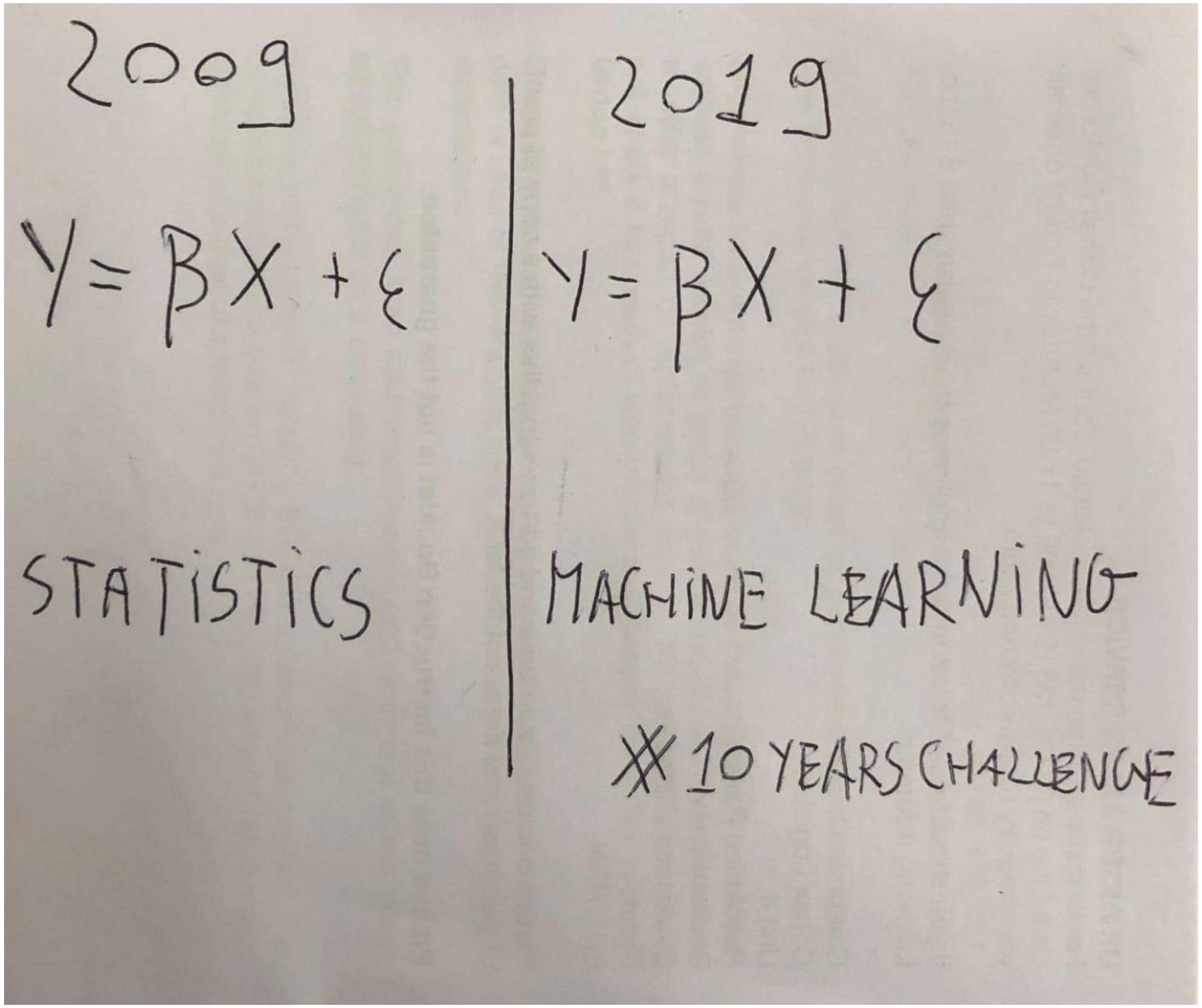


2009

$Y = \beta X + \varepsilon$

STATISTICS

2019

$Y = \beta X + \varepsilon$

MACHINE LEARNING

#10 YEARS CHALLENGE

Yesterday we…

- Learned about some common discrete and continuous r.v.
- Plotted some pmf/pdf in R
- Calculated statistical moments of r.v.
- Simulated some common r.v.

