

# Yesterday...

- Understand discrete and continuous r.v.
- Plot pmf/pdf in R
- Calculate mean and variance of some r.v. by hand (then verify in R)
- Generate some common r.v. in R

- BTW readings on blackboard
- Charlesworth & Charlesworth (2010)

# Today

- Multivariate random variable
- Independence
- Likelihood function (finally!)

# Multivariate r.v.

- Sometimes events happen at the same time, or interact with each other
- e.g. Rolling two dice, flipping two coins, the temperature in London and Oxford
- The pmf/pdf isn't univariate anymore
- For two r.v.,  $X$  and  $Y$ , the joint distribution is  $f_{XY}(x, y)$
- $f_{XY}(x, y)$  looks like a landscape.

- Bivariate normal distribution
- $f(x, y) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\left(\frac{(x-\mu_1)^2}{\sigma_1^2} - \frac{2\rho(x-\mu_1)(y-\mu_2)}{\sigma_1\sigma_2} + \frac{(y-\mu_2)^2}{\sigma_2^2}\right)$
- $\begin{pmatrix} X \\ Y \end{pmatrix} \sim MVN\left(\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \Sigma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}\right)$
- <http://socr.ucla.edu/htmls/HTML5/BivariateNormal/>

# Independence

- Two events are **independent** if the occurrence of one does not affect the probability of the other.
- i.e. Event A gives no extra information about the occurrence of Event B
- Perhaps the strongest assumption in statistics (we cannot actually test for independence).
- If  $X$  and  $Y$  are independent then  $cor(X, Y) = 0$
- But  $cor(X, Y) = 0$  does **NOT** imply independence!

- If  $X$  and  $Y$  are independent, then the joint probability density function of  $X$  and  $Y$  is just the product of their own:

$$f_{XY}(x, y) = f_X(x)f_Y(y)$$

- Remember our definition to independence: “The outcome of  $X$  provides no extra information about  $Y$ ”
- $var(X + Y) = var(X) + var(Y)$
- This is very important when we construct our likelihood function later on.

# Maximum Likelihood Estimation

- Likelihood is the central idea of statistics
- Invented (?) by Sir Ronald Fisher in this country
- “One of the greatest ideas of the 20<sup>th</sup> century and probably one of the greatest of civilisation” –Dan Reuman, who taught this module last year



# MLE

- Maximum Likelihood estimation (MLE) is a method to estimate parameters of a statistical model
- When the method is applied to a **dataset** with a statistical **model**, maximum likelihood estimation provides estimates for the **parameters**.
- “The parameter values that makes the observed dataset most *probable*.”

- The likelihood function  $L(\underline{\theta})$  is used to quantify such “probability”. The symbol  $\underline{\theta}$  is used to denote a vector of parameters.

$x_i$  given  $\theta$

- If  $\underline{x} = x_1, \dots, x_n$  are i.i.d. samples from a population with pdf (or pmf)  $f(x_i|\underline{\theta})$ , then likelihood function is defined by the **joint density** of  $\underline{x}$

$$L(\underline{\theta}|\underline{x}) = f(x_1, \dots, x_n|\underline{\theta}) = \prod_{i=1}^n f(x_i|\underline{\theta})$$

- Once  $\underline{x}$  is observed,  $L(\underline{\theta}|\underline{x})$  **becomes a function of  $\underline{\theta}$  only** (does not depend on anything else)

- For each sample  $\underline{x}$  (fixed) and given a model, let  $\hat{\theta}$  be a parameter value at which  $L(\theta|\underline{x})$  attains its maximum.  $\hat{\theta}$  is the maximum likelihood estimate for the observed data  $\underline{x}$ .
- Maximising the log-likelihood function is equivalent to maximising the likelihood function.

