Norwegian University
of Science and Technology
Department of Mathematical Sciences

English

Contact during the exam: Professor Jarle Tufto
Phone: 99 70 55 19

Statistical modelling for biologists and biotechnologists, ST2304

9. juni, 2011
Kl. 9–13
Grades to be announced: 30. juni, 2011

Permitted aids: One handwritten yellow A4 paper, pocket calculator, “Tabeller og formler i statistikk” (Tapir forlag), K. Rottmann: Matematisk formelsamling.

Help pages for some R functions you may need to use follows on page 7.

Problem 1 Suppose that the number of individuals of a given plant species inside rectangles of size A is Poisson distributed with expected value λA where $\lambda = 0.5$ per square meter.

a) Write an R expression that computes the probability that there is exactly 5 individuals inside a rectangle with an area of 10 square meters.

Suppose that we consider 5 rectangles and that we create a vector `A` in R which represents the area (in square meters) of these rectangles as follows.

```
A <- c(10,15,20,25,30)
```

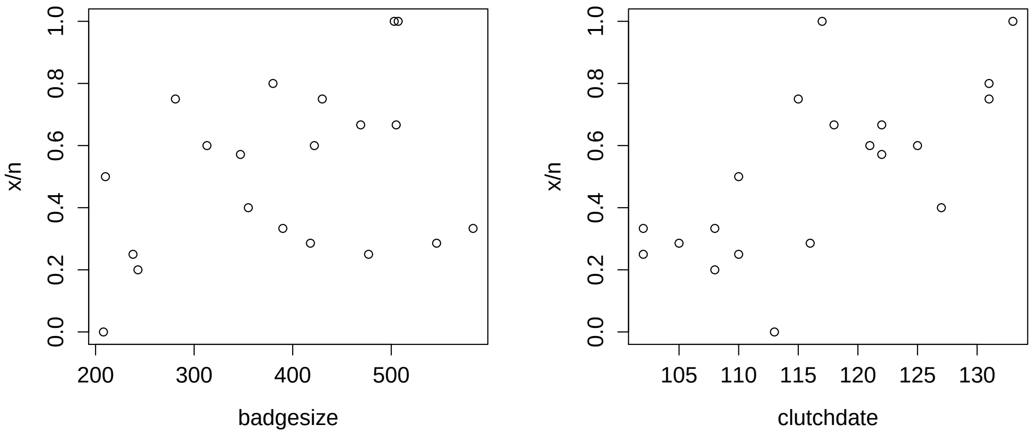
b) Write an R expression that, for each of the rectangles, computes the probability that the number of individual inside the rectangle is greater than 5.

c) Write an R expression that simulates the number of individuals inside each of the 5 rectangles.

Problem 2 Suppose that we study house sparrow nestling survival in 20 nests and observe the following data where `x` is the number of surviving nestlings after 12 days, `n` is the number of eggs laid in each nest, `badgesize` is the size of the badge of the male parent (mm^2), and clutchdate is the date at which the last egg in the clutch was laid after january the 1st.

```
> sparrows
   x n badgesize clutchdate
1  1 3       583        108
2  1 4       477        102
3  3 3       507        133
4  1 3       390        102
5  3 5       313        121
6  2 7       546        116
7  3 5       422        125
8  2 3       505        122
9  2 4       210        110
10 4 5       380        131
11 2 5       355        127
12 2 3       469        118
13 3 4       281        115
14 1 5       243        108
15 3 4       430        131
16 4 7       347        122
17 4 4       503        117
18 1 4       238        110
19 0 3       208        113
20 2 7       418        105
```

Plots of proportion of surviving nestlings versus badgesize and clutchsize are shown below.



a) We first fit a linear model using the proportion of surviving nestlings as the response variable and clutchdate as the only explanatory variable after having omitted badgesize from the model

```
> prop <- x/n
> summary(lm(prop ~ clutchdate))

Call:
lm(formula = prop ~ clutchdate)

Residuals:
     Min       1Q   Median       3Q      Max
-0.43890 -0.08541  0.00396  0.10957  0.48400

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.73908    0.57646  -3.017  0.00741 **
clutchdate   0.01927    0.00492   3.918  0.00101 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.206 on 18 degrees of freedom
Multiple R-squared: 0.4603,Adjusted R-squared: 0.4303
F-statistic: 15.35 on 1 and 18 DF,  p-value: 0.001009
```

Based on this model, what is the predicted proportion of surviving nestlings if the clutch has a clutchdate of 150? Is this precited value reasonable? Are any other assumptions of the model unreasonable? Would you trust the finding that clutchdate has a significant effect on nestling survival based on the above model?