Today

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

Multivariate r.v.

- Sometimes events happen at the same time, or interact with each other. For example,
  - allele frequencies at different loci (genetic linkage)
  - population sizes of species within a dynamical / eco system
  - wind speed and rainfall in the same location
  - different traits of an individual
  - stock prices

- The pmf/pdf is multi-dimensional
- For bivariate case, the joint distribution of the two r.v. X and Y is denoted by $f_{XY}(x, y)$
- $f_{XY}(x, y)$ looks like a landscape, 3D plot

Bivariate normal r.v.

- Support: $\mathbf{R}^2$ (two-dimensional real number plane)
- pdf: $f_{XY}(x,y) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}}\exp(\frac{(x-\mu_1)^2}{\sigma_1^2} - \frac{2\rho(x-\mu_1)(y-\mu_2)}{\sigma_1\sigma_2} + \frac{(y-\mu_2)^2}{\sigma_2^2})$
- Parameters: mean vector $\boldsymbol{\mu} = (\mu_1, \mu_2)$ and variance-covariance matrix $\boldsymbol{\Sigma}$
- $\binom{X}{Y} \sim MVN\left(\boldsymbol{\mu} = \binom{\mu_1}{\mu_2}, \boldsymbol{\Sigma} = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix}\right)$
- http://socr.ucla.edu/htmls/HTML5/BivariateNormal/

Marginal distribution (optional)

- Given $f_{XY}(x, y)$ the joint pdf. Sometimes we are only interested in one of them
  - e.g. we would like to obtain the marginal pdf of X
  - without referencing to the values of Y
- $f_X(x) = \int_{-\infty}^{+\infty} f_{XY}(x, y)dy$
  - integrate (marginalise) out the uninterested r.v. Y

- Similarly, the marginal of Y is $f_Y(y) = \int_{-\infty}^{+\infty} f_{XY}(x, y)dx$

Conditional distribution (optional)

- If I know $Y = y$, this may give extra information on another r.v. X
- This conditional distribution of X is

$$f_{X|Y}(x|y) = \frac{f_{XY}(x,y)}{f_Y(y)}$$

- $X|Y$ reads as 'X given Y'
- $f_{X|Y}(x|y)$ is a slice of the joint pdf at $Y = y$

- $f_{Y|X}(y|x)f_X(x) = f_{XY}(x,y) = f_{X|Y}(x|y)f_Y(y)$
  - "Joint = conditional × marginal"
  - MCMC and Bayesian!

Covariance and Correlation

- Describe the (linear) association between two r.v.

- $cov(X,Y) = E[XY] - E[X]E[Y]$
  - $E[XY] = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} xyf_{XY}(x,y)dxdy$

- $cor(X,Y) = \frac{cov(X,Y)}{\sqrt{var(X)var(Y)}}$
  - bounded between -1 and +1

Independence

- Two events are independent if the occurrence of one does not affect the occurrence of another.
  - i.e. gives no extra information
- 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!

- Two r.v. X and Y are independent if and only if their joint probability density/mass function is the product of their marginal distributions:

