# R-Adelaide 2016

## Day 3:
## Linear and generalised linear models in R

Dr Steven Delean - Benham Rm 103
steven.delean@adelaide.edu.au

School of Biological Sciences - The University of Adelaide

# Linear model design matrix

- Two R functions, `formula` and `model.matrix` are required to produce *design matrices* for a variety of models

$$Y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \quad i = 1, \ldots, N$$

- Focus on the *observational unit* or *experimental unit* to $N$ different entities from which we obtain a measurement

- For categorical variables often called *dummy or indicator variables*

    - Simply indicate if experimental unit had a certain characteristic or not

# Linear model design matrix

Using linear model algebra to represent model for design matrix $\mathbf{X}$:

$$\mathbf{Y} = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_N \end{pmatrix}, \mathbf{X} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \\ 1 & x_N \end{pmatrix}, \beta = \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} \text{ and } \varepsilon = \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_N \end{pmatrix}$$

as

$$\begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_N \end{pmatrix} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \\ 1 & x_N \end{pmatrix} \begin{pmatrix} \beta_0 \\ \beta_1 \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \varepsilon_2 \\ \vdots \\ \varepsilon_N \end{pmatrix}$$

or simply

$$\mathbf{Y} = \mathbf{X}\beta + \varepsilon$$

# Linear model design matrix

- Define a design matrix $\rightarrow$ find the least squares estimates by *fitting the model*
- In R, we directly provide a *formula* to the model fitting function (e.g., `lm`)
- The formula statement in central to model fitting in R generally
- Generates a `model.matrix`, which is used internally by `lm` function

# Specifying the design

- Design matrix is critical step in linear modeling
  - Encodes coefficients to be fit in model, and
  - Inter-relationship between samples
- Encodes assumptions about the relationships between the response and explanatory variables
- First component usually an *intercept* column (column of 1's)
- To compare differences between groups, second column specifies which samples are in second group

# The basic formula - comparing groups

- Encode experimental design with:
  - Formula with the tilde symbol '∼'
    - So model observations using variables to right of tilde
  - Then specify name of a variable that classifies samples into groups
- Two groups, control and high fat diet, with two samples each
- Tell R that these values should not be interpreted numerically, but as different levels of a **factor**

```r
# Group is a factor
group <- factor( c(1,1,2,2) )
model.matrix(~ group)
# Do not need to use formula function as '~' implies this
model.matrix(formula(~ group))
```

```
  (Intercept) group2
1           1      0
2           1      0
3           1      1
4           1      1
attr(,"assign")
[1] 0 1
attr(,"contrasts")
attr(,"contrasts")$group
[1] "contr.treatment"
```

# The basic formula - comparing groups

- What happens if we don't tell R that 'group' should be interpreted as a factor?
  - Provided numeric variable so Formula with tilde symbol '~'
  - Want second column to have only 0 and 1, indicating group membership
- Note: names of factor levels are irrelevant to `model.matrix` and `lm`
  - All that matters is order
- Tell R that these values should not be interpreted numerically, but as different levels of a **factor**

```r
# Group is not a factor
group <- c(1,1,2,2)
model.matrix(~ group)
# Using different names produces same model.matrix
group <- factor(c("control","control","highfat","highfat"))
model.matrix(~ group)
```

```
  (Intercept) group
1           1     1
2           1     1
3           1     2
4           1     2
attr(,"assign")
[1] 0 1

  (Intercept) grouphighfat
1           1            0
2           1            0
3           1            1
4           1            1
attr(,"assign")
[1] 0 1
attr(,"contrasts")
attr(,"contrasts")$group
[1] "contr.treatment"
```

# Formula extensions

```r
# More groups
group <- factor(c(1,1,2,2,3,3))
model.matrix(~ group)
```

```
  (Intercept) group2 group3
1           1      0      0
2           1      0      0
3           1      1      0
4           1      1      0
5           1      0      1
6           1      0      1
attr(,"assign")
[1] 0 1 1
attr(,"contrasts")
attr(,"contrasts")$group
[1] "contr.treatment"
```

```r
# Alternate formulation
group <- factor(c(1,1,2,2,3,3))
model.matrix(~ group + 0)
```

```
  group1 group2 group3
1      1      0      0
2      1      0      0
3      0      1      0
4      0      1      0
5      0      0      1
6      0      0      1
attr(,"assign")
[1] 1 1 1
attr(,"contrasts")
attr(,"contrasts")$group
[1] "contr.treatment"
```

# More variables

Assume diet effect is same across sexes

$$Y_i = \beta_0 + \beta_1 x_{i,1} + \beta_2 x_{i,2} + \varepsilon_i$$

```r
# Effect of diet and difference in sexes
diet <- factor(c(1,1,1,1,2,2,2,2))
sex <- factor(c("f","f","m","m","f","f","m","m"))
table(diet,sex)
# To fit the additive model
diet <- factor(c(1,1,1,1,2,2,2,2))
sex <- factor(c("f","f","m","m","f","f","m","m"))
model.matrix(~ diet + sex)
```

```
  (Intercept) diet2 sexm
1           1     0    0
2           1     0    0
3           1     0    1
4           1     0    1
5           1     1    0
6           1     1    0
7           1     1    1
8           1     1    1
attr(,"assign")
[1] 0 1 2
attr(,"contrasts")
attr(,"contrasts")$diet
[1] "contr.treatment"

attr(,"contrasts")$sex
[1] "contr.treatment"
```

```r
model.matrix(~ diet + sex + diet:sex)
```

```r
model.matrix(~ diet*sex)
```

```
  (Intercept) diet2 sexm diet2:sexm
1           1     0    0          0
2           1     0    0          0
3           1     0    1          0
4           1     0    1          0
5           1     1    0          0
6           1     1    0          0
7           1     1    1          1
8           1     1    1          1
attr(,"assign")
[1] 0 1 2 3
attr(,"contrasts")
attr(,"contrasts")$diet
[1] "contr.treatment"

attr(,"contrasts")$sex
[1] "contr.treatment"
```

