Examples Programming in R

From C programming to R style

Goal
• two football teams with different averages
• how often does the better team win?
  • poisson processes
Ingredients / Plan

Some ingredients / the plan

• poisson processes / exponential distribution / time between events
• how to code it
  • naïve – time-based simulation
  • changing distributions
  • C programmer version
• using R features
Taylor expansion of $\ln x$

Will need (soon)
\[ \lim_{n \to \infty} \left(1 + \frac{T}{N}\right)^N \]

First I want to know about $\ln(x + 1)$
Remember Taylor expansion for some $a$
\[ f(x) = f(a) + f'(a)(x - a) + \frac{f''(a)}{2!}(x - a)^2 + \frac{f'''(a)}{3!}(x - a)^3 + \ldots \]
for logarithms
\[ \ln(x) = \ln a + \frac{x - a}{a} - \frac{(x - a)^2}{2a^2} + \frac{(x - a)^3}{3a^3} + \ldots \]

why? do not forget $\frac{d}{dx} \ln x = \frac{1}{x}$
from previous slide

\[ \ln(x) = \ln a + \frac{x-a}{a} - \frac{(x-a)^2}{2a^2} + \frac{(x-a)^3}{3a^3} + \ldots \]

so

\[ \ln(x + 1) = \ln(a) + \frac{x + 1 - a}{a} - \frac{(x + 1 - a)^2}{2a^2} + \frac{(x + 1 - a)^3}{3a^3} - \ldots \]

let me set \( a = 1 \)

\[ \ln(x + 1) = \ln(1) + x - \frac{x^2}{2} + \frac{x^3}{3} - \ldots = 0 + x - \frac{x^2}{2} + \frac{x^3}{3} - \ldots \]

what happens as \( x \to 0 \) ?

\[ \lim_{x \to 0} (\ln(x + 1)) = x \]

will need this later
Uniformly distributed events

Decay of a particle / chemistry

- $A \rightarrow B + C$ long term average is clear $A(t) = A_0 e^{-\lambda t}$
- intuitive
  - in some time $\Delta T$ I can talk about the probability of a breakdown
  - if the decay rate $\lambda$ is high, the probability is higher
  - say time between breakdown is $\tau = \lambda^{-1}$
- we rarely look at individual molecules ($\Delta T \gg \tau$)
- when do we see individual events?
  - football game (and Geiger counters, ion channels)
Non-Uniformly distributed events

- Football – long term average is clear (1300 goals in 1000 games)
- short term? very uncertain – no goals, 5 goals are possible
- order of magnitude..
  - \( \tau = \frac{T}{N} = \frac{90}{2} = 45 \text{ min} \) (for about two goals scored)
- other systems in biology / chemistry?
  - ion channels in nerves open / close spontaneously (rare, but easy to measure)
  - few copies of DNA repressor per cell
    - DNA + protein \( \rightarrow \) (DNA-protein) rare event – hard to see
    - classical chemical kinetics is not helpful
Distribution for these events

- Start from average over long $T$
- divide into $N \times \Delta T$
- get limit as $N \to \infty$ and $\Delta T \to 0$

my nomenclature
- rate $\lambda = \frac{1}{\tau}$ the average time between goals / channel opening /..

- what is the average number of goals in $\Delta T$? $P(\Delta T) = \lambda \Delta T$
- and probability of no goal in some $\Delta T$ is $P_0(\Delta T) = (1 - \lambda \Delta T)$
longer time with many $\Delta T$

\[ P_0(T) \approx (1 - \lambda \Delta T)^N \quad \text{or} \quad \left( 1 - \frac{\lambda T}{N} \right)^N \]