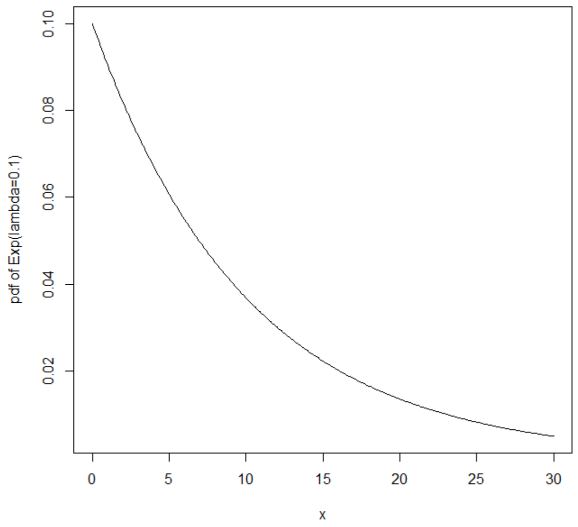
$$f_{XY}(x, y) = f_X(x) f_Y(y)$$

- Remember our definition to independence: “The outcome of X provides no extra information about Y”

-PAUSE-

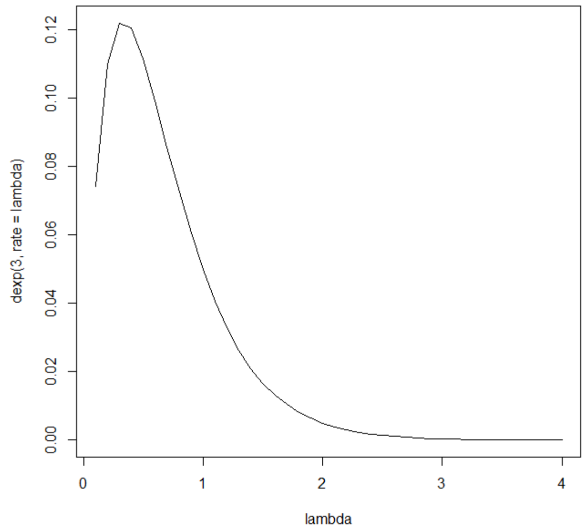
- So far we have been studying r.v., their outcomes, expectations etc.
  - given a fixed parameter value λ
- For example, if the waiting time for a bus is exponentially distributed with rate $\lambda = 0.1$
- $f(x;\lambda) = \lambda e^{-\lambda x} = 0.1 * e^{-0.1x}$

- Express f as a function of x



-PAUSE-

- Alternatively, if we have waited 3 units of time before getting on a bus
- $f(x;\lambda) = \lambda e^{-\lambda x} = \lambda e^{-3\lambda}$
- Express f as a function of λ
- Inference on the parameter λ
- Statistics!



Maximum Likelihood Estimation

- Likelihood is the central idea of statistics
- Invented (?) by Sir Ronald Fisher
- “One of the greatest ideas of the 20th century and probably one of the greatest of civilization” – Dr Dan Reuman, a co-founder of this MSc course.

Maximum Likelihood Estimation

- 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, MLE provides estimates for the associated parameters.

- "The parameter values that make the observed dataset most *probable*."

- The likelihood function $L(\underline{\theta})$ is used to quantify how "likely" the parameter values are. The symbol $\underline{\theta}$ denotes a vector of parameters.

x_i given θ

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

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

- Once $\underline{x}$ is observed, $L\left(\underline{\theta} \middle| \underline{x}\right)$ becomes a function of $\underline{\theta}$ only

- For each sample $\underline{x}$ (fixed) and given a model, let $\hat{\theta}$ be a parameter value at which $L\left(\underline{\theta}\middle|\underline{x}\right)$ 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 unknowns (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 for 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

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

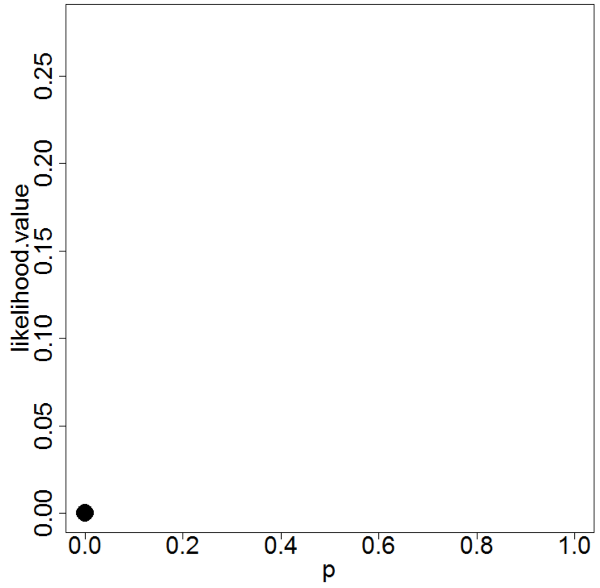
- And this is just our likelihood function