- Treat the parameters as unknown (a bit counter-intuitive)
- The triplets:
  - Model
  - Parameters
  - Data

# Example 1: Coin tossing


- If we flip 10 coins, independently, and observe 7 heads and 3 tails
- If we define  $p$  as  $Prob(head)$ , what is the MLE of  $p$ ?
- Each coin toss is a Bernoulli trial, and the joint density of 10 independent coin tosses is *binomial*( $n, p$ ).
- Let  $Y$  be the number of heads out of 10 tosses
$$f(Y = y) = C_y^{10} p^y (1 - p)^{10-y}$$

- Now, put  $y = 7$  as this is what we observed

$$f(Y = 7) = C_7^{10} p^7 (1 - p)^{10-7}$$

- And this is just our likelihood function

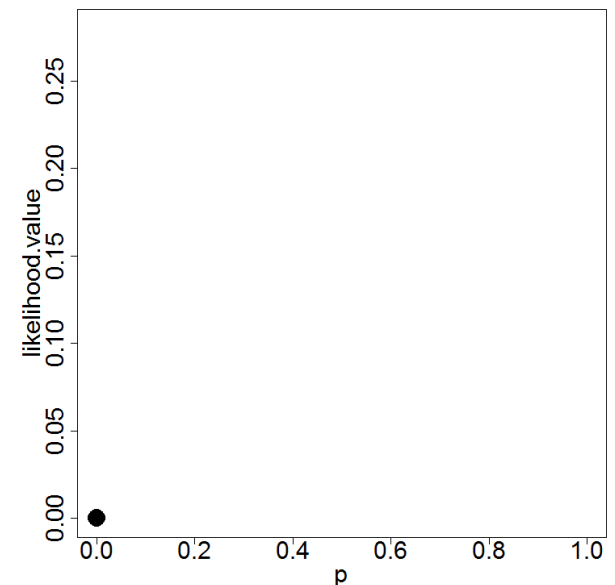
$$L(p) = f(Y = 7) = C_7^{10} p^7 (1 - p)^3$$



The likelihood function depends on  $p$  only after observing the data.

- For each value of  $p$ , there is a corresponding value of the likelihood function  $L(p)$

$p$	$L(p)$
0	$C_7^{10} 0^7 (1 - 0)^3 = 0$
0.1	$C_7^{10} 0.1^7 0.9^3 = 8.748 \times 10^{-6}$
0.2	$C_7^{10} 0.2^7 0.8^3 = 0.000786$
0.3	$C_7^{10} 0.3^7 0.7^3 = 0.0090$
0.4	$C_7^{10} 0.4^7 0.6^3 = 0.0424$
$\vdots$	$\vdots$



# Some R code

```
# WRITE OUR OWN LIKELIHOOD FUNCTION
binomial.likelihood<-function(p) {
  choose(10,7)*p^7*(1-p)^3
}

# LET'S CHECK WHAT THE LIKELIHOOD VALUE IS FOR p=0.1
binomial.likelihood(p=0.1)
      # YOU GOT SOMETHING AROUND 8.748e-06, RIGHT?

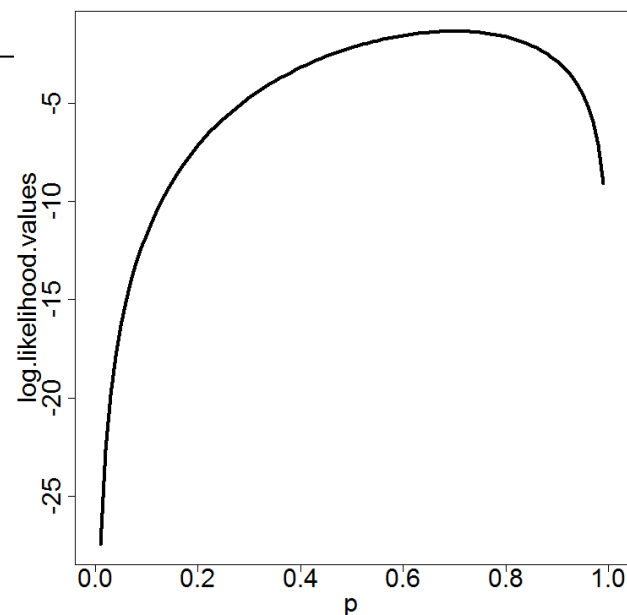
# PLOT THE LIKELIHOOD FUNCTION AGAINST p
p<-seq(0,1,0.01)
likelihood.values<-binomial.likelihood(p)
plot(p, likelihood.values, type='l')
```

Always ?function\_name  
if you have doubts.  
Say, ?choose

```
# WE CAN LOOK AT THE LOG-LIKELIHOOD AS WELL
# WE CAN REUSE THE FUNCTION WE'VE JUST WRITTEN
log.binomial.likelihood<-function(p) {
log(binomial.likelihood(p=p))
}

# PLOT THE LOG-LIKELIHOOD
p<-seq(0,1,0.01)
log.likelihood.values<-log.binomial.likelihood(p)
plot(p, log.likelihood.values, type='l')
```

