-1} P(Y_i=k)\\
&= 1-\sum_{k=1}^{K-1} P(Y_i=K)\cdot e^{\boldsymbol{\beta_k}\cdot\mathbf{X_i}}
\end{aligned}$$

$$P(Y_i=K)=\frac{1}{1+\sum_{k=1}^{K-1} e^{\boldsymbol{\beta_k}\cdot\mathbf{X_i}}}$$

Multinomial Regression

```
library(nnet) # multinom (neural networks)
str(data) # Steffi, Marcel (FLI)

# 'data.frame': 181 obs. of 7 variables:
# $ Iso1 : num 6.76 6.4 5.3 4.52 4.37 ...
# $ Iso2 : num 8.55 8.4 4.98 5 8.01 ...
# $ Iso3 : num 75.2 74.6 80.1 82.1 78.5 ...
# $ Iso4 : num 9.51 10.55 9.6 8.39 9.14 ...
# $ Age : int 58 72 65 68 78 62 62 63 45 49 ...
# $ Gender: Factor w/ 2 levels "female","male": 2 1 1 2 1 2 1 2 1 1 ...
# $ group : Factor w/ 3 levels "Control","Sepsis",..: 1 1 1 1 1 1  ...

levels (group) # "Control", "Sepsis", "SIRS" : 3 possible outcomes
```

Additive model:

```
m1 = multinom(group ~ Iso1 + Iso2 + Iso3 + Iso4 + Age + Gender, data = data)
summary(m1)

regression.coef = coef(m1)                    # regression coefficients
regression.CI = confint(m1, level = 0.95)     # confidence interval
```

Multinomial Regression

A (much) smaller model:

```
m.age = multinom(group ~ Age, data = data) # just age as predictor?
summary(m.age)

anova(m1, m.age, test="Chisq") # compare models using anova

#                Model Resid.df Resid.Dev   Test  Df   LR stat.      Pr(Chi)
# 1                Age      358  251.4668         NA        NA           NA
# 2 Iso1 ... + Gender      350  165.4214 1 vs 2    8  86.0454  2.997602e-15
```

- The bigger model is significantly better: $p = 3e^{-15} \rightarrow$ age alone does not explain the vulnerability to SIRS or sepsis.
- Models can also be compared using the Akaike Information Criterion (AIC):

```
extractAIC(m1)     # 189.4214
extractAIC(m.age)  # 392.4355
```

AIC tells "how much information is lost" if the real data is replaced by the model.

$$AIC = 2k - 2\ln(L)$$

k = # parameters, L: maximized likelihood

Multinomial Regression
- Extractor functions -

```
methods(class="multinom")

add1.multinom*        anova.multinom*          coef.multinom*
confint.multinom*     drop1.multinom*          extractAIC.multinom*
logLik.multinom*      model.frame.multinom*    predict.multinom*
print.multinom*       summary.multinom*        vcov.multinom*
```

- functions for model search and comparison
- confidence intervals
- prediction of outcome
- graphs

GLM and edgeR

4288–4297 Nucleic Acids Research, 2012, Vol. 40, No. 10 Published online 28 January 2012
doi:10.1093/nar/gks042

Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation

Davis J. McCarthy[1], Yunshun Chen[1,2] and Gordon K. Smyth[1,3,*]

see DEG.edgeR.GLM.R

- The package ... implements statistical methods based on generalized linear models, suitable for multifactor experiments
- “classic edgeR” and “glm edgeR” (“`glmFit`”)
- "classic" edgeR ignores (averages over) all factors except for the one just examined