NTNU
Kunnskap for ei betre verd

Institutt for (instituttnamn): Institutt for matematiske fag.

Eksamensoppgåve i (emnekode) (emnenamn)

ST2304 Statistisk modellering for biologer og bioteknologer

Fagleg kontakt under eksamen: Jarle Tufto

Tlf.: 99705519

Eksamensdato: 15. mai, 2015

Eksamenstid (frå-til): 9-13

Hjelpemiddelkode/Tillatne hjelpemiddel: Tillatne hjelpemidler: Eit håndskreve gult A4 ark, kalkulator, ``Tabeller og formler i statistikk'' (Tapir forlag), K. Rottmann: Matematisk formelsamling.

Annan informasjon:

Målform/språk: Nynorsk

Sidetal (utan framside): 7

Sidetal vedlegg: 0

Kontrollert av:

Dato Sign

Noregs teknisk-naturvitskaplege universitet
Institutt for matematiske fag

Hjelpesider for nokre R funksjonar det kan hende du får bruk for følgjer på side 6.

Oppgåve 1 Gå ut i frå at vi set opp eit stort tal innsektsfellar langs ein linje i terrenget for å finne ein skjeldsynt innsektsart. La X_i vere talet på eksemplar av arten som er gått i den i'te fella når denne vert undersøkt etter eitt døger, og gå ut i frå at kvar X_i er uavhengig Poisson-fordelt med forventningsverdi 0.002.

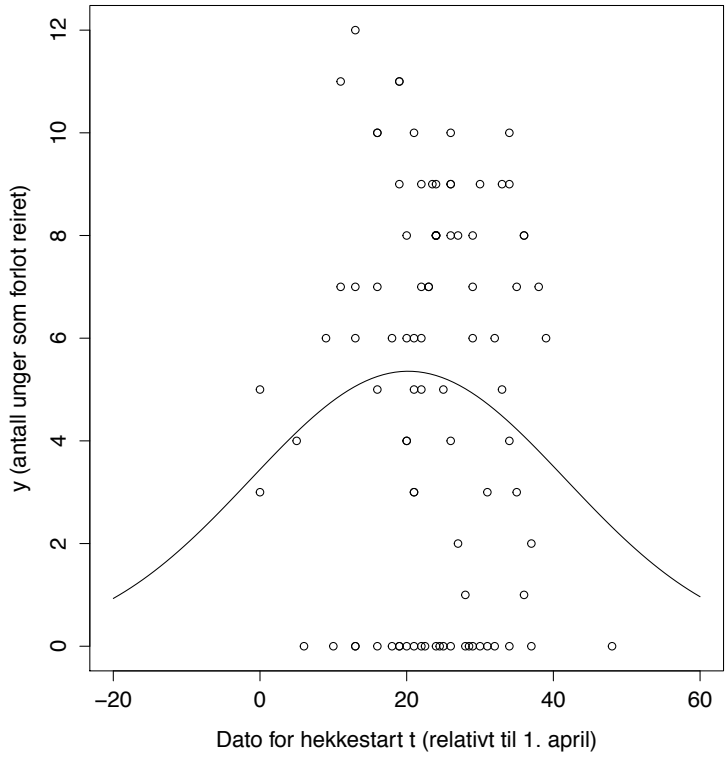
a) Grunngje kort kvifor føresetnaden om Poisson-fordeling kan vere rimeleg og skriv eit R-uttrykk som reknar ut sannsynet for at minst eitt eksemplar av arten vert fanga i ei gjeve felle. Skriv uttrykket slik at verdien blir teke vare på i variabelen `p.tilstades`.

La Y være talet på feller som vi må undersøke før vi finn ei felle kor den sjeldsynte arten er tilstades.

b) Kva for fordeling har Y om vi føreset at talet på fellar er ubegrensa? Skriv R-uttrykk som reknar ut sannsynet for at Y tek ein verdi mellom 18.2 og 30.5 og sannsynet for at Y tek akkurat verdien 20.

c) Skriv eit R-uttrykk som reknar ut eit 95%-sannsynsintervall for Y, altså nedre og øvre 2.5%-kvantil til Y.