We can see that both the likelihood and log-likelihood function are maximised when  $p$  is around 0.7 (our best guess)

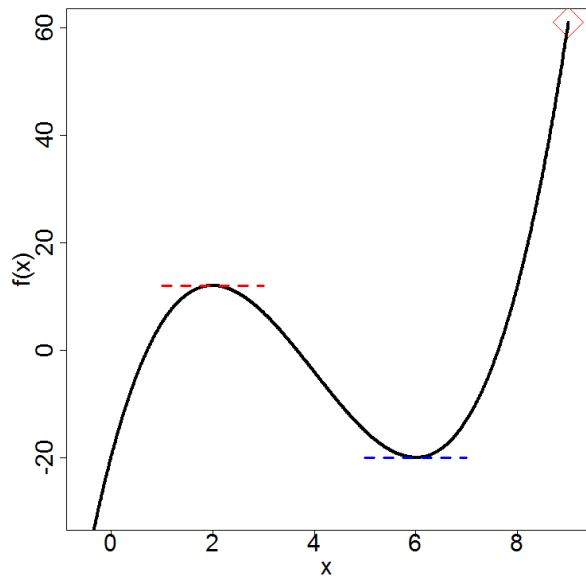




- Remember in day 1 we made some **probabilistic** statements: If we toss  $n$  fair coins, independently, then the probability of getting 7 heads out of 10 tosses is
- Today we make some **statistical inferences**: If we observed 7 heads out of 10 tosses, what can we say about the coin?

# Maximisation: some mathematical considerations

- If we want to maximise the likelihood function, we may need to solve  $\frac{\partial l}{\partial \theta} = 0$ . We also need to check for other conditions as well.
- Some knowledge in calculus is required, and things can be very complicated if we have more than one parameter (multivariate calculus).



- However, in many real cases, because of the complexity of the model, or the high dimensionality of the parameters (or both!), MLE cannot be solved explicitly.
- The good thing is, in R, we can do it through `optim()` or `optimize()`.

```
optimize(binomial.likelihood, interval=c(0,1), maximum=TRUE)
```

```
$maximum  
[1] 0.6999843
```

```
$objective  
[1] 0.2668279
```

# Solve MLE analytically

In general, if we obtain  $y$  heads out of  $n$  tosses, the likelihood function is

$$L(p) = f(y|p) = C_y^n p^y (1 - p)^{n-y}$$

and the log-likelihood is

$$l(p) = \ln(L(p)) = \ln(C_y^n) + y \ln p + (n - y) \ln(1 - p)$$

Differentiate  $l(p)$  w.r.t.  $p$

$$\frac{\partial}{\partial p} l(p) = 0 + y \left( \frac{1}{p} \right) + (n - y) \left( \frac{-1}{1 - p} \right)$$

Then find  $p = \hat{p}$  such that  $\frac{\partial}{\partial p} l(p)|_{p=\hat{p}} = 0$

$$\frac{y}{\hat{p}} + (n - y) \left( \frac{-1}{1 - \hat{p}} \right) = 0$$

$$\frac{y}{\hat{p}} = \frac{n - y}{1 - \hat{p}}$$

$$\dots$$
$$\hat{p} = \frac{y}{n}$$

$\hat{p}$  is called the maximum likelihood estimator for the parameter  $p$

# Example 2: i.i.d. normal samples

- $X_1, X_2, \dots, X_n$  are i.i.d. random samples from  $N(\mu, 1)$ . Variance is known but we need to estimate the mean.

- Parameter of interest:  $\mu$

- $L(\mu) = f(x_1, x_2, \dots, x_n) = \prod_{i=1}^n f(x_i)$

$$= \prod_{i=1}^n \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{(x_i - \mu)^2}{2}\right)$$

$$= \left(\frac{1}{\sqrt{2\pi}}\right)^n \exp\left(-\frac{1}{2} \sum_{i=1}^n (x_i - \mu)^2\right)$$

Because of  
independence!

$x_i$  are the observed  
samples, fixed.

$\mu$  is the only quantity to be  
estimated.