# Reference level, data and continuous predictors

```r
# Releveling
group <- factor(c(1,1,2,2))
group <- relevel(group, "2")
model.matrix(~ group)
# Providing factor levels
group <- factor(group, levels=c("1","2"))
model.matrix(~ group)
```

```
  (Intercept) group1
1           1      1
2           1      1
3           1      0
4           1      0
attr(,"assign")
[1] 0 1
attr(,"contrasts")
attr(,"contrasts")$group
[1] "contr.treatment"


  (Intercept) group2
1           1      0
2           1      0
3           1      1
4           1      1
attr(,"assign")
[1] 0 1
attr(,"contrasts")
attr(,"contrasts")$group
[1] "contr.treatment"
```

```r
# Finding data
group <- 1:4
model.matrix(~ group, data=data.frame(group=5:8))
# Continuous variables
tt <- seq(0,3.4,len=4)
model.matrix(~ tt + I(tt^2))
# Different contrasts
d <- data.frame(time=factor(1:4))
model.matrix(~time,data=d, contrasts.arg= list(time="contr.sum"))
```

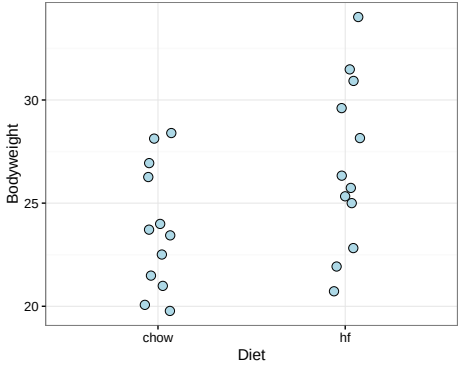
```
  (Intercept) group
1           1     5
2           1     6
3           1     7
4           1     8
attr(,"assign")
[1] 0 1

  (Intercept)          tt      I(tt^2)
1           1 0.000000000  0.000000000
2           1 1.133333333  1.284444444
3           1 2.266666667  5.137777778
4           1 3.400000000 11.560000000
attr(,"assign")
[1] 0 1 2

  (Intercept) time1 time2 time3
1           1     1     0     0
2           1     0     1     0
3           1     0     0     1
4           1    -1    -1    -1
attr(,"assign")
[1] 0 1 1 1
attr(,"contrasts")
```

# Linear models - a first taste

- Does high fat diet lead to higher average body weight in mice?
  - Simple *t*-test problem but will demonstrate equivalence with `lm`
- High fat diet group has higher weights, but with overlap
- Build design matrix $\mathbf{X}$ using formula `~Diet`
  - Group with the 1's in second column is determined by second level of 'Diet'

```r
dat <- read.csv("./data-raw/femaleMiceWeights.csv")
set.seed(1) #same jitter in stripchart
# Plot
ggplot(dat, aes(x=Diet, y=Bodyweight)) + geom_jitter(position=position_
    ↪ jitter(0.2), size=3, shape=21, bg="lightblue") + theme_bw()
# Design
levels(dat$Diet)
X <- model.matrix(~ Diet, data=dat)
head(X)
```

```
[1] "chow" "hf"
  (Intercept) Diethf
1           1      0
2           1      0
3           1      0
4           1      0
5           1      0
6           1      0
```



# The Mathematics Behind `lm()`

- Quick look behind the scenes of function `lm`
- Form the design matrix $\mathbf{X}$ and calculate $\beta$
  - Minimizes sum of squares
- Formula for solution is:

$$\hat{\beta} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{Y}$$

- Calculate in R using:
  - Matrix multiplication operator %*%
  - Inverse function `solve`, and
  - Transpose function `t`

```r
Y <- dat$Bodyweight
X <- model.matrix(~ Diet, data=dat)
solve(t(X) %*% X) %*% t(X) %*% Y
```

```
                  [,1]
(Intercept) 23.813333333
Diethf       3.020833333
```

Coefficients are average of control group and difference of averages

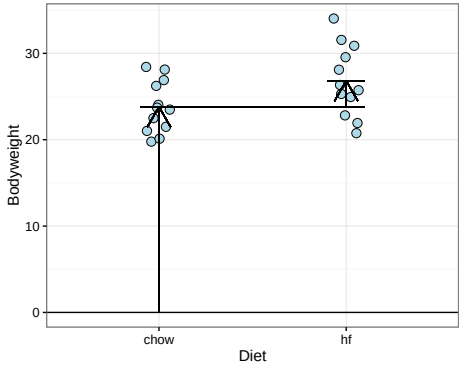
```r
s <- with(dat, split(Bodyweight, Diet))
mean(s[["chow"]])
mean(s[["hf"]]) - mean(s[["chow"]])
```

```
[1] 23.81333333
[1] 3.020833333
```

