



Bokmål

Faglig kontakt under eksamen: Professor Jarle Tufto
Telefon: 73591888

Statistisk modellering for biologer og bioteknologer, ST2304

4. juni, 2012

Kl. 9–13

Sensur: 25. juni, 2012

Tillatte hjelpemidler: Et håndskrevet gult A4 ark, kalkulator, “Tabeller og formler i statistikk” (Tapir forlag), K. Rottmann: Matematisk formelsamling.

Hjelpesider for noen R funksjoner det kan hende du får bruk for følger på side 6.

Oppgave 1 Anta at X er binomisk fordelt med parametere $n = 50$ og $p = 0.2$. Skriv R uttrykk som beregner følgende.

- Sannsynligheten for hendelsen $X < 10 \cap X > 8$.
- Sannsynligheten for hendelsen $X \geq 11$.
- Medianen samt nedre og øvre 5%-kvantil til X .
- Et histogram av 1000 simulerte realisasjoner av X .

Oppgave 2 For å undersøke spredningen av den fremmede arten Parkslirekne (*Fallopia japonica*) teller vi, i et gitt år, antall individer y i ruter på 10×10 meter plassert langs en 2 km lang transekt på tvers av spredningsfronten med mellomrom 100 meter. De observerte dataene er vist i Fig. 1.

- Forklar hvorfor det ville vært problematisk og estimere hvordan y avhenger av x ved hjelp av vanlig lineær regresjon. Ville det hjulpet å log-transformere responsvariabelen y ?

Vi tilpasser en generalisert lineær modell med log link-funksjon hvor vi antar at responsvariabelen er Poissonfordelt i R på følgende måte.

```
> fit0 <- glm(y~x,fam=poisson(link="log"))
> summary(fit0)
```

Call:

```
glm(formula = y ~ x, family = poisson)
```

Deviance Residuals:

Min	1Q	Median	3Q	Max
-1.8690	-1.0176	-0.1243	0.3866	2.3161

Coefficients:

	Estimate	Std. Error	z value	Pr(> z)
(Intercept)	3.3007888	0.1135983	29.06	<2e-16 ***
x	-0.0018694	0.0001814	-10.30	<2e-16 ***

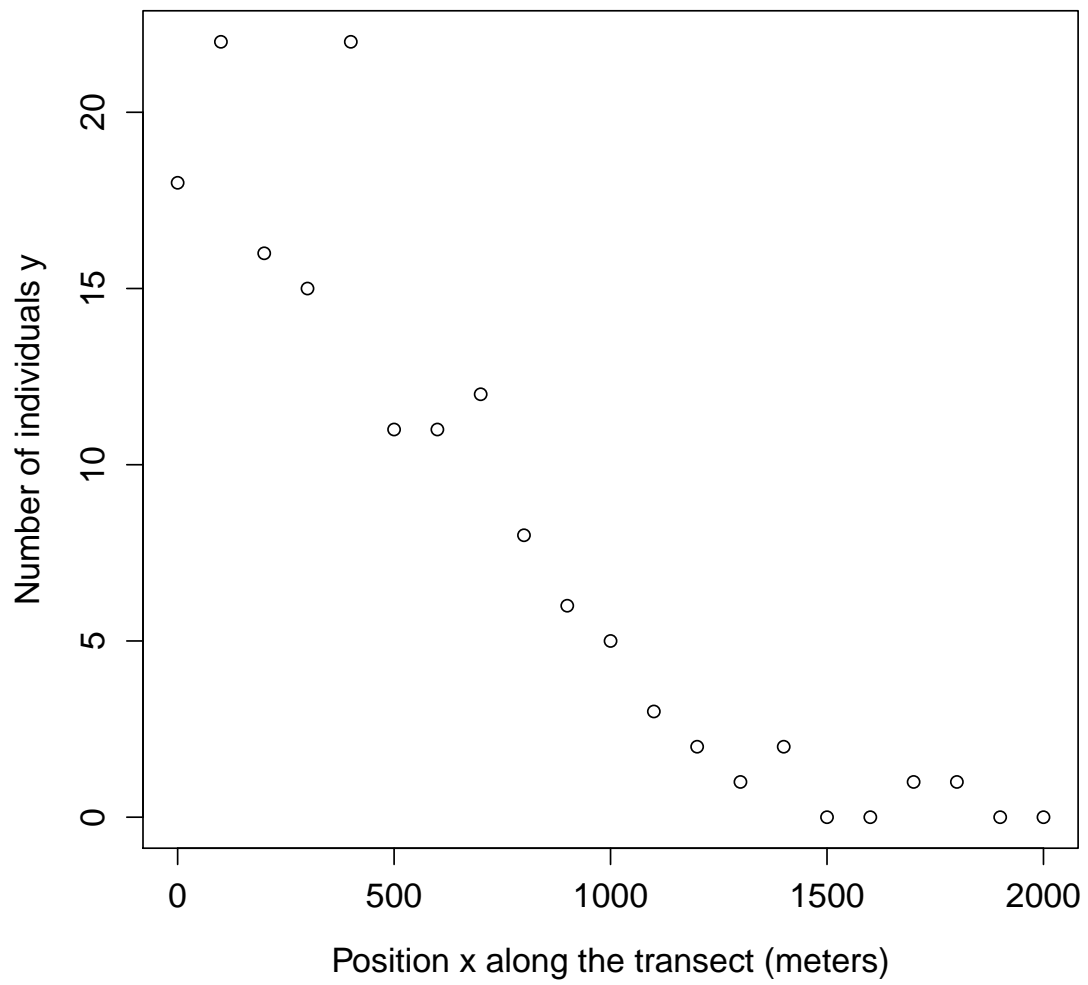
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