Oppgåve 2 Vi ønskjer å undersøke korleis tid for hekkestart hos kjøttmeispar påverkar talet på utflygne ungar. Kjøttmeis legg opptil to kull kvar sesong. Vi registrerer difor tid for hekkestart t (talet på dagar etter 1. april) og totalt tal på utflygne ungar y for tilsaman 92 hekkande kjøttmeispar. Dei observerte dataene er som i figuren under.



Vi tilpassar så ein generalisert lineær modell til dataene på følgjande måte.

```
> t2 <- t^2
> fit <- glm(y ~ t + t2, family=poisson(link="log"))
> summary(fit)

Call:
glm(formula = y ~ t + t2, family = poisson(link = "log"))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-3.2729  -2.8407   0.2778   1.1982   2.6146

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.2361689  0.2547556   4.852 1.22e-06 ***
```

```
t            0.0437550  0.0222333   1.968   0.0491 *
t2          -0.0010827  0.0004782  -2.264   0.0236 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 360.23  on 91  degrees of freedom
Residual deviance: 353.81  on 89  degrees of freedom
AIC: 606.07

Number of Fisher Scoring iterations: 5
```

a) La β_0, β_1 og β_2 vere parameterane i modellen (der β_2 er regressjonkoeffisienten for t^2). Skriv opp modellen i matematisk notasjon. Skriv i tillegg opp kva forventa tal på utflydde ungar, $E(Y)$, blir som funksjon av tid for hekkestart t.

b) $E(Y)$ som funksjon av t er plottet i figuren over. Om vi reparameteriserar modellen kan vi skrive samanhengen på formen

$$E(Y) = y_0 e^{-\frac{1}{2}\left(\frac{t-t_0}{\omega}\right)^2}, \tag{1}$$

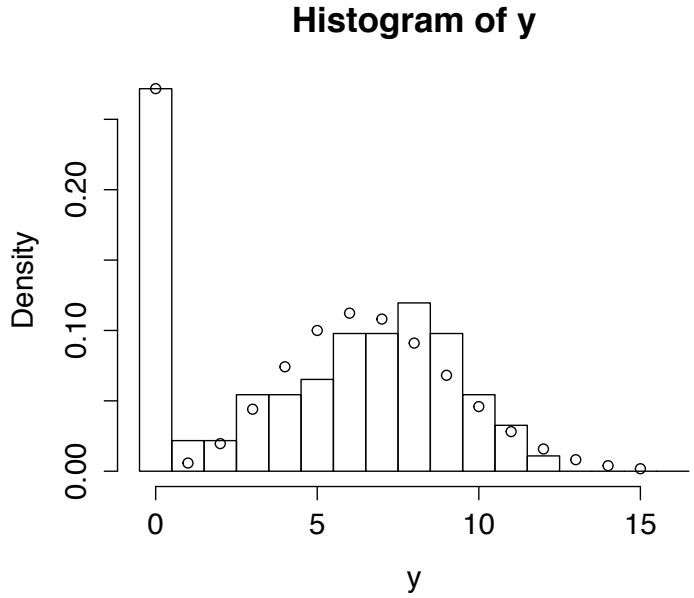
altså ei Gauss-kurve (sjå figur) der t_0 er optimalt hekkestartstidspunkt, y_0 er forventa tal på utflydde ungar for hekkestart ved $t = t_0$, og ω er eit mål på «breidda» til Gausskurva (i dagar) (analogt med standardavviket til ein normalfordelt variabel). Vis at samanhengen mellom ω og β_2 i regresjonsmodellen over er gjeve ved

$$\omega = \sqrt{-\frac{1}{2\beta_2}}. \tag{2}$$

c) Rekn ut eit estimat av ω og tilhøyrande standardfeil til dette estimatet.

d) Test om det er overdispersjon i dataene. Rund om nødvendig av talet på fridomsgrader til næraste multiplum av 10. Kan vi i lys av dette stole på standardfeilet rekna ut i punkt c? Vi estimerar variansen til Y til å være omlag 2.8 gangar større enn forventa dersom Y hadde hatt same varians som ein Poisson-fordelt variabel. Rekn ut nye korrigera estimat av standardfeilen til $\hat{\beta}_2$ og $\hat{\omega}$. Diskuter kort konkrete mekanismar som kan generere overdispersjon i dette tilfellet.