# Linear model using `lm()`

```r
fit <- lm(Bodyweight ~ Diet, data=dat)
summary(fit)
(coefs <- coef(fit))
```

```

Call:
lm(formula = Bodyweight ~ Diet, data = dat)

Residuals:
    Min      1Q  Median      3Q     Max
-6.104 -2.436 -0.414  2.834  7.186

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    23.81       1.04   22.91   <2e-16 ***
Diethf          3.02       1.47    2.06    0.052 .
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.6 on 22 degrees of freedom
Multiple R-squared:  0.161,     Adjusted R-squared:  0.123
F-statistic: 4.22 on 1 and 22 DF,  p-value: 0.0519

 (Intercept)       Diethf
23.813333333  3.020833333
```

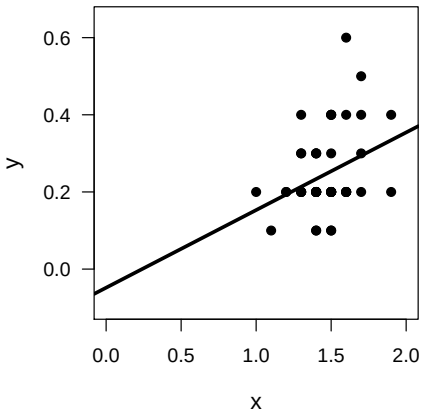



```r
mytt <- t.test(s[["hf"]], s[["chow"]], var.equal=TRUE)
summary(fit)$coefficients[2, 3]
mytt$statistic
```

```
[1] 2.055173992
        t
2.055173992
```

# Overarching regression framework

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$
$$\varepsilon_i \sim N\left(0, \sigma^2\right) \tag{1}$$




**Data**

$y$ = response variable

$x$ = predictor

**Parameters**

$\beta_0$ = intercept

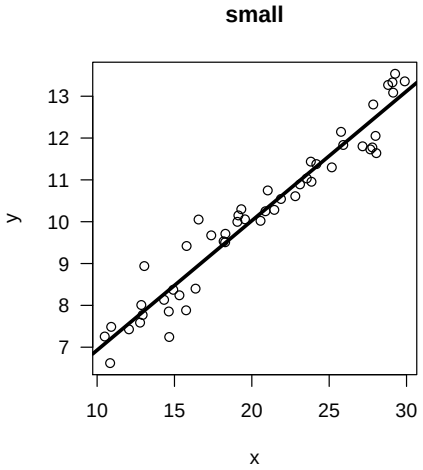
$\beta_1$ = slope

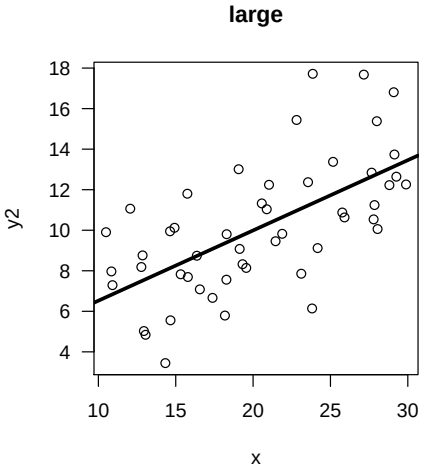
$\sigma$ = residual variation

$\varepsilon$ = residuals

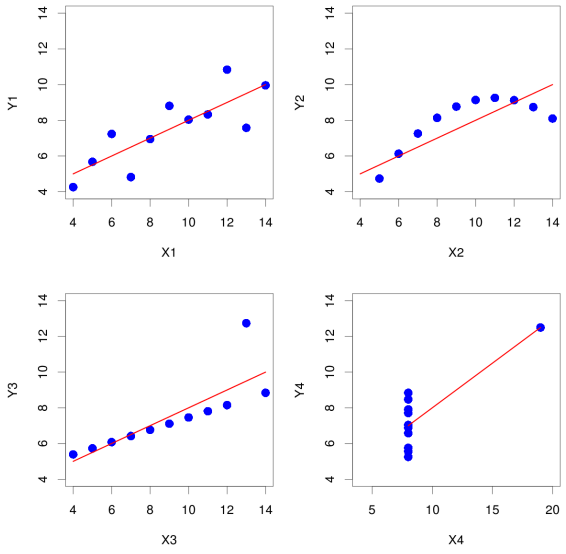
# Residual variation (error)

- Our model doesn't perfectly predict Y; residuals are used to estimate error of model



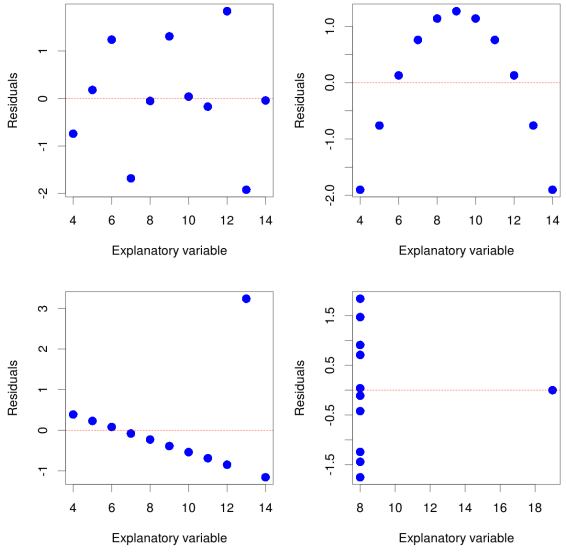
# Always plot your data first!