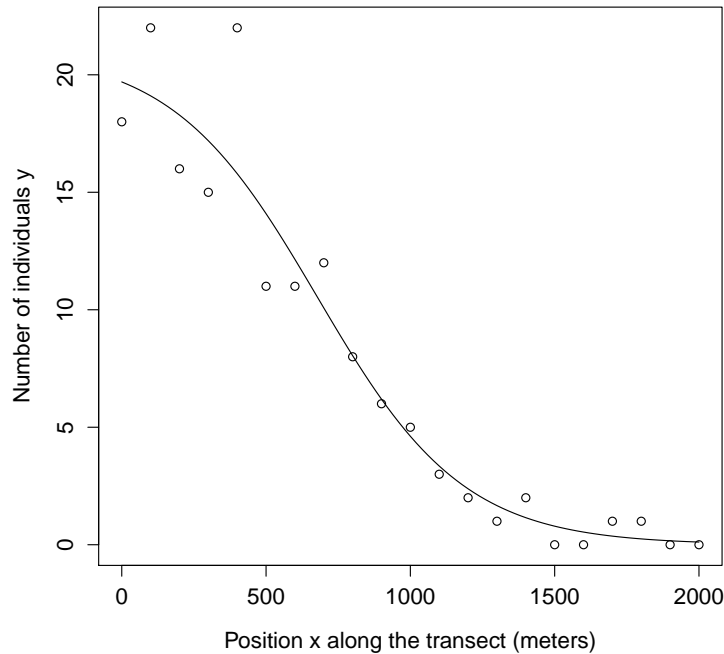
Null deviance: 168.514 on 20 degrees of freedom
Residual deviance: 23.428 on 19 degrees of freedom
AIC: 89.35

Number of Fisher Scoring iterations: 5

- b) Skriv opp modellen i matematisk notasjon. Begrunn kort hvorfor modellantakelsene dette innebærer kan være fornuftige.
- c) Tegn en kurve (for hånd) i Fig. 1 som representerer estimert forventet antall individ i hver rute som funksjon av avstand x . Lever side 3 sammen med besvarelsen eller tegn opp figuren på nytt på eget ark.
- d) Test om det er overdispersjon i dataene og diskuter ulike mekanismer som kan generere overdispersjon i denne konkrete situasjonen.