Result from earlier ... \[ \lim_{x \to 0} \ln(1 + x) = x \]

\[ P_0(T) \approx \left( 1 - \frac{\lambda T}{N} \right)^N \]

\[ = \lim_{N \to \infty} \left( 1 - \frac{\lambda T}{N} \right)^N \]

\[ = \exp \left( \log \lim_{N \to \infty} \left( 1 - \frac{\lambda T}{N}\right)^N \right) = \exp \left( N \log \lim_{N \to \infty} \left( 1 - \frac{\lambda T}{N}\right) \right) \]

\[ = \exp \left( N \frac{-\lambda T}{N} \right) \]

\[ = e^{-\lambda T} \]
The exponential distribution

- probability for no goal $P_0(T) = e^{-\lambda T}$
- check intuition
- what I want is the probability of 1 goal, 2 goals, ... within time $t$

\[I_1 dt = \text{probability of no goal in } t \times \text{probability of goal in } dt\]

\[I_1 dt = P_0(t) \lambda dt = \lambda e^{-\lambda T} dt\]

$I_1 = \lambda e^{-\lambda T}$

exponential distribution
One possibility – use exponential distribution

- naïve inefficient simulation

we have rates $\lambda_1$ and $\lambda_2$, work out total $\lambda_0 = \lambda_1 + \lambda_2$

set up counters $n_1$ and $n_2$

set up $tmp_1$ and $tmp_2$

while ($t < T_{\text{game}}$){
  $n_1 += tmp_1;$  $n_2 += tmp_2;$
  $tmp_1 = tmp_2 = 0$
  $\Delta t = \text{random\_from\_exponential}(\lambda_0)$
  decide who gets goal (random based on $\frac{\lambda_1}{\lambda_1+\lambda_2}$)
  increment $tmp_1$ or $tmp_2$
}

- we can be much more efficient
expected number of goals in $t$

- start with binomial distribution
- probability of success in one try is $p$
- I have $n$ tries
- what is the probability of seeing $k$ successes
  \[ P(k|n, p) = \binom{n}{k} p^k (1 - p)^{n-k} \]
- you remember $\binom{n}{k} = \frac{n!}{k!(n-k)!}$
- When do you see this?
  Probability of seeing $k = 5$ heads from $n = 10$ coin tosses with $p = \frac{1}{2}$
• rate per unit time game $\lambda$
• some rules
  • events (goals) are independent
  • events are rare – probability of one in short time $t$ is $\lambda t$
  • you never see two events in a very short time

• take unit time and divide by $n$ (trials)
• probability in one of $n$ units is $p = \lambda/n$
• we are interesting in case of very small $p$ in any one $\delta t$
original name binomial $P(k|n,p) = \frac{n!}{k!(n-k)!} p^k (1-p)^{n-k}$

write as $P(X = x) = \frac{n!}{x!(n-x)!} p^x (1-p)^{n-x}$  remember $p = \frac{\lambda}{n}$

consider limit

$$\lim_{n \to \infty} \frac{n!}{x!(n-x)!} \left( \frac{\lambda}{n} \right)^x \left( 1 - \frac{\lambda}{n} \right)^{n-x} = \lim_{n \to \infty} \frac{n(n-1)\cdots(n-x+1) \lambda^x}{n^x} \frac{\lambda^x}{x!} \left( 1 - \frac{\lambda}{n} \right)^n \left( 1 - \frac{\lambda}{n} \right)^{-x}$$
from binomial to poisson

$$\lim_{n \to \infty} \frac{n!}{x! (n-x)!} \left( \frac{\lambda}{n} \right)^x \left( 1 - \frac{\lambda}{n} \right)^{n-x} = \lim_{n \to \infty} \frac{n(n-1) \cdots (n-x+1) \lambda^x}{n^x} \frac{(1 - \frac{\lambda}{n})^n}{x!} \left( 1 - \frac{\lambda}{n} \right)^{-x}$$

$$\lim_{n \to \infty} \frac{n(n-1) \cdots (n-x+1)}{n^x} = \lim_{n \to \infty} \left[ \frac{n}{n} \left( 1 - \frac{1}{n} \right) \cdots \left( 1 - \frac{x-1}{n} \right) \right] = 1$$

$$\lim_{n \to \infty} \left( 1 - \frac{\lambda}{n} \right)^n = e^{-\lambda} \text{ and } \lim_{n \to \infty} \left( 1 - \frac{\lambda}{n} \right)^{-x} = 1$$

$$\lim_{n \to \infty} \frac{n!}{x! (n-x)!} \left( \frac{\lambda}{n} \right)^x \left( 1 - \frac{\lambda}{n} \right)^{n-x} = \frac{\lambda^x e^{-\lambda}}{x!} = P(X = x)$$
Poisson distribution

\[ P(X = x) = \frac{\lambda^x e^{-\lambda}}{x!} \]

