Norges teknisk-naturvitenskapelige universitet
Institutt for matematiske fag

Bokmål

Faglig kontakt under eksamen: Førsteamanuensis Jarle Tufto
Telefon: 99 70 55 19

Bioberegninger, ST1301

Mandag 22. mai, 2006
Kl. 15-19
Hjelpemidler: Alle trykte og skrevne hjelpemidler, lommeregner.
Sensur: 12. juni 2006

Hjelpesider for noen R-funksjoner som er omhandlet nedenfor eller som du vil kunne få bruk for i programmeringsoppgavene følger på side 4.

Oppgave 1

a) Anta at vi definerer følgende vektor i R:

```
a <- c("x","y","z")
```

Hva blir da verdien av følgende uttrykk?

```
a[3]
a[-3]
a[c(2,3)]
a[c(3,2)]
a[c(1,2,2,1,3,3,3)]
```

b) Anta at vi definerer følgende funksjon i R.

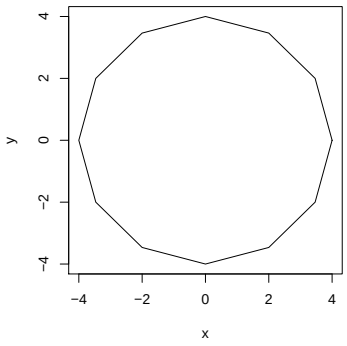
```
funk <- function(x) {
  n <- length(x)
  y <- rep(0,n)
  y[1] <- x[1]
  for (i in 2:n) {
     y[i] <- y[i-1]+x[i]
  }
```

```
  y
}
```

Hva vil funksjonen da returnere dersom vi gjør følgende funksjonskall?

```
funk(c(3,2,5,1,3))
```

Oppgave 2 Hvilke kommandoer vil du bruke for å lage følgende figur i R?



Oppgave 3 Hos mange organismer er hunnene mer langtlevende enn hannene. Anta at levetidene til hanner og hunner er eksponentiellt fordelte med parametere lik henholdsvis λ_1 og λ_2.

a) La X være levetiden til en tilfeldig valgt hann og Y levetiden til en tilfeldig valgt hunn. Programmer en funksjon som ved å simulere mange realisasjoner av X og Y beregner et estimat av $P(X > Y)$, altså sannsynligheten for at en tilfeldig valgt hann lever lenger enn en tilfeldig valgt hunn gitt λ_1 og λ_2. Kan du lage et eksempel på inndata hvor de utdata funksjonen skal returnere er kjent?

b) Anta at vi observerer levetidene til 20 hanner og 20 hunner og at vi utfører en to-utvalgs t-test av nullhypotesen $\lambda_1 = \lambda_2$. Hvilken forutsetning bygger t-testen på som i tilfelle ikke ville vært oppfyllt? Diskuter hvorvidt det likevel ville kunne være fornuftig å bruke en slik t-test. Skisser med ord (uten å skrive noe program) hvordan du ville undersøkt hvorvidt t-testen oppfører seg slik den skal også for eksponentiellt fordelte data.

Oppgave 4 Hos en territoriell art vokser populasjonsstørrelsen geometrisk begrenset oppad av en viss bæreevne K. Populasjonsstørrelsen i år $t+1$ gitt populasjonsstørrelsen i år t er med andre ord gitt ved

$$N_{t+1} = \min(RN_t, K) \tag{1}$$

a) Programmer en funksjon som beregner $N_2, N_3, \ldots, N_n$ gitt N_1, R, K, og n.

b) Med hvor mange prosent øker populasjonsstørrelsen hvert år dersom $R = 1.05$ og N_t ligger langt under bæreevnen K? Skisser hvordan N_t vil se ut som funksjon av t gitt $K = 1000$, $N_1 = 500$, og $R = 1.11$.