Figur 1: Observerte data



Figur 2: Alternative model

Ut i fra teoretiske spredningsmodeller er det rimelig å anta at populasjonstettheten (f.eks. uttrykket ved gjennomsnittlig antall individ per rute) vil stabilisere seg rundt en viss lokal bæreevne K et stykke bak spredningsfronten og at populasjonstettheten λ som funksjon av posisjon x dermed bør være beskrevet ved en sigmoid kurve, f.eks. funksjonen

$$\lambda = \frac{K}{1 + e^{c(x-x_0)}} \quad (1)$$

Denne funksjonen går mot K når $x \rightarrow -\infty$, har x_0 som vendepunkt, og parameteren c bestemmer hvor “bratt” kurven er, se Fig. 2. Anta som før at observert antall individ i de ulike posisjonene x er Poissonfordelt med parameter λ gitt ved ligning (1).

- e) Skriv en R-funksjon `lnL` som beregner den negative log likelihoodfunksjonen for denne modellen, gitt at vektorene \mathbf{x} og \mathbf{y} som før inneholder de observerte dataene. La funksjonens første argument være en vektor som inneholder verdien av parameterne x_0 , c og K .
- f) Tilpasning av modellen gitt ved (1) ved hjelp av numeriske metoder gir følgende resultat i R.

```
> fitH1 <- optim(c(1000,.005,30),lnL,x=x,y=y,hessian=TRUE)
> fitH1
$par
[1] 678.620207245 -0.003950444 21.046689225

$value
[1] 36.32522

$counts
function gradient
      431      NA

$convergence
[1] 0

$message
NULL

$hessian
      [,1]      [,2]      [,3]
[1,] 4.129781e-04 2.478797e+01 9.426691e-03
[2,] 2.478797e+01 5.609630e+06 1.312663e+02
[3,] 9.426691e-03 1.312663e+02 3.521738e-01

> solve(fitH1$hessian)
      [,1]      [,2]      [,3]
[1,] 1.261668e+04 -4.826937e-02 -3.197211e+02
[2,] -4.826937e-02 3.645042e-07 1.156172e-03
[3,] -3.197211e+02 1.156172e-03 1.096659e+01
```

Det kan vises at den generaliserte lineære modellen i punkt b) er nøstet i modellen gitt ved ligning (1). Bruk dette til å utføre en tilnærmet test av modellen i punkt b) versus modellen gitt ved ligning (1). Hvor mange frihetsgrader har testobservatoren? Det oppgis at det maksimale log likelihoodet for den generaliserte lineære modellen i punkt b) er -42.67. Beregn også et estimat av standardfeilen til estimatet av K .

Binomial package:stats R Documentation

The Binomial Distribution

Description:

Density, distribution function, quantile function and random generation for the binomial distribution with parameters 'size' and 'prob'.

Usage:

```
dbinom(x, size, prob, log = FALSE)
pbinom(q, size, prob, lower.tail = TRUE, log.p = FALSE)
qbinom(p, size, prob, lower.tail = TRUE, log.p = FALSE)
rbinom(n, size, prob)
```

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.

`size`: number of trials (zero or more).

`prob`: probability of success on each trial.

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

`lower.tail`: logical; if TRUE (default), probabilities are $P[X \leq x]$, otherwise, $P[X > x]$.

Details:

The binomial distribution with 'size' = `n` and 'prob' = `p` has density

$$p(x) = \text{choose}(n,x) p^x (1-p)^{(n-x)}$$

for $x = 0, \dots, n$. Note that binomial `_coefficients_` can be computed by 'choose' in R.

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

The quantile is defined as the smallest value `x` such that $F(x) \geq p$, where F is the distribution function.

Value:

'dbinom' gives the density, 'pbinom' gives the distribution function, 'qbinom' gives the quantile function and 'rbinom' generates random deviates.

If 'size' is not an integer, 'NaN' is returned.

Source:

For 'dbinom' a saddle-point expansion is used: see

Catherine Loader (2000). `_Fast and Accurate Computation of Binomial Probabilities_`; available from <URL: <http://www.herine.net/stat/software/dbinom.html>>.

'pbinom' uses 'pbeta'.

'qbinom' uses the Cornish-Fisher Expansion to include a skewness correction to a normal approximation, followed by a search.

'rbinom' (for 'size < .Machine\$integer.max') is based on

Kachitvichyanukul, V. and Schmeiser, B. W. (1988) Binomial random variate generation. `_Communications of the ACM_`, *31*, 216-222.

For larger values it uses inversion.