The log-likelihood is

$$l(\mu) = \text{constant} - \frac{1}{2} \left( \sum_{i=1}^n (x_i - \mu)^2 \right)$$

Differentiate the log-likelihood wr.t.  $\mu$

$$\frac{\partial l}{\partial \mu} = 0 - \frac{1}{2} \left[ -2 \sum_{i=1}^n (x_i - \mu) \right]$$

Does not depend on  $\mu$

Find  $\mu = \hat{\mu}$  such that the derivative is zero

$$\sum_{i=1}^n (x_i - \hat{\mu}) = 0$$

$$\sum_{i=1}^n x_i - n\hat{\mu} = 0$$

$$\hat{\mu} = \frac{\sum_{i=1}^n x_i}{n}$$

So according to MLE, the best guess for  $\mu$  is the sample average (mean) of all the samples.

# Example 3: normal samples with unknown variance

- $X_1, X_2, \dots, X_n$  are i.i.d. random samples from  $N(\mu, \sigma^2)$ . Both  $\mu, \sigma^2$  are unknown.
- Parameters of interest:  $\mu, \sigma^2$  (bivariate parameter space)
- Similar to the previous example, the likelihood function is  $L(\mu, \sigma^2) = \left(\frac{1}{\sqrt{2\pi\sigma^2}}\right)^n \exp\left(-\frac{1}{2}\sum_{i=1}^n \left(\frac{x_i - \mu}{\sigma}\right)^2\right)$

$$l(\mu, \sigma^2) = -\frac{n}{2}\ln(2\pi\sigma^2) - \frac{1}{2}\sum_{i=1}^n \left(\frac{x_i - \mu}{\sigma}\right)^2$$

We need to find  $\frac{\partial l}{\partial \mu}$  and  $\frac{\partial l}{\partial \sigma^2}$  (exercise)

The remaining question is to find  $(\hat{\mu}, \hat{\sigma}^2)$  such that  $\frac{\partial l}{\partial \mu} = 0$  and  $\frac{\partial l}{\partial \sigma^2} = 0$  simultaneously. (more exercise!)

You are getting there!



# Example 4: Linear regression

- The model

$$y_i = a + bx_i + \varepsilon_i$$

with i.i.d. normally distributed error term  $\varepsilon_i \sim N(0, \sigma^2)$ ,  $i = 1, 2, \dots, n$

- Data:  $\begin{cases} x_i & \text{independent variable} \\ y_i & \text{response} \end{cases}$
- Parameters:  $\underline{\theta} = \begin{cases} a & \text{intercept} \\ b & \text{slope} \\ \sigma^2 & \text{variance} \end{cases}$
- If we rearrange the terms such that  $\varepsilon_i$  is the subject, the model becomes  $\varepsilon_i = y_i - a - bx_i$

$$\begin{aligned} L(\underline{\theta}) &= f(\underline{\varepsilon}|\underline{\theta}) = f(\varepsilon_1|\underline{\theta}) f(\varepsilon_2|\underline{\theta}) \cdots f(\varepsilon_n|\underline{\theta}) \\ &= \prod_{i=1}^n f(\varepsilon_i|\underline{\theta}) \\ &= \prod_{i=1}^n f(y_i - a - bx_i|\underline{\theta}) \end{aligned}$$

Note:  $f(\varepsilon_i|\underline{\theta})$  is the pdf of normal distribution

- And the log-likelihood becomes

$$l(\underline{\theta}) = \sum_{i=1}^n \ln(f(\varepsilon_i | \underline{\theta}))$$

Can you see why we prefer log-likelihood to the original likelihood?

- We can find a set of  $a$ ,  $b$  and  $\sigma^2$  such that the likelihood function is maximised.
- Can we write down the log-likelihood function in R?
- Can we optimise the log-likelihood function?
- Now we have a dataset on a mark-recapture experiment

# This afternoon

- A rabbit example
  - Get your hands dirty...