I resten av oppgåva skal vi i meir detalj studere fordelinga til variabelen Y skildra tidlegare (talet på ungar produsert av ulike kjøttmeispar i løpet av hekkesesongen). Eit histogram av den observerte fordelinga til Y (representert ved stolpane i histogrammet) er gjeve nedanfor.



Histogrammet tydar på at det er eit høgare sannsyn for at Y tek verdien null, såkalla null-inflasjon, i forhold til kva ein kunne forvente dersom fordelinga hadde vore beskrivd ved ein Poisson-fordeling. For å modellere dette tilpassar vi ein model H_1 der

$$P(Y = y) = p_0 I_0(y) + (1 - p_0)\frac{e^{-\lambda}\lambda^y}{y!}, \tag{3}$$

kor $I_0(y)$ er ein funksjon av y som tek verdien 1 for $y = 0$ og verdien 0 for $y \neq 0$. Denne modellen representerer da at null-inflasjon inntreff med sannsyn p_0 (mislykka hekking f.eks. som følgje av at ein rugekasse vert teke over av ein svart-kvit fluesnappar). Gjeve null-inflasjon tek Y verdien 0 med sannsyn 1. Gjeve at null-inflasjon ikkje intreffer (med sannsynlighet $1-p_0$) er Y poissonfordelt med parameter λ.

Vi estimerer p_0 og λ ved å programmere minus log-likelihoodfunksjonen for modellen omtala over som ein funksjon med navn `lnL`. Vi tilpassar så modellen numerisk på følgjande måte i R. Den tilpassa modellen er representert ved sirklane i figuren over.

```
> fit <- optim(c(.2,5),lnL,y=y,hessian=TRUE)
> fit
$par
[1] 0.2709197 6.7380696

$value
[1] 213.6592

$counts
function gradient
      57       NA

$convergence
[1] 0

$message
NULL

$hessian
            [,1]       [,2]
[1,] 463.7016481 -0.4010448
[2,]  -0.4010448  9.8763866

> solve(fit$hessian)
             [,1]         [,2]
[1,] 2.156635e-03 8.757323e-05
[2,] 8.757323e-05 1.012552e-01
```

e) Kva er sannsynsmaksimeringsestimata av p_0 og λ samt tilnærma standardfeil til desse estimatane?

f) Skriv opp eit matematisk uttrykk for log-likelihoodfunksjonen for modellen utan null-inflasjon H_0 og rekn ut det maksimale log likelihoodet under denne modellen. Det er ikkje nødvendig å utleie SMEen av λ dersom du kjem i håg denne. Det er oppgjeve at utvalsstorleiken $n = 92$, at $\sum_{i=1}^{n} y_i = 452$, og at $\sum_{i=1}^{n} \ln(y_i!) = 452$. Kan vi forkaste H_0 til fordel for H_1?

g) Programmer funksjonen `lnL` som vi har brukt over for å tilpasse modellen H_1. Dette kan gjerast f.eks. ved å nytte vektorisert if-else-setning, sjå hjelpesida til `ifelse`, eller ved å nytte logiske vektorar som indeksar. Det kan også være høveleg å nytte funksjonen `dpois`.

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.

The length of the result is determined by 'n' for 'rpois', and is the maximum of the lengths of the numerical arguments for the other functions.

The numerical arguments other than 'n' are recycled to the length of the result. Only the first elements of the logical arguments are used.