See Also:

Distributions for other standard distributions, including

'dtnbinom' for the negative binomial, and 'dpois' for the Poisson distribution.

Examples:

```
require(graphics)
# Compute P(45 < X < 55) for X Binomial(100,0.5)
sum(dbinom(46:54, 100, 0.5))

## Using "log = TRUE" for an extended range :
n <- 2000
k <- seq(0, n, by = 20)
plot(k, dbinom(k, n, pi/10, log=TRUE), type='l', ylab="log density",
      main = "dbinom(*, log=TRUE) is better than log(dbinom(*))")
lines(k, log(dbinom(k, n, pi/10)), col='red', lwd=2)
## extreme points are omitted since dbinom gives 0.
mtext("dbinom(k, log=TRUE)", adj=0)
mtext("extended range", adj=0, line = -1, font=4)
mtext("log(dbinom(k))", col="red", adj=1)
```

hist package:graphics R Documentation

Histograms

Description:

The generic function 'hist' computes a histogram of the given data values. If 'plot=TRUE', the resulting object of class 'histogram' is plotted by 'plot.histogram', before it is returned.

Usage:

```
hist(x, ...)
```

Default S3 method:

```
hist(x, breaks = "Sturges",
     freq = NULL, probability = !freq,
     include.lowest = TRUE, right = TRUE,
     density = NULL, angle = 45, col = NULL, border = NULL,
     main = paste("Histogram of", xname),
     xlim = range(breaks), ylim = NULL,
     xlab = xname, ylab,
     axes = TRUE, plot = TRUE, labels = FALSE,
     nclass = NULL, warn.unused = TRUE, ...)
```

Arguments:

`x`: a vector of values for which the histogram is desired.

`breaks`: one of:

- a vector giving the breakpoints between histogram cells,
- a single number giving the number of cells for the histogram,
- a character string naming an algorithm to compute the number of cells (see 'Details'),
- a function to compute the number of cells.

In the last three cases the number is a suggestion only.

`freq`: logical; if 'TRUE', the histogram graphic is a representation of frequencies, the 'counts' component of the result; if 'FALSE', probability densities, component 'density', are plotted (so that the histogram has a total area of one). Defaults to 'TRUE' if and only if 'breaks' are equidistant (and 'probability' is not specified).

`probability`: an `_alias_` for 'freq', for S compatibility.

`include.lowest`: logical; if 'TRUE', an `x[i]` equal to the 'breaks' value will be included in the first (or last, for 'right = FALSE') bar. This will be ignored (with a warning) unless 'breaks' is a vector.

`right`: logical; if 'TRUE', the histogram cells are right-closed (left open) intervals.

`density`: the density of shading lines, in lines per inch. The default value of 'NULL' means that no shading lines are drawn.

Non-positive values of 'density' also inhibit the drawing of shading lines.

angle: the slope of shading lines, given as an angle in degrees (counter-clockwise).

col: a colour to be used to fill the bars. The default of 'NULL' yields unfilled bars.

border: the color of the border around the bars. The default is to use the standard foreground color.

main, xlab, ylab: these arguments to 'title' have useful defaults here.

xlim, ylim: the range of x and y values with sensible defaults. Note that 'xlim' is `_not_` used to define the histogram (breaks), but only for plotting (when `plot = TRUE`).

axes: logical. If 'TRUE' (default), axes are draw if the plot is drawn.

plot: logical. If 'TRUE' (default), a histogram is plotted, otherwise a list of breaks and counts is returned. In the latter case, a warning is used if (typically graphical) arguments are specified that only apply to the 'plot = TRUE' case.

labels: logical or character. Additionally draw labels on top of bars, if not 'FALSE'; see 'plot.histogram'.

nclass: numeric (integer). For S(-PLUS) compatibility only, 'nclass' is equivalent to 'breaks' for a scalar or character argument.

warn.unused: logical. If 'plot=FALSE' and 'warn.unused=TRUE', a warning will be issued when graphical parameters are passed to 'hist.default()'.
 ...: further arguments and graphical parameters passed to 'plot.histogram' and thence to 'title' and 'axis' (if 'plot=TRUE').