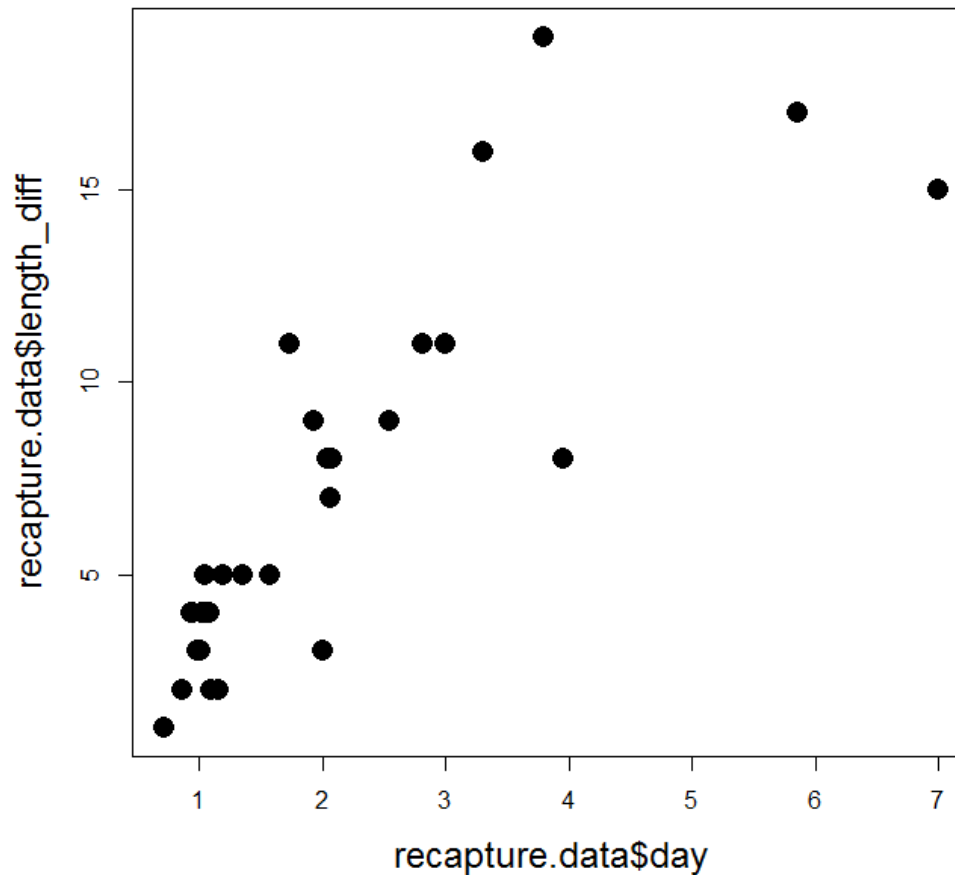
# Rabbit example

- A number of rabbits are marked, then some 29 rabbits are recaptured
- `day` measures the days between mark and recapture (Explanatory variable)
- `diff_length` is the grow in body length between mark and recapture (Response)
- We want to look at the relationship between these two variables



```
# READ IN DATASET
recapture.data<-read.csv('recapture.csv', header=T)

# PLOT IT OUT
plot(recapture.data$day, recapture.data$length_diff)
```