# Anscombe data - Linear models

- $\bar{x} = 9$ and $\bar{y} = 7.5$ for each dataset
- $s_x^2 = 11$, $s_y^2 = 4.1$
- For all data sets, fitted regression is same:
  - $\hat{Y} = 3.0 + 0.5 * X$
- All models have $R^2 = 0.67$, $\sigma^2 = 1.24$ and slope coefficients are significant at $< 1\%$ level

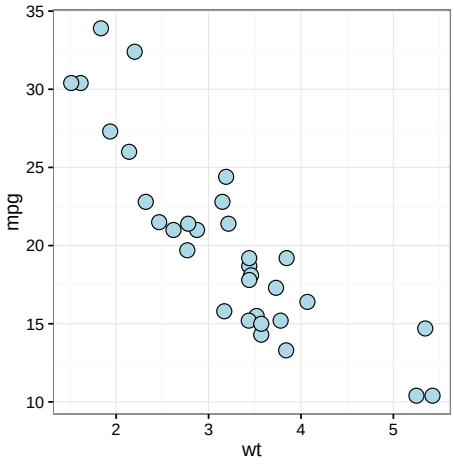
# Checking assumptions graphically - Anscombe data residual plots



# Exploratory Data Analysis (EDA)

```r
head(mtcars)
```

```
                   mpg cyl disp  hp drat   wt qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.62 16.5  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.88 17.0  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.32 18.6  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.21 19.4  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.44 17.0  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.46 20.2  1  0    3    1
```

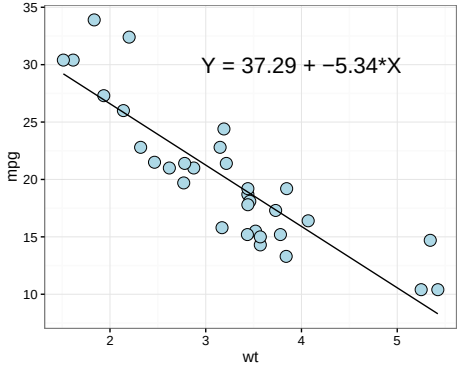


# Fitting, saving and working with a linear model

```
# Example:
fit <- with(mtcars, lm(mpg ~ wt))
summary(fit)
confint(fit)
```

```

Call:
lm(formula = mpg ~ wt)

Residuals:
    Min      1Q  Median      3Q     Max
-4.543 -2.365 -0.125  1.410  6.873

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   37.285      1.878   19.86  < 2e-16 ***
wt            -5.344      0.559   -9.56  1.3e-10 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.05 on 30 degrees of freedom
Multiple R-squared:  0.753,     Adjusted R-squared:  0.745
F-statistic: 91.4 on 1 and 30 DF,  p-value: 1.29e-10


            2.5 % 97.5 %
(Intercept) 33.45   41.1
wt          -6.49   -4.2
```

- Results of linear saved as:
  - `fit.lm <- lm(Y ~ X1 + X2, mydata)`
- `fit.lm` is a linear model **object**
- Use **extractor** functions to view
- e.g., 'summary', 'coef', 'residuals', and 'fitted'



# Model diagnostics

Estimation and inference from a linear model depend on several assumptions We check these assumptions using *regression diagnostics*; Assume. . .

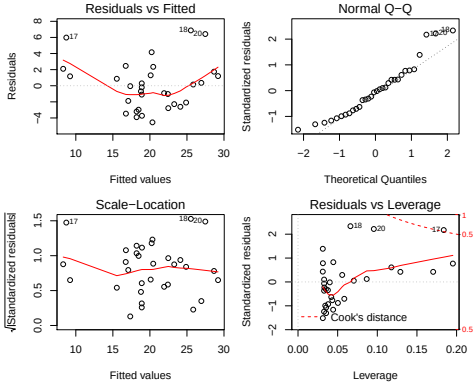
- errors are independent
- errors have constant variance
- errors are normally distributed
- all observations "fit" the model and none have large influence on the model

Violations of these assumptions can invalidate our model

# Model diagnostics

- Residuals vs Fitted (*check constant variance assumption*)
- Normal Q-Q (*check normality assumption*)
- Scale-Location (*check constant variance assumption*)
- Residuals vs Leverage (*check for influential observations*)

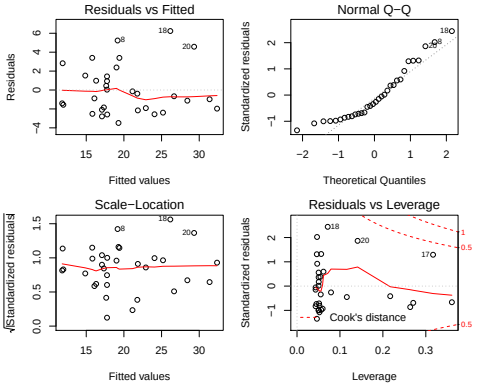
```
1  # Call the plot function on model object
2  plot(fit)
```



# Updating linear models - quadratic term

```r
# Example:
fit.poly2 <- update(fit, . ~ poly(wt, degree=2))
summary(fit.poly2)
confint(fit.poly2)
```