Anta at bæreevnen i stedet for å være en konstant varierer fra år til år og at bæreevnen K_t i ulike år er uavhengig normalfordelt med forventning μ og standardavvik σ, slik at

$$N_{t+1} = \min(RN_t, K_t), \quad K_t \sim N(\mu, \sigma^2). \tag{2}$$

c) Endre på funksjonen fra punkt (a) slik at den simulerer en realisasjon av populasjonsstørrelsene $N_2, N_3, \ldots, N_n$ gitt N_1, R, μ, σ, og n.

d) Skisser hvordan en realisasjon av N_t nå vil kunne se ut som funksjon av t gitt $\mu = 1000$, $\sigma = 200$, $N_1 = 500$, og $R = 1.05$.

```
Exponential             package:stats             R Documentation

The Exponential Distribution

Description:

     Density, distribution function, quantile function and random
     generation for the exponential distribution with rate 'rate'
     (i.e., mean '1/rate').

Usage:

     dexp(x, rate = 1, log = FALSE)
     pexp(q, rate = 1, lower.tail = TRUE, log.p = FALSE)
     qexp(p, rate = 1, lower.tail = TRUE, log.p = FALSE)
     rexp(n, rate = 1)

Arguments:

    x, q: vector of quantiles.

       p: vector of probabilities.

       n: number of observations. If 'length(n) > 1', the length is
          taken to be the number required.

    rate: vector of rates.

log, log.p: logical; if TRUE, probabilities p are given as log(p).

lower.tail: logical; if TRUE (default), probabilities are P[X <= x],
          otherwise, P[X > x].

Details:

     If 'rate' is not specified, it assumes the default value of '1'.

     The exponential distribution with rate lambda has density

                     f(x) = lambda e^(- lambda x)

     for x >= 0.

Value:

     'dexp' gives the density, 'pexp' gives the distribution function,
     'qexp' gives the quantile function, and 'rexp' generates random
     deviates.

Note:

     The cumulative hazard H(t) = - log(1 - F(t)) is '-pexp(t, r, lower
     = FALSE, log = TRUE)'.

------------------------------------------------------------------------

plot                package:graphics                R Documentation

Generic X-Y Plotting

Description:

     Generic function for plotting of R objects.  For more details
     about the graphical parameter arguments, see 'par'.

Usage:

     plot(x, y, ...)

Arguments:

       x: the coordinates of points in the plot. Alternatively, a
          single plotting structure, function or _any R object with a
          'plot' method_ can be provided.

       y: the y coordinates of points in the plot, _optional_ if 'x' is
          an appropriate structure.

     ...: graphical parameters can be given as arguments to 'plot'.
          Many methods will also accept the following arguments:

     'type' what type of plot should be drawn.  Possible types are

               *  '"p"' for *p*oints,

               *  '"l"' for *l*ines,

               *  '"b"' for *b*oth,

               *  '"c"' for the lines part alone of '"b"',

               *  '"o"' for both "*o*verplotted",

               *  '"h"' for "*h*istogram" like (or "high-density")
                  vertical lines,

               *  '"s"' for stair *s*teps,

               *  '"S"' for other *s*teps, see _Details_ below,

               *  '"n"' for no plotting.

          All other 'type's give a warning or an error; using, e.g.,
          'type = "punkte"' being equivalent to 'type = "p"' for S
          compatibility.

     'main' an overall title for the plot: see 'title'.

     'sub' a sub title for the plot: see 'title'.

     'xlab' a title for the x axis: see 'title'.

     'ylab' a title for the y axis: see 'title'.

Details:

     For simple scatter plots, 'plot.default' will be used. However,
     there are 'plot' methods for many R objects, including
     'function's, 'data.frame's, 'density' objects, etc.  Use
     'methods(plot)' and the documentation for these.

     The two step types differ in their x-y preference: Going from
     (x1,y1) to (x2,y2) with x1 < x2, 'type = "s"' moves first
     horizontal, then vertical, whereas 'type = "S"' moves the other
     way around.

See Also:

     'plot.default', 'plot.formula' and other methods; 'points',