- from average rate of events \( \lambda \) I can calculate the probability of seeing some number \( x \) events
- change simulation strategy

\[ P(n) \]

\( \lambda = 2 \)
simulation strategy

• look up rate of goals for team 1 ($\lambda_1$) and 2 ($\lambda_2$)
• say $\text{Pois}(\lambda)$ is a random number drawn from poisson distribution
• a game is
  
  $n_1 = \text{Pois}(\lambda_1) \text{ and } n_2 = \text{Pois}(\lambda_2)$

  if ($n_1 > n_2$) team 1 wins
  elseif ($n_2 > n_1$) team 2 wins
  else draw

• repeat many times to get probabilities

• first approach C style
game <- function (mu_1, mu_2) {
  team1_result <- rpois(1, mu_1)
  team2_result <- rpois(1, mu_2)

  if (team1_result > team2_result) {
    result <- 1
  } else if (team2_result > team1_result) {
    result <- 2
  } else {
    result <- 0
  }

  return (result)
}

to run the game.
result <- c()
for (i in 1:n_games) {
    result <- c(result, game(team1_mu, team2_mu))
}
w1 = length(result[result==1]); w2 = length(result[result==2])
draw = n_games - (w1 + w2)
cat ("team 1", w1, w1/n_games * 100, 
    
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
"
• games are independent events
• use vectors in R

```
team1_mu <- 1.  # Average number goals per match
team2_mu <- 1.3
n_games <- 100000  # How many games to play

team1 <- rpois(n_games, team1_mu)  # generate 100000 results in one go
team2 <- rpois(n_games, team2_mu)  # team1/2 are long vectors

w1 <- sum (team1 > team2)  # sum over logicals
w2 <- sum (team2 > team1)  # team2 > team1 is a long logical vector
draws <- n_games - (w1 + w2)
```
• from 10 ½ s to 0.13 s (including printing results) let us plot results
• build up text, put in "a", define colour1, colour2, setup breaks
  xlim=c(0,9)
  hist (team1, breaks=breaks, probability=T, main="", col=colour1,xlab= "n goals", ylab = "frequency", xlim=xlim)
  hist (team2, breaks=breaks, probability=T, main="", col=colour2, add=T)
  legend(x=3, y=0.25, legend=c("team1", "team2"), col=c(colour1, colour2), lwd=5, box.lty=0)
  text(a, x=4, y=0.3, adj=0)

• can make the plot clearer
  • box and whiskers
• remember the scores are in team1 and team2
• add a command for two rows and 

```
boxplot (df, horizontal=T, ylim=c(0,8), col=c(colour1, colour2))
```

• line shows median
• half box is 25 %
• dots for outliers

• histogram is not so clear
• flip limits on second histogram – no editing of data

• some tedious placement of plots
can we make football better?

- instead of one time unit $\mu = \lambda$ make games $n$ times longer so 
  $$\mu = \lambda n$$

- put 100 000 games into a function

```r
onерound <- function (mu1, mu2, n_games) {
  team1 <- rpois(n_games, mu1)
  team2 <- rpois(n_games, mu2)
  w1 <- sum (team1 > team2)
  w2 <- sum (team2 > team1)
  draw <- n_games - (w1 + w2)
  return (c(w1, w2, draw))
}
```

- and call this for different values of $\mu_1, \mu_2$ scaled by $n$
more compact

How often does the better team win?

```r
oneround <- function (mu1, mu2, n_games) {
  team1 <- rpois(n_games, mu1)
  team2 <- rpois(n_games, mu2)
  w2 <- sum (team2 > team1)
}
```

play repeatedly

```r
mult <- seq(from = 1, to = 100, by = 1)
wins <- c()
for (m in mult) {
  r <- oneround (team1_mu * m, team2_mu * m, n_games)
  wins = c(wins, r)
}
```

• plot the results
• if game are 10 times longer better team wins 62 %

• if football games are about 60 fold longer, results are interesting

\[ \mu_1 = 1 \]
\[ \mu_2 = 1.3 \]
what has one seen?