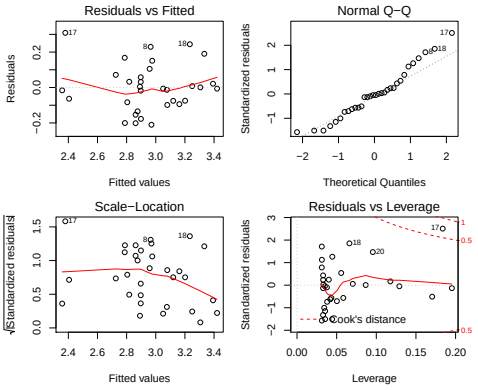
```

Call:
lm(formula = mpg ~ poly(wt, degree = 2))

Residuals:
   Min     1Q Median     3Q    Max
-3.483 -1.997 -0.773  1.462  6.238

Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)           20.091      0.469   42.88  < 2e-16 ***
poly(wt, degree = 2)1 -29.116      2.651  -10.98  7.5e-12 ***
poly(wt, degree = 2)2   8.636      2.651    3.26   0.0029 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.65 on 29 degrees of freedom
Multiple R-squared:  0.819,     Adjusted R-squared:  0.807
F-statistic: 65.6 on 2 and 29 DF,  p-value: 1.71e-11

                     2.5 % 97.5 %
(Intercept)          19.13   21.0
poly(wt, degree = 2)1 -34.54  -23.7
poly(wt, degree = 2)2   3.21   14.1
```



# Updating linear models - transformation

```r
# Example:
fit.log <- update(fit, log(mpg) ~ .)
summary(fit.log)
confint(fit.log)
```

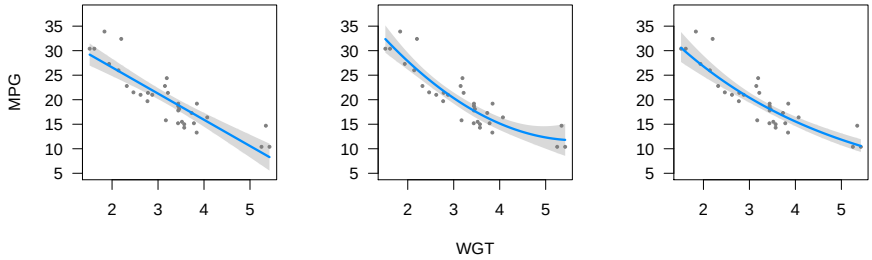
```

Call:
lm(formula = log(mpg) ~ wt)

Residuals:
     Min       1Q   Median       3Q      Max
-0.21035 -0.08593 -0.00614  0.06133  0.30862

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)    3.832      0.084    45.6  < 2e-16 ***
wt            -0.272      0.025   -10.9  6.3e-12 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.136 on 30 degrees of freedom
Multiple R-squared:  0.798,     Adjusted R-squared:  0.791
F-statistic:  118 on 1 and 30 DF,  p-value: 6.31e-12

             2.5 % 97.5 %
(Intercept)  3.660  4.003
wt          -0.323 -0.221
```



# Visualising linear models - visreg

- Comparison of fitted models

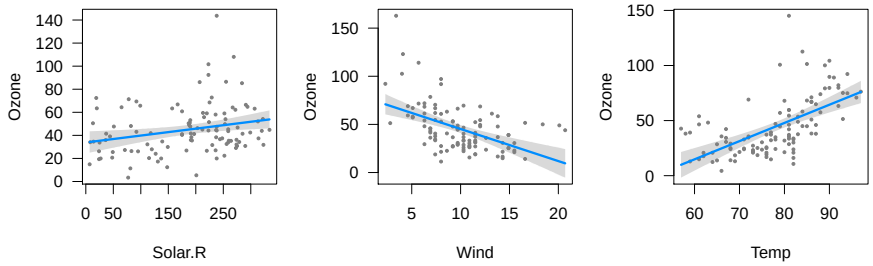
```
library(visreg)
visreg(fit, ylim = c(5, 38), ylab = "MPG", xlab = "")
visreg(fit.poly2, ylim = c(5, 38), ylab = "", xlab = "WGT")
visreg(fit.log, "wt", trans = exp, ylim = c(5, 38), partial = TRUE, rug = FALSE, ylab = "", xlab = "")
```



# Visualising linear models - `visreg`

- Particularly useful when you have multiple predictors, combinations of continuous and categorical predictors, interactions

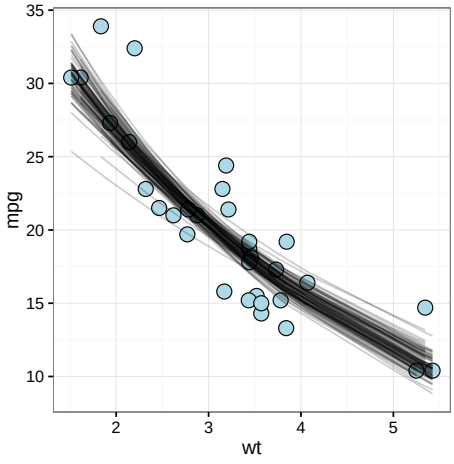
```
fitShow <- lm(Ozone ~ Solar.R + Wind + Temp, data = airquality)
visreg(fitShow)
```