Suppose that we instead fit a generalized linear model as follows.

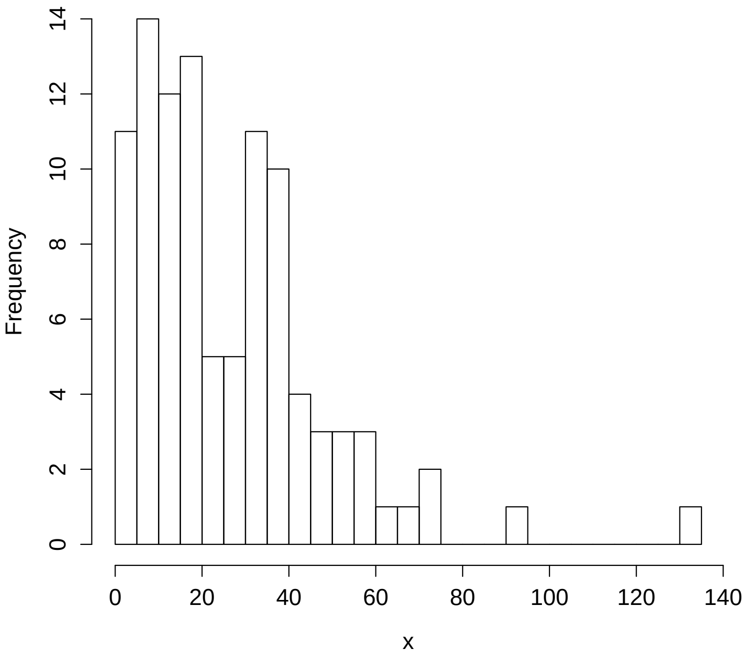
```
> summary(glm(prop ~ clutchdate+badgesize,weight=n,family=binomial(link=logit)))

Call:
glm(formula = prop ~ clutchdate + badgesize, family = binomial,
    weights = n)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.62259  -0.28653  -0.04847   0.37706   2.19469

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -9.730598   3.116205  -3.123  0.00179 **
clutchdate   0.077864   0.026339   2.956  0.00311 **
badgesize    0.001614   0.002110   0.765  0.44426
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 26.667  on 19  degrees of freedom
Residual deviance: 15.571  on 17  degrees of freedom
AIC: 53.893

Number of Fisher Scoring iterations: 4
```

b) Write down the assumptions of this model and an equation representing the model in mathematical notation. Explain why this model may be more reasonable than the linear model considered previously.

c) What is the predicted nestling survival in a clutch with a clutchdate equal to 150 and with a male parent with a badgesize of 400mm^2?

d) Are there any signs of over- or under-dispersion in the data? Discuss briefly possible mechanisms which may lead to over- and under-dispersion in this situation.

Problem 3 Suppose that we observe the lifespan X (in years) of 100 different pine trees. A histogram of the observed lifespan (contained in the vector `x`) is shown below.



We assume that these lifespans follow a gamma distribution with density function

$$f(x) = \frac{1}{\beta^{\alpha}\Gamma(\alpha)} x^{\alpha-1} e^{-x/\beta} \tag{1}$$

a) Write a function `lnL` which takes two arguments; a vector containing the parameters α and β and second vector containing the observations. The function should then returns the negative log likelihood.

Suppose that we then minimise the negative log likelihood function as follows in R.

```
> fit <- optim(c(1,1),lnL,x=x,hessian=TRUE)
> fit
$par
[1]  1.270927 20.400906

$value
```

```
[1] 423.9039

$counts
function gradient
      73       NA

$convergence
[1] 0

$message
NULL

$hessian
           [,1]      [,2]
[1,] 117.014163 4.9017432
[2,]   4.901743 0.3055501

> solve(fit$hessian)
            [,1]       [,2]
[1,]  0.02605615 -0.4180021
[2,] -0.41800208  9.9785242
```

b) What parameter values which maximises the likelihood? What is the maximum log likelihood? Find approximate standard errors of the estimates.