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")
```

Geometric package:stats R Documentation

The Geometric Distribution

Description:

Density, distribution function, quantile function and random generation for the geometric distribution with parameter 'prob'.

Usage:

```
dgeom(x, prob, log = FALSE)
pgeom(q, prob, lower.tail = TRUE, log.p = FALSE)
qgeom(p, prob, lower.tail = TRUE, log.p = FALSE)
rgeom(n, prob)
```

Arguments:

x, q: vector of quantiles representing the number of failures in a sequence of Bernoulli trials before success occurs.

p: vector of probabilities.

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

prob: probability of success in each trial. '0 < prob <= 1'.

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 geometric distribution with 'prob' = p has density

$$p(x) = p\,(1-p)^x$$

for x = 0, 1, 2, ..., 0 < p <= 1.

If an element of 'x' is not integer, the result of 'dgeom' is zero, with a warning.

The quantile is defined as the smallest value x such that F(x) >= p, where F is the distribution function.

Value:

'dgeom' gives the density, 'pgeom' gives the distribution function, 'qgeom' gives the quantile function, and 'rgeom' generates random deviates.

Invalid 'prob' will result in return value 'NaN', with a warning.

The length of the result is determined by 'n' for 'rgeom', and is

```
     the maximum of the lengths of the numerical arguments for the
     other functions.

     The numerical arguments other than 'n' are recycled to the length
     of the result.  Only the first elements of the logical arguments
     are used.

Source:

     'dgeom' computes via 'dbinom', using code contributed by Catherine
     Loader (see 'dbinom').

     'pgeom' and 'qgeom' are based on the closed-form formulae.

     'rgeom' uses the derivation as an exponential mixture of Poissons,
     see

     Devroye, L. (1986) _Non-Uniform Random Variate Generation._
     Springer-Verlag, New York. Page 480.

See Also:

     Distributions for other standard distributions, including
     'dnbinom' for the negative binomial which generalizes the
     geometric distribution.

Examples:

     qgeom((1:9)/10, prob = .2)
     Ni <- rgeom(20, prob = 1/4); table(factor(Ni, 0:max(Ni)))
--------------------------------------------------------------------------
ifelse                  package:base                   R Documentation

Conditional Element Selection

Description:

     'ifelse' returns a value with the same shape as 'test' which is
     filled with elements selected from either 'yes' or 'no' depending
     on whether the element of 'test' is 'TRUE' or 'FALSE'.

Usage:

     ifelse(test, yes, no)

Arguments:

    test: an object which can be coerced to logical mode.

     yes: return values for true elements of 'test'.

      no: return values for false elements of 'test'.

Details:

     If 'yes' or 'no' are too short, their elements are recycled.
     'yes' will be evaluated if and only if any element of 'test' is
     true, and analogously for 'no'.

     Missing values in 'test' give missing values in the result.

Value:

     A vector of the same length and attributes (including dimensions
     and '"class"') as 'test' and data values from the values of 'yes'
     or 'no'.  The mode of the answer will be coerced from logical to
     accommodate first any values taken from 'yes' and then any values
     taken from 'no'.

Warning:

     The mode of the result may depend on the value of 'test' (see the
     examples), and the class attribute (see 'oldClass') of the result
     is taken from 'test' and may be inappropriate for the values
     selected from 'yes' and 'no'.

     Sometimes it is better to use a construction such as

       (tmp <- yes; tmp[!test] <- no[!test]; tmp)
     , possibly extended to handle missing values in 'test'.

References:

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

See Also:

     'if'.

Examples:

     x <- c(6:-4)
     sqrt(x)  #- gives warning
     sqrt(ifelse(x >= 0, x, NA))  # no warning

     ## Note: the following also gives the warning !
     ifelse(x >= 0, sqrt(x), NA)

     ## example of different return modes:
     yes <- 1:3
     no <- pi^(0:3)
     typeof(ifelse(NA, yes, no))    # logical
     typeof(ifelse(TRUE, yes, no))  # integer
     typeof(ifelse(FALSE, yes, no)) # double
```