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

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 $L(p)$

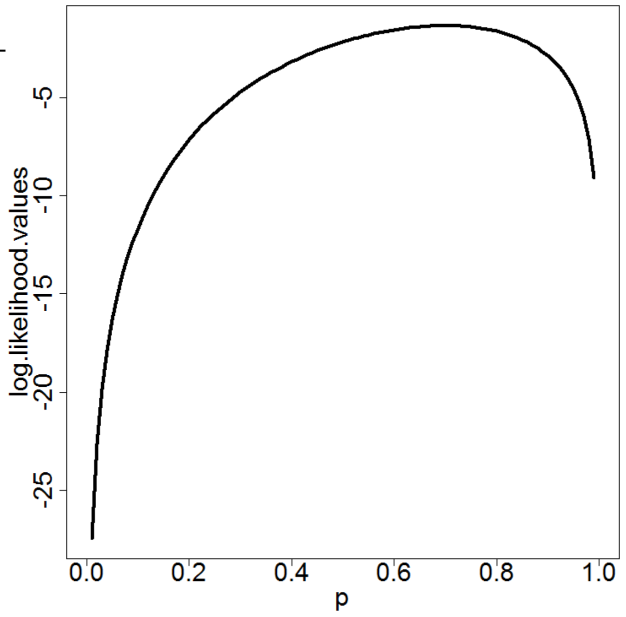
p	$L(p)$
0	$C_7^{10} 0^7 (1-0)^8 = 0$
0.1	$C_7^{10} 0.1^7 0.9^3 = 8.748 * 10^{-6}$
0.2	$C_7^{10} 0.2^7 0.8^3$ =0.000786
0.3	$C_7^{10} 0.3^7 0.7^3$ =0.0090
0.4	$C_7^{10} 0.4^7 0.6^3 = 0.0424$
⋮	⋮



Some R code

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

# LET US CALCULATE THE LIKELIHOOD VALUE AT p=0.1
binomial.likelihood(p=0.1)
        # YOU GOT SOMETHING AROUND 8.748e-06, RIGHT?

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

```
# MORE OFTEN WE STUDY THE LOG-LIKELIHOOD
# 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')
```



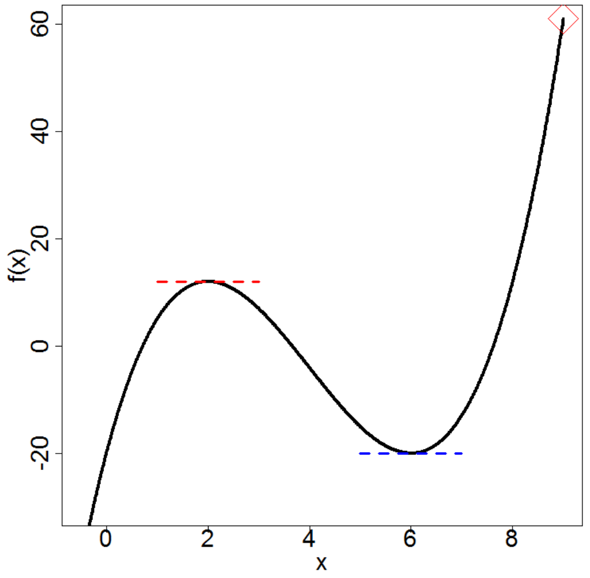
We can see that both the likelihood and log-likelihood function are maximised when p is around 0.7

- Remember in day 1 we made some probabilistic statements: If we toss 10 fair coins, independently, then the probability of getting 7 heads out of 10 tosses is around 0.117.