     'lines', 'par'.

Examples:

     plot(cars)
     lines(lowess(cars))

     plot(sin, -pi, 2*pi)

     ## Discrete Distribution Plot:
     plot(table(rpois(100,5)), type = "h", col = "red", lwd=10,
          main="rpois(100,lambda=5)")

     ## Simple quantiles/ECDF, see ecdf() {library(stats)} for a better one:
     plot(x <- sort(rnorm(47)), type = "s", main = "plot(x, type = \"s\")")
     points(x, cex = .5, col = "dark red")

------------------------------------------------------------------------
seq                     package:base                     R Documentation

Sequence Generation

Description:

     Generate regular sequences.

Usage:

     from:to
        a:b

     seq(from, to)
     seq(from, to, by= )
     seq(from, to, length.out= )
     seq(along.with= )
     seq(from)
```

```
Arguments:

    from: starting value of sequence.

      to: (maximal) end value of the sequence.

      by: increment of the sequence.

length.out: desired length of the sequence.

along.with: take the length from the length of this argument.

     a,b: 'factor's of same length.

Details:

     The binary operator ':' has two meanings: for factors 'a:b' is
     equivalent to 'interaction(a, b)' (except for labelling by 'la:lb'
     not 'la.lb').  For numeric arguments 'a:b' is equivalent to
     'seq(from=a, to=b)'.

     The interpretation of the unnamed arguments of 'seq' is _not_
     standard, and it is recommended always to name the arguments when
     programming.

     Function 'seq' is generic, and only the default method is
     described here.

     The operator ':' and the 'seq(from, to)' form generate the
     sequence 'from, from+1, ..., to'.

     The second form generates 'from, from+by', ..., up to the sequence
     value less than or equal to 'to'.

     The third generates a sequence of 'length.out' equally spaced
     values from 'from' to 'to'.

     The fourth form generates the sequence '1, 2, ...,
     length(along.with)'.

     The last generates the sequence '1, 2, ..., length(from)' (as if
     argument 'along' had been specified), _unless_ the argument is
     numeric of length 1 when it is interpreted as '1:from' (even for
     'seq(0)' for compatibility with S).

     If 'from' and 'to' are factors of the same length, then 'from :
     to' returns the "cross" of the two.

     Very small sequences (with 'from - to' of the order of 10^{-14}
     times the larger of the ends) will return 'from'.

Value:

     Currently, the default method returns a result of _storage mode_
     '"integer"' if 'from' is (numerically equal to an) integer and,
     e.g., only 'to' is specified, or also if only 'length' or only
     'along.with' is specified.  *Note:* this may change in the future
     and programmers should not rely on it.


--------------------------------------------------------------------------
Extremes                package:base                R Documentation

Maxima and Minima

Description:

     Returns the (parallel) maxima and minima of the input values.

Usage:

     max(..., na.rm=FALSE)
     min(..., na.rm=FALSE)

Arguments:

     ...: numeric arguments.

   na.rm: a logical indicating whether missing values should be
          removed.

Value:

'max' and 'min' return the maximum or minimum of _all_ the  values
present in their arguments, as 'integer' if all are 'integer',  or
as 'double' otherwise.

The minimum and maximum of an empty set are '+Inf' and '-Inf' (in
this order!) which ensures _transitivity_, e.g., 'min(x1, min(x2))
== min(x1,x2)'. In R versions before 1.5, 'min(integer(0)) ==
.Machine$integer.max', and analogously for 'max', preserving
argument _type_, whereas from R version 1.5.0, 'max(x) == -Inf'
and 'min(x) == +Inf' whenever 'length(x) == 0' (after removing
missing values if requested).

If 'na.rm' is 'FALSE' an 'NA' value in any of the arguments will
cause a value of 'NA' to be returned, otherwise 'NA' values are
ignored.

'max' and 'min' are generic functions: methods can be defined for
them individually or via the 'Summary' group generic.
```