# R code for the log-likelihood function

```
# THE LOG-LIKELIHOOD FOR THE LINEAR REGRESSION
# PARAMETERS HAVE TO BE INPUT AS A VECTOR
regression.log.likelihood<-function(parm, dat)
{
# DEFINE THE PARAMETERS parm
# WE HAVE THREE PARAMETERS: a, b, sigma. BE CAREFUL OF THE ORDER
a<-parm[1]
b<-parm[2]
sigma<-parm[3]

# DEFINE THE DATA dat
# FIRST COLUMN IS x, SECOND COLUMN IS y
x<-dat[,1]
y<-dat[,2]

# DEFINE THE ERROR TERM
error.term<-(y-a-b*x)

# REMEMBER THE NORMAL pdf?
density<-dnorm(error.term, mean=0, sd=sigma, log=T)

# THE LOG-LIKELIHOOD IS THE SUM OF INDIVIDUAL LOG-DENSITY
return(sum(density))
}
```

```
# JUST TO SEE WHAT THE LOG-LIKELIHOOD VALUE IS WHEN a=1, b=1, and sigma=1
# YOU MAY TRY ANY DIFFERENT VALUES
regression.log.likelihood(c(1,1,1), dat=recapture.data)
```

```
[1] -452.6903
```

```
# TO OPIMISE THE LOG-LIKELIHOOD FUNCTION IN R
# optimize() IS ONE-DIMENSIONAL,
# optim() GENERALISES TO MULTI-DIMENSIONAL CASES

optim(par=c(1,1,1), regression.log.likelihood, method='L-BFGS-B',
      lower=c(-1000,-1000,0.0001), upper=c(1000,1000,10000),
      control=list(fnscale=-1), dat=recapture.data, hessian=T)
```

```
$par
[1] 1.527870 2.676240 2.678428
```

```
$value
[1] -69.72089
```

```
$counts
function gradient
      40      40
```

```
$convergence
[1] 0
```

```
$message
[1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

par=c(1,1,1)	Initial values for the parameters
log.likelihood.regression	The function you wish to be optimised
method='L-BFGS-B'	Optimisation algorithm
lower=c(-1000,-1000,0.0001)	Lower bound of your parameter space
upper=c(1000,1000,10000)	Upper bound of your parameter space
control=list((fnscale=-1))	fnscale=-1 means to maximise

# Notes on using `optim()`

- You always need to give it an initial parameter vector by `par=`
- Choice of `method` can be tricky for advanced users: See R help for details
- If you use `L-BFGS-B`, then you need to specify the `upper` and `lower` bound of the parameter values for searching for the maximum. No need to specify if you use `Nelder-Mead`
- If you wish to maximise a function, put `fnscale=-1` in your `control` list, default is to minimise. You can put multiple control parameters in the `control` list.
- Precision can be adjusted by tolerance or maximum number of iterations, say `maxit` or `abstol` within `control`
- The Hessian matrix (optional) tells you about the variance-covariance structure of your parameter estimates
- Try multiple sets of initial parameters and ensure they all go 'home'



- “Stumble around” the parameter space towards the best parameters, just like a drunkard trying to stumble home (the best place).
- Not every step is in the right direction, and it takes some time to go home.
- Ideal if the drunkard find his place.
- But also he may get stuck at the local minimum (not the most comfortable place, but, still, okay, like the tube station?)

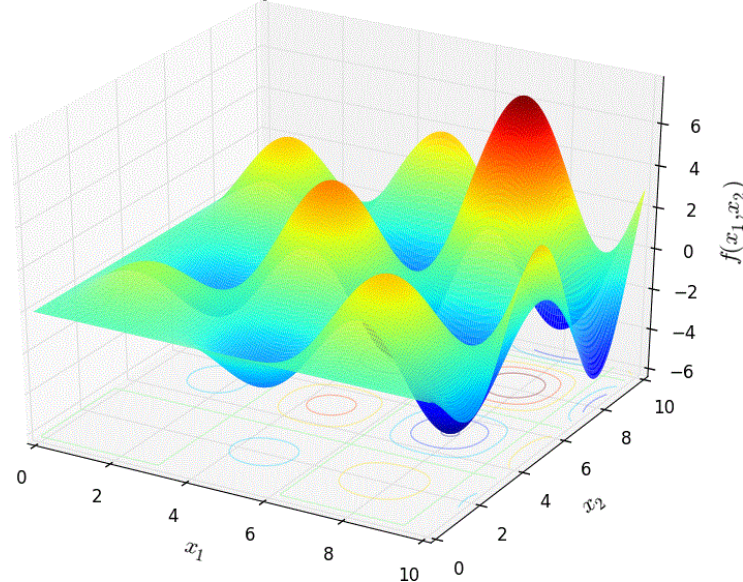


Photo credit: Dan Reuman

- Of course you can perform the same analysis with `lm()`

```
# REGRESSION WITH THE BUILT-IN lm()
m<-lm(length_diff~day, data=recapture.data)
summary(m)

> summary(m)

Call:
lm(formula = length_diff ~ day, data = recapture.data)

Residuals:
    Min       1Q   Median       3Q      Max
-5.2499 -1.2226 -0.1297  0.9099  7.3179

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)   1.5279     0.8833   1.730   0.0951 .
day           2.6762     0.3464   7.725 2.62e-08 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.776 on 27 degrees of freedom
Multiple R-squared:  0.6885,    Adjusted R-squared:  0.677
F-statistic: 59.67 on 1 and 27 DF,  p-value: 2.622e-08
```

```
n<-nrow(recapture.data)
sqrt(var(m$residual)*(n-1)/n)
```

```
[1] 2.6784281
```