- Today we make some statistical inferences: If we observed 7 heads out of 10 tosses, what can we say about the coin? Is the coin loaded?

Maximisation: some mathematical considerations

- To optimise a function we need to calculate its derivatives. Other conditions (e.g. the boundaries or saddle points) need to be examined as well.
- Some proficiency in calculus is certainly required, and things can be complicated if we have multiple parameters (multivariate calculus).



- In many cases, because of the complexity of the model, or high dimensionality of the parameters (or both!), MLE cannot be solved explicitly.

- More often, log-likelihood functions are maximised numerically via computer
  - `optim()` or `optimize()` in R

```
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

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

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

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

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

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

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

$$\dots$$

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

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 μ, the population mean.
- Parameter of interest: μ
- $L(\mu) = f(x_1, x_2, \dots, x_n) = \prod_{i=1}^{n} f(x_i)$

Because of independence!

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

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

x_i are the observed samples, fixed.
μ is the only quantity to be estimated.

The log-likelihood is

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

Does not depend on μ

Differentiate the log-likelihood wr.t. μ

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

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

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

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

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

MLE suggests that the arithmetic average of our samples is an estimate for μ.

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 μ, σ^2 are unknown.
- Parameters of interest: μ, σ^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 $(\hat{\mu}, \widehat{\sigma^2})$ 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 ε_i is the subject, the model becomes $\varepsilon_i = y_i - a - bx_i$
- $L(\underline{\theta}) = f(\underline{\varepsilon}|\underline{\theta}) = f(\varepsilon_1|\underline{\theta})\, f(\varepsilon_2|\underline{\theta}) \cdots f(\varepsilon_n|\underline{\theta})$

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

$$= \prod_{i=1}^{n} f(y_i - a - bx_i|\underline{\theta})$$

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 σ^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 marked-recapture experiment

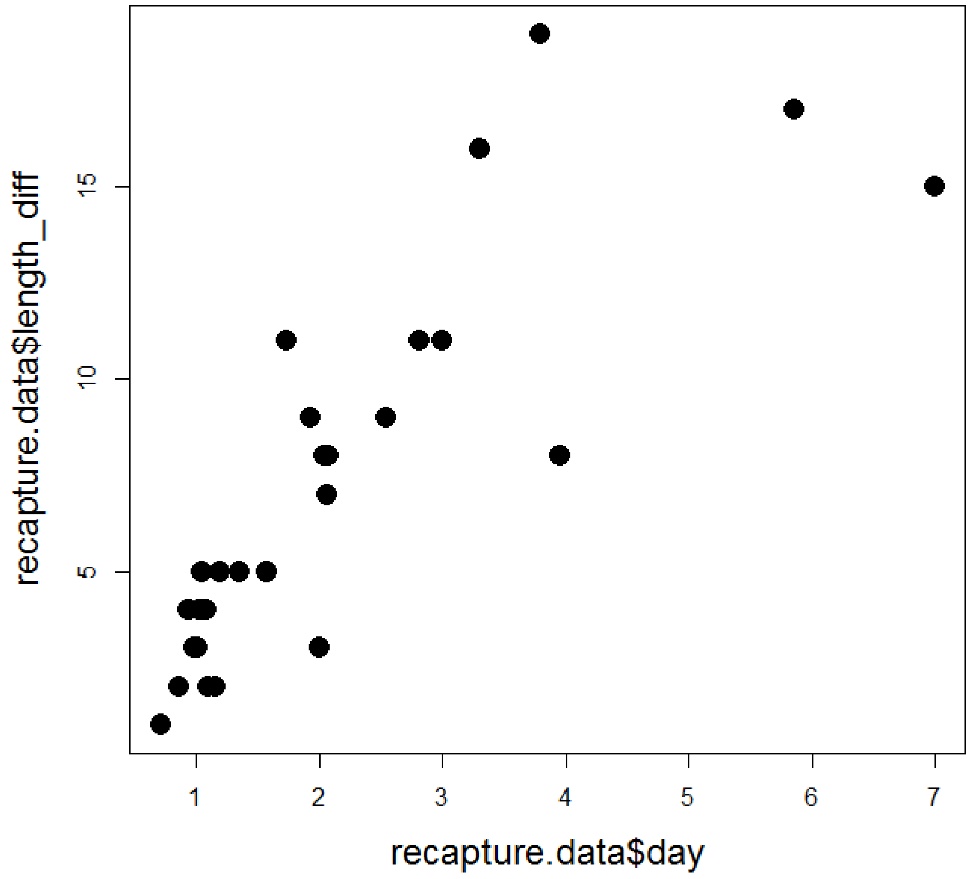
This afternoon

- A rabbit example
  - Get your hands dirty…

Rabbit example

- We had tagged and released some rabbits, and then some 29 of them were recaptured
- `day` measures the days a rabbit had spent before being recaptured (Explanatory variable)
- `diff_length` is the grow in body length in between (Response)
- What is the relationship between these two variables?

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

# SCATTERPLOT
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)
```