Suppose that we also fit the simpler exponential model to the data and that this gives an observed maximum log likelihood of-425.56.

c) Explain why the exponential model is nested within the gamma model. Relying on an asymptotic approximation, can you reject the simpler exponential model in favour of the gamma model?

```
Poisson                 package:stats                  R Documentation

The Poisson Distribution

Description:

     Density, distribution function, quantile function and random
     generation for the Poisson distribution with parameter ‘lambda’.

Usage:

     dpois(x, lambda, log = FALSE)
     ppois(q, lambda, lower.tail = TRUE, log.p = FALSE)
     qpois(p, lambda, lower.tail = TRUE, log.p = FALSE)
     rpois(n, lambda)

Arguments:

       x: vector of (non-negative integer) quantiles.

       q: vector of quantiles.

       p: vector of probabilities.

       n: number of random values to return.

  lambda: vector of (non-negative) means.

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:

     The Poisson distribution has density

                      p(x) = lambda^x exp(-lambda)/x!

     for x = 0, 1, 2, ... .  The mean and variance are E(X) = Var(X) =
     lambda.

     If an element of ‘x’ is not integer, the result of ‘dpois’ is
     zero, with a warning.  p(x) is computed using Loader's algorithm,
     see the reference in ‘dbinom’.

     The quantile is right continuous: ‘qpois(p, lambda)’ is the
     smallest integer x such that P(X <= x) >= p.

     Setting ‘lower.tail = FALSE’ allows to get much more precise
     results when the default, ‘lower.tail = TRUE’ would return 1, see
     the example below.

Value:

     ‘dpois’ gives the (log) density, ‘ppois’ gives the (log)
     distribution function, ‘qpois’ gives the quantile function, and
     ‘rpois’ generates random deviates.

     Invalid ‘lambda’ will result in return value ‘NaN’, with a
     warning.

Source:

     ‘dpois’ uses C code contributed by Catherine Loader (see
     ‘dbinom’).

     ‘ppois’ uses ‘pgamma’.

     ‘qpois’ uses the Cornish-Fisher Expansion to include a skewness
     correction to a normal approximation, followed by a search.

     ‘rpois’ uses

     Ahrens, J. H. and Dieter, U. (1982).  Computer generation of
     Poisson deviates from modified normal distributions.  _ACM
     Transactions on Mathematical Software_, *8*, 163-179.

See Also:

     Distributions for other standard distributions, including ‘dbinom’
     for the binomial and ‘dnbinom’ for the negative binomial
     distribution.

     ‘poisson.test’.

Examples:

     require(graphics)

     -log(dpois(0:7, lambda=1) * gamma(1+ 0:7)) # == 1
     Ni <- rpois(50, lambda = 4); table(factor(Ni, 0:max(Ni)))

     1 - ppois(10*(15:25), lambda=100)  # becomes 0 (cancellation)
         ppois(10*(15:25), lambda=100, lower.tail=FALSE)  # no cancellation

     par(mfrow = c(2, 1))
     x <- seq(-0.01, 5, 0.01)
     plot(x, ppois(x, 1), type="s", ylab="F(x)", main="Poisson(1) CDF")
     plot(x, pbinom(x, 100, 0.01),type="s", ylab="F(x)",
          main="Binomial(100, 0.01) CDF")
--------------------------------------------------------------------------------
GammaDist               package:stats                  R Documentation

The Gamma Distribution

Description:

     Density, distribution function, quantile function and random
     generation for the Gamma distribution with parameters ‘shape’ and
     ‘scale’.

Usage:

     dgamma(x, shape, rate = 1, scale = 1/rate, log = FALSE)
     pgamma(q, shape, rate = 1, scale = 1/rate, lower.tail = TRUE,
            log.p = FALSE)
     qgamma(p, shape, rate = 1, scale = 1/rate, lower.tail = TRUE,
            log.p = FALSE)
     rgamma(n, shape, rate = 1, scale = 1/rate)

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: an alternative way to specify the scale.

shape, scale: shape and scale parameters.  Must be positive, ‘scale’
          strictly.

log, log.p: logical; if ‘TRUE’, probabilities/densities p are returned
          as log(p).

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

Details:

     If ‘scale’ is omitted, it assumes the default value of ‘1’.

     The Gamma distribution with parameters ‘shape’ = a and ‘scale’ = s
     has density

                f(x)= 1/(s^a Gamma(a)) x^(a-1) e^-(x/s)

     for x >= 0, a > 0 and s > 0.  (Here Gamma(a) is the function
     implemented by R's ‘gamma()’ and defined in its help.  Note that
     a=0 corresponds to the trivial distribution with all mass at point
     0.)

     The mean and variance are E(X) = a*s and Var(X) = a*s^2.

     The cumulative hazard H(t) = - log(1 - F(t)) is ‘-pgamma(t, ...,
     lower = FALSE, log = TRUE)’.

     Note that for smallish values of ‘shape’ (and moderate ‘scale’) a
     large parts of the mass of the Gamma distribution is on values of
     x so near zero that they will be represented as zero in computer
     arithmetic.  So ‘rgamma’ can well return values which will be
     represented as zero.  (This will also happen for very large values
     of ‘scale’ since the actual generation is done for ‘scale=1’.)
```

```
Value:

     ‘dgamma’ gives the density, ‘pgamma’ gives the distribution
     function, ‘qgamma’ gives the quantile function, and ‘rgamma’
     generates random deviates.

     Invalid arguments will result in return value ‘NaN’, with a
     warning.

Note:

     The S parametrization is via ‘shape’ and ‘rate’: S has no ‘scale’
     parameter.

     ‘pgamma’ is closely related to the incomplete gamma function.  As
     defined by Abramowitz and Stegun 6.5.1 (and by ‘Numerical
     Recipes’) this is

          P(a,x) =      1/Gamma(a) integral_0^x t^(a-1) exp(-t) dt

     P(a, x) is ‘pgamma(x, a)’.  Other authors (for example Karl
     Pearson in his 1922 tables) omit the normalizing factor, defining
     the incomplete gamma function as ‘pgamma(x, a) * gamma(a)’.  A few
     use the ‘upper’ incomplete gamma function, the integral from x to
     infinity which can be computed by ‘pgamma(x, a, lower=FALSE) *
     gamma(a)’, or its normalized version. See also <URL:
     http://en.wikipedia.org/wiki/Incomplete_gamma_function>.

Source:

     ‘dgamma’ is computed via the Poisson density, using code
     contributed by Catherine Loader (see ‘dbinom’).

     ‘pgamma’ uses an unpublished (and not otherwise documented)
     algorithm ‘mainly by Morten Welinder’.

     ‘qgamma’ is based on a C translation of

     Best, D. J. and D. E. Roberts (1975).  Algorithm AS91. Percentage
     points of the chi-squared distribution.  _Applied Statistics_,
     *24*, 385-388.

     plus a final Newton step to improve the approximation.

     ‘rgamma’ for ‘shape >= 1’ uses

     Ahrens, J. H. and Dieter, U. (1982).  Generating gamma variates by
     a modified rejection technique.  _Communications of the ACM_,
     *25*, 47-54,

     and for ‘0 < shape < 1’ uses

     Ahrens, J. H. and Dieter, U. (1974).  Computer methods for
     sampling from gamma, beta, Poisson and binomial distributions.
     _Computing_, *12*, 223-246.

References:

     Becker, R. A., Chambers, J. M. and Wilks, A. R. (1988) _The New S
     Language_.  Wadsworth & Brooks/Cole.

     Shea, B. L. (1988) Algorithm AS 239, Chi-squared and incomplete
     Gamma integral, _Applied Statistics (JRSS C)_ *37*, 466-473.

     Abramowitz, M. and Stegun, I. A. (1972) _Handbook of Mathematical
     Functions._ New York: Dover.  Chapter 6: Gamma and Related
     Functions.

See Also:

     ‘gamma’ for the gamma function.

     Distributions for other standard distributions, including ‘dbeta’
     for the Beta distribution and ‘dchisq’ for the chi-squared
     distribution which is a special case of the Gamma distribution.

Examples:

     -log(dgamma(1:4, shape=1))
     p <- (1:9)/10
     pgamma(qgamma(p,shape=2), shape=2)
     1 - 1/exp(qgamma(p, shape=1))
```

```
# even for shape = 0.001 about half the mass is on numbers
# that cannot be represented accurately (and most of those as zero)
pgamma(.Machine$double.xmin, 0.001)
pgamma(5e-324, 0.001)  # on most machines 5e-324 is the smallest
                       # representable non-zero number
table(rgamma(1e4, 0.001) == 0)/1e4
```