Details:

The definition of `_histogram_` differs by source (with country-specific biases). R's default with equi-spaced breaks (also the default) is to plot the counts in the cells defined by 'breaks'. Thus the height of a rectangle is proportional to the number of points falling into the cell, as is the area `_provided_` the breaks are equally-spaced.

The default with non-equi-spaced breaks is to give a plot of area one, in which the `_area_` of the rectangles is the fraction of the data points falling in the cells.

If 'right = TRUE' (default), the histogram cells are intervals of the form '(a, b]', i.e., they include their right-hand endpoint, but not their left one, with the exception of the first cell when 'include.lowest' is 'TRUE'.

For 'right = FALSE', the intervals are of the form '[a, b)', and 'include.lowest' means 'include highest'.

A numerical tolerance of $1e-7$ times the median bin size is applied when counting entries on the edges of bins. This is not included in the reported 'breaks' nor (as from R 2.11.0) in the calculation of 'density'.

The default for 'breaks' is "Sturges": see 'nclass.Sturges'. Other names for which algorithms are supplied are "Scott" and "FD" / "Freedman-Diaconis" (with corresponding functions 'nclass.scott' and 'nclass.FD'). Case is ignored and partial matching is used. Alternatively, a function can be supplied which will compute the intended number of breaks as a function of 'x'.

Value:

an object of class "histogram" which is a list with components:

breaks: the $n+1$ cell boundaries (= 'breaks' if that was a vector). These are the nominal breaks, not with the boundary fuzz.

counts: n integers; for each cell, the number of 'x[]' inside.

density: values $f^{\wedge}(x[i])$, as estimated density values. If 'all(diff(breaks) == 1)', they are the relative frequencies 'counts/n' and in general satisfy $\sum[i; f^{\wedge}(x[i]) (b[i+1]-b[i])] = 1$, where $b[i] = \text{'breaks'}[i]$.

intensities: same as 'density'. Deprecated, but retained for compatibility.

mids: the n cell midpoints.

xname: a character string with the actual 'x' argument name.

equidist: logical, indicating if the distances between 'breaks' are all the same.

References:

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

Venables, W. N. and Ripley, B. D. (2002) *Modern Applied Statistics with S*. Springer.

See Also:

'nclass.Sturges', 'stem', 'density', 'truehist' in package 'MASS'.
 Typical plots with vertical bars are `_not_` histograms. Consider 'barplot' or 'plot(*, type = "h")' for such bar plots.

Examples:

```
op <- par(mfrow=c(2, 2))
hist(islands)
utils::str(hist(islands, col="gray", labels = TRUE))

hist(sqrt(islands), breaks = 12, col="lightblue", border="pink")
##-- For non-equidistant breaks, counts should NOT be graphed unscaled:
r <- hist(sqrt(islands), breaks = c(4*0:5, 10*3:5, 70, 100, 140),
          col='blue1')
text(r$mids, r$density, r$counts, adj=c(.5, -.5), col='blue3')
sapply(r[2:3], sum)
sum(r$density * diff(r$breaks)) # == 1
lines(r, lty = 3, border = "purple") # -> lines.histogram(*)
par(op)

require(utils) # for str
str(hist(islands, breaks=12, plot= FALSE)) #-> 10 (~= 12) breaks
str(hist(islands, breaks=c(12,20,36,80,200,1000,17000), plot = FALSE))

hist(islands, breaks=c(12,20,36,80,200,1000,17000), freq = TRUE,
      main = "WRONG histogram") # and warning

require(stats)
set.seed(14)
x <- rchisq(100, df = 4)

## Comparing data with a model distribution should be done with qqplot()!
qqplot(x, qchisq(ppoints(x), df = 4)); abline(0,1, col = 2, lty = 2)

## if you really insist on using hist() ... :
hist(x, freq = FALSE, ylim = c(0, 0.2))
curve(dchisq(x, df = 4), col = 2, lty = 2, lwd = 2, add = TRUE)
```

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 $P[X \leq x]$, otherwise, $P[X > x]$.

Details:

The Poisson distribution has density

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

for $x = 0, 1, 2, \dots$. The mean and variance are $E(X) = \text{Var}(X) = \lambda$.

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 $P(X \leq x) \geq p$.

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")
```