# The extra mile - augment and bootstrap

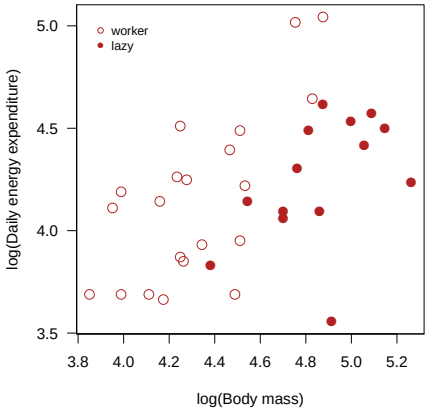
```r
# Add variables to data frame
library(broom)
aug.mtcars <- mtcars %>% do(augment(lm(mpg ~ wt, .)))
head(aug.mtcars)
# Bootstrap model predictions and plot
boot.aug <- mtcars %>% bootstrap(100) %>% do(augment(lm(log(mpg) ~ wt, .)
        ↪ ))
ggplot(mtcars, aes(wt, mpg)) + geom_point(size=4, shape=21, bg="lightblue
        ↪ ") + theme_bw() +
   geom_line(data = boot.aug, aes(x=wt, y=exp(.fitted), group=replicate),
           ↪  alpha=.2)
```

```
          .rownames  mpg   wt .fitted .se.fit .resid
1         Mazda RX4 21.0 2.62    23.3   0.634 -2.283
2     Mazda RX4 Wag 21.0 2.88    21.9   0.571 -0.920
3        Datsun 710 22.8 2.32    24.9   0.736 -2.086
4    Hornet 4 Drive 21.4 3.21    20.1   0.538  1.297
5 Hornet Sportabout 18.7 3.44    18.9   0.553 -0.200
6           Valiant 18.1 3.46    18.8   0.555 -0.693
```



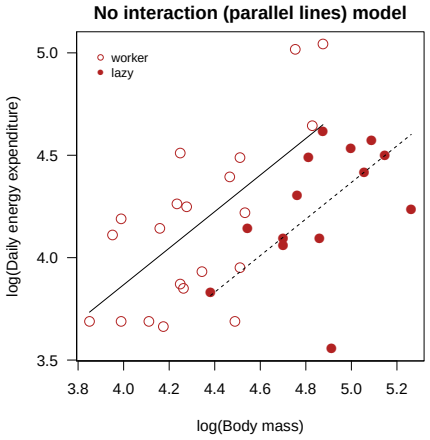
# Continuous and categorical explanatory variables

- Mole rats have distinct social castes
- Single queen and small number males reproducing, remaining are workers
- Damaraland mole rats have two worker castes (worker and lazy)
- Do the groups have physiological differences?



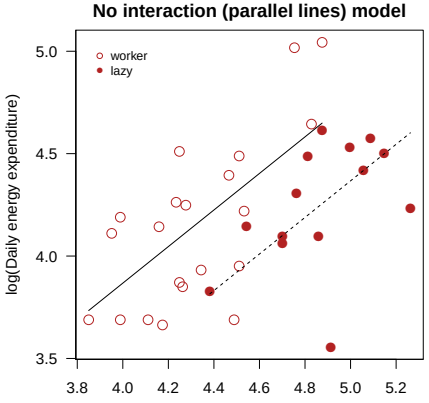
# Energy-body mass relationships for mole rats

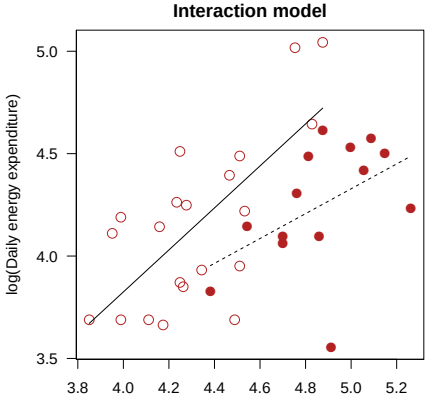
- Energy expenditure appears to vary with body mass
- But lazy workers are heavier than frequent workers
- How different is mean daily energy expenditure between caste groups when adjusted for differences in body mass?