Did I cheat in scripts? not much
• some code for placing plots on page, setting random seed, histogram breaks

Football results are close to meaningless

R programming
• ugly but very powerful – basic poisson competition less than 10 lines
• graphics easy, but syntax horrible
• use built-in functions
• work with vectors not scalars
more serious R

Real statistician
- would have looked up poisson race

R – these lectures too short
- much serious statistics
- interesting fitting in the Übung (general non-linear, arbitrary function)
- most R users would
  - work in rcmdr, rstudio, ..
  - used a higher level graphics library (ggplot2, lattice)
Fitting

You are given

\[
\begin{align*}
  x & \quad y \\
  0.09991595 & \quad 0.031080097 \\
  0.09982738 & \quad -0.012845276 \\
  0.09946064 & \quad 0.026970036
\end{align*}
\]

[ .. 500 .. lines]

and asked to make sense of it.

Start with a plot
Hint that it is of form
\[ y = e^{-\beta x} \sin(2\pi \omega x + \varphi) \]
for some
\( \beta \) decay rate
\( \omega \) frequency
\( \varphi \) phase

- there is noise
- points are not evenly spaced

\[
d <- \text{read.table}(\text{datafile}, \text{header=}\text{TRUE})
\]
\[
\text{plot}(d$x, d$y, xlab = "x", ylab="amplitude")
\]
Code up the likely function

\[
\text{sinexp} \leftarrow \text{function} \left( x, \text{phase}, \text{freq}, \text{decay} \right) \{ \\
\hspace{1em} v \leftarrow \sin(2 \cdot \pi \cdot x \cdot \text{freq} + \text{phase}) \\
\hspace{1em} v \leftarrow v \times \exp(-\text{decay} \times x) \\
\}
\]

- this function acts on a vector \( (x) \)
- returns a vector \( (v) \)

Have we got the right form?
Check if our function looks sensible

curve (\texttt{sinexp(x, 1, 100, 30)}, \texttt{from = 0, to = 0.1})
• \texttt{sinexp()} seems to be possible

Note
• \texttt{curve()} has a default name of \texttt{x}
• it takes guessing / experience to get sensible values
• these values can also be used as starting points for fitting
non-linear least-squares fitting

- `?nls`
- gives you about 330 lines of help `?nls.control` gives more
- what works here? defaults including Gauss-Newton method

We are asking R to move around in 3-parameter space, $\beta, \omega, \varphi$

```r
nlmod <- nls(y~sinexp(x, phase, freq, decay), data = d,
             start = list(phase=3, freq=150, decay=30))
```

You may not have seen the `~` in R – used to define models
Results are stored in nlmod

> nlmod

Nonlinear regression model

  model: y ~ sinexp(x, phase, freq, decay)
  data: d

  phase   freq   decay
  3.801  108.846  30.922

residual sum-of-squares: 0.342

Number of iterations to convergence: 10
Achieved convergence tolerance: 2.912e-06

accessing elements is a bore, but you can say coef(nlmod)[phase]

> coef(nlmod)

  note coef() is a function call

  see why coef(nlmod)[phase] works?

phase   freq   decay
  3.8   108.8   30.9
to see if you really reproduce data

use the predict() function

plot (d$x, d$y, xlab = expression (italic(x)), ylab="amplitude")

pred = predict (nlmod)

lines(d$x, predict(nlmod), col = 3, lwd = 3)
Summary

• R syntax is as ugly as last week
• numerical functions remarkably
  • concise
  • simple