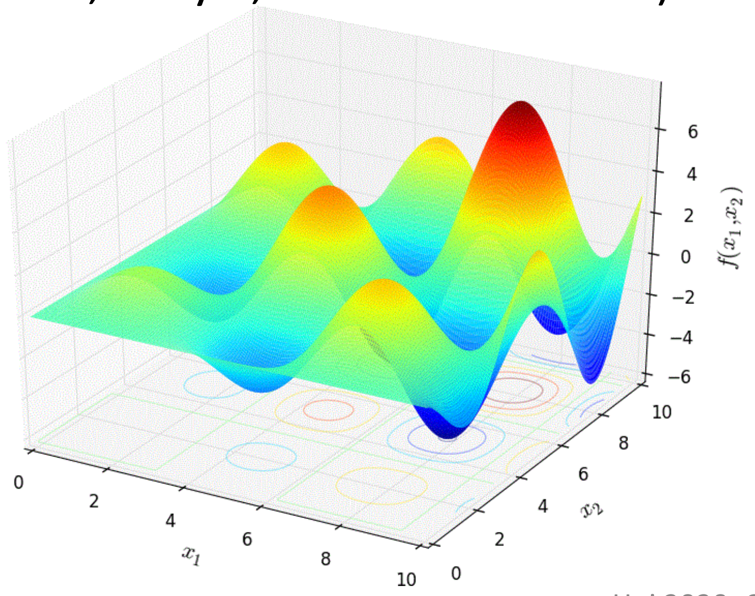
```
$par
[1] 1.527870 2.676240 2.678428

$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 to be optimised
`method='L-BFGS-B'`	Optimisation algorithm
`lower=c(-1000,-1000,0.0001)`	Lower bound of parameter space
`upper=c(1000,1000,10000)`	Upper bound of parameter space
`control=list((fnscale=-1))`	`fnscale=-1` means to maximise

Notes on using `optim()`

- Parameters are input as a vector. Order does matter.
- Initial parameter values is set by the first argument `par=`
- Choice of optimisation `method` can be tricky even for advanced users. See R help for details.
- The method `L-BFGS-B` requires a box-like `upper` and `lower` bound for parameter values. Nothing to specify for `Nelder-Mead`
- If you wish to maximise a function, put `fnscale=-1` in your `control` list. The default is to minimise. You can put multiple control parameters (such as tolerance) in the `control` list.
- The Hessian matrix provide information about the variance-covariance structure of your parameter estimates (more later).
- Try multiple sets of initial parameters to ensure they all converge to the same maximum.

- “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 finds his place.
- But he may get stuck at the local maximum (not the most comfortable place, but, still…, okay…, at a tube station?)



UNDERGROUND

Photo credit: Dan Reuman

photo credit: http://infinity77.net/global_optimization/test_functions_nd_A.html

- 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
```