# Energy-body mass relationships for mole rats

- Need to test for interaction - non-parallel slopes

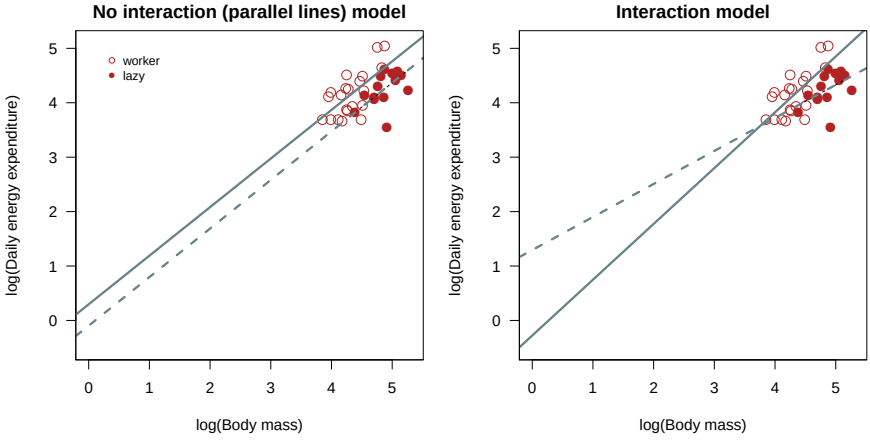


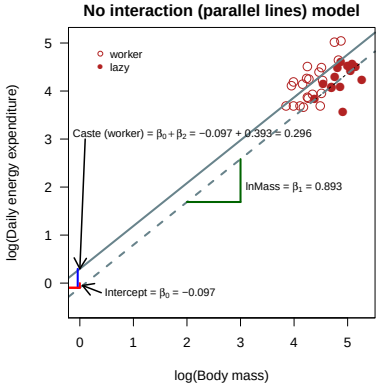

```
Analysis of Variance Table

Response: lnEnergy
             Df Sum Sq Mean Sq F value Pr(>F)
lnMass        1   1.31    1.31    14.9  5e-04 ***
caste         1   0.64    0.64     7.3   0.01 *
lnMass:caste  1   0.09    0.09     1.0   0.32
Residuals    31   2.72    0.09
```

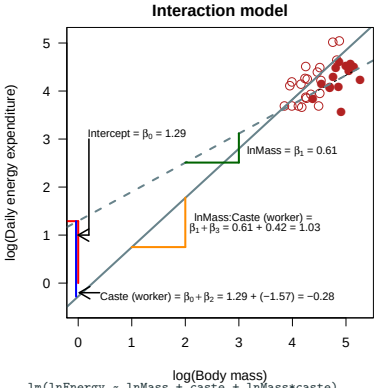
# Visualising the parameter space

- Extend plotting limits to 'see' intercepts



# Interpretation of parameters




```
lm(lnEnergy ~ lnMass + caste)

             Estimate Std. Error t value Pr(>|t|)
(Intercept)    -0.097      0.942    -0.1     0.92
lnMass          0.893      0.193     4.6   6e-05 ***
casteworker     0.393      0.146     2.7    0.01 *

Residual standard error: 0.3 on 32 degrees of freedom
Multiple R-squared:  0.41,Adjusted R-squared:  0.37
F-statistic:  11 on 2 and 32 DF,  p-value: 0.00022
```

```
lm(lnEnergy ~ lnMass + caste + lnMass*caste)

                 Estimate Std. Error t value Pr(>|t|)
(Intercept)          1.29       1.67     0.8     0.44
lnMass               0.61       0.34     1.8     0.09 .
casteworker         -1.57       1.95    -0.8     0.43
lnMass:casteworker   0.42       0.41     1.0     0.32

Residual standard error: 0.3 on 31 degrees of freedom
Multiple R-squared:  0.43,Adjusted R-squared:  0.37
F-statistic: 7.7 on 3 and 31 DF,  p-value: 0.00054
```

# Generalised Linear Models (GLMs) can ...

- Describe relationship between mean of response and linear predictor using a *link function*

- Describe relationship between variance and mean using a *variance function*

# Logistic Regression

- Sometimes our response is binary: yes/no, success/failure, diseased/non-diseased, etc. Usually coded as 1 and 0

- We use logistic regression to model the **log odds** of binary response

- Let $p$ be the probability that response is 1. Then linear model is:

$$log(\frac{p}{1-p}) = \beta_0 + \beta_1 x_1 + \cdots + \beta_k x_k$$

# Converting log odds to probability

- We usually transform log odds to probability when making predictions

- Say $log(\frac{p}{1-p}) = x$. We transform to probability as follows:

$$p = \frac{e^x}{1 + e^x}$$

- R can easily do this for us (`plogis`)

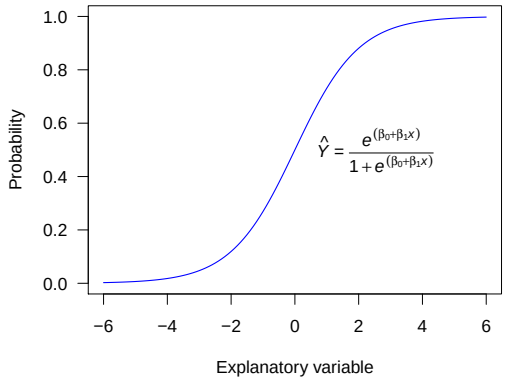
# Doing Logistic Regression in R

- Logistic regression is a type of *generalized linear model* (`glm`)
- We fit GLM models in R using the `glm()` function. It works like the `lm()` function except we specify which `glm` to fit using the `family` argument
- Logistic regression requires `family=binomial`
  - Default link is `logit`, but should be specified for transparency
  - Alternatives are `probit` and `cloglog`

```
mod <- glm(y ~ x, data=yourdata, family=binomial)
summary(mod)
fitted(mod) # gives predicted probabilities
```

# Generalised linear model for binary data - inverse link

- The logistic function has the advantage that it:
  - Yields probabilities between 0 and 1
  - Can incorporate relationships with explanatory variables through the $\beta$ parameters



# Generalised linear model for binary data - link function

- So our model for the probability that $Y = 1$ is the logistic function:
  - $\hat{Y} = \frac{e^{\beta_0 + \beta_1 * X_1 + \cdots + \beta_p * X_p}}{1 + e^{\beta_0 + \beta_1 * X_1 + \cdots + \beta_p * X_p}}$
- But, the logistic is a nonlinear function, so we cannot estimate the $\beta$ parameters using a (generalised) linear model
- Remember that we have used the inverse link $g^{-1}(\hat{Y})$
- So we have to specify this in terms of the *link function* itself, not its inverse
  - This is the *logit link* (*logit* is the *log* of the *odds*)
  - $g(\hat{Y}) = \textit{logit}(\hat{Y}) = \log(\frac{\hat{Y}}{1 - \hat{Y}})$

# Example: Binomial GLM horseshoe crabs

- Nesting horseshoe crabs; female with male in nest
- Do female characteristics dictate satellite males?
- $H_A$: Big females have more satellites

# Example: Binomial GLM horseshoe crabs

- Binary response; female with satellite male or not
- Carapace width is measure of female fitness

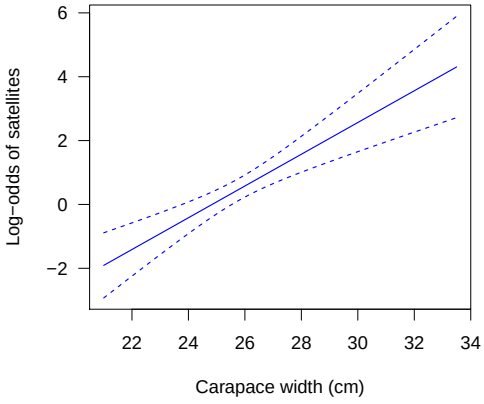
```
satt <- read.csv("./data-raw/satellites.csv", header = TRUE)
satt$satPresent <- (satt$nsatellites > 0)*1
summary(satt)
satt.binom <- glm(satPresent ~ width.cm, data = satt, family = binomial("
    ↪ logit"))
```

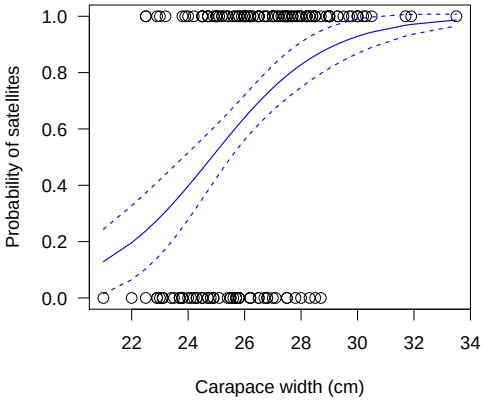
```
          color    spine width.cm nsatellites mass.kg satPresent
1       medium both.bad     28.3           8    3.05          1
2  dark-medium both.bad     22.5           0    1.55          0
3 light-medium      good     26.0           9    2.30          1
4  dark-medium both.bad     24.8           0    2.10          0
5  dark-medium both.bad     26.0           4    2.60          1
6       medium both.bad     23.8           0    2.10          0
```

```
Call:
glm(formula = satPresent ~ width.cm, family = binomial("logit"),
    data = satt)

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-2.028  -1.046   0.548   0.907   1.694

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -12.351      2.629   -4.70  2.6e-06 ***
width.cm       0.497      0.102    4.89  1.0e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 225.76  on 172  degrees of freedom
Residual deviance: 194.45  on 171  degrees of freedom
AIC: 198.5

Number of Fisher Scoring iterations: 4

              2.5 % 97.5 %
(Intercept) -17.810 -7.457
width.cm      0.308  0.709
Single term deletions

Model:
satPresent ~ width.cm
         Df Deviance   AIC   LRT Pr(>Chi)
<none>          194.4 198.4
width.cm  1    225.8 227.8 31.31  2.2e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

# Example: Binomial GLM horseshoe crabs

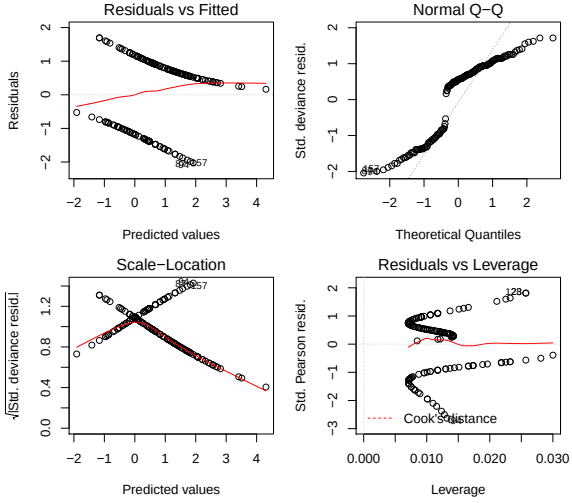
- Binary response; female with satellite male or not
- Carapace width is measure of female fitness



# Example: Binomial GLM horseshoe crabs

- Standard residual diagnostic plots are not informative



# Challenge: Survival of passengers on the Titanic

- Sinking of the RMS Titanic is one of most infamous shipwrecks in history
- One of reasons that shipwreck led to such loss of life was that there were not enough lifeboats for passengers and crew
- Although there was some element of luck involved in surviving the sinking, some groups of people were more likely to survive than others, such as women, children, and the upper-class

**Challenge question**

What sorts of people were likely to survive?

This is actually part of a kaggle competition

# Challenge: Survival of passengers on the Titanic

```r
titanic <- read.csv("data-raw/titanic_long.csv")
head(titanic)
```

```
  class   age  sex survived
1 first adult male        1
2 first adult male        1
3 first adult male        1
4 first adult male        1
5 first adult male        1
6 first adult male        1
```