NTNU

Kunnskap for en bedre verden

Institutt for matematiske fag

Eksamensoppgave i
ST2304 Statistisk modellering for biologer og bioteknologer

Faglig kontakt under eksamen: Jarle Tufto

Tlf: 99 70 55 19

Eksamensdato: 12. august 2015

Eksamenstid (fra–til): _____

Hjelpemiddelkode/Tillatte hjelpemidler: *Tabeller og formler i statistikk*, Tapir Forlag, K. Rottmann: *Matematisk formelsamling*, Kalkulator Casio fx-82ES PLUS, CITIZEN SR-270X, CITIZEN SR-270X College eller HP30S, ett gult A4-ark med egne håndskrevne notater.

Annen informasjon:
Hjelpesider for noen R funksjoner som du kan få bruk for følger i vedlegget. Alle svar skal begrunnes og besvarelsen skal inneholde naturlig mellomregning.

Målform/språk: bokmål

Antall sider: 5

Antall sider vedlegg: 3

Kontrollert av:

Dato Sign

Oppgave 1 Om vi har to tilfeldig utvalg $X_1, X_2, \ldots, X_{n_X}$ og $Y_1, Y_2, \ldots, Y_{n_Y}$ fra to normalfordelinger $N(\mu_X, \sigma_X^2)$ og $N(\mu_Y, \sigma_Y^2)$ kan det vises at et $(1-\alpha)$-konfidensintervall for forholdet σ_X^2/σ_Y^2 mellom de to ukjente variansene i hver populasjon er gitt ved

$$\left(\frac{S_X^2}{S_Y^2 F_{\alpha/2, n_X-1, n_Y-1}}, \frac{S_X^2}{S_Y^2 F_{1-\alpha/2, n_X-1, n_Y-1}} \right), \tag{1}$$

hvor S_X^2 og S_Y^2 er de to utvalgsvariansene og $F_{\alpha/2, n_X-1, n_Y-1}$ betegner øvre $\alpha/2$-kvantil i en F-fordeling med $n_X - 1$ og $n_Y - 1$ frihetsgrader.

a) Gitt to vektorer i R med navn `x` og `y`, skriv et kort R-skript eller -uttrykk som beregner intervallet gitt at vi velger $\alpha = 0.05$.

b) Skriv også et R-uttrykk som beregner et $(1-\alpha)$-konfidensintervall for forholdet σ_X/σ_Y mellom de to standardavvikene.

Oppgave 2 Anta at vi bestemmer genotypen til $n = 100$ individer trukket tilfeldig fra en tilnærmet uendelig stor populasjon. La de stokastiske variablene X_{AA}, X_{Aa} og X_{aa} betegne antall individ av genotype AA, Aa og aa i utvalget og anta at disse antallene tar verdiene 22, 54 og 24 i vårt konkrete observerte utvalg.

a) Hva er navnet på den simultane fordelingen til X_{AA}, X_{Aa} og X_{aa} og hva er parameterene i fordelingen?

Skriv et R-uttrykk som beregner sannsynligheten for at X_{AA}, X_{Aa} og X_{aa} tar akkurat verdiene i det observerte utvalget under forutsetning av at populasjonen vi trekker fra er i Hardy-Weinberg likevekt og at frekvensen av de tre genotypene i hele populasjonen er henholdvis 0.25, 0.5 og 0.25.

I den ene øvingen i kurset har vi vist at sannsynlighetsmaksimeringsestimatet av populasjonsfrekvensen av den ene genvarianten (allelet) A er gitt ved

$$\hat{p}_A = \frac{2X_{AA} + X_{Aa}}{2n} \tag{2}$$

under forutsetningene gitt ovenfor.

b) Beregn $\hat{p}_A$ og estimater $\hat{P}_{AA}, \hat{P}_{Aa}, \hat{P}_{aa}$ av de tre tilhørende genotypefrekvensene i populasjonen gitt det observerte utvalget. Forutsett Hardy-Weinberglikevekt.

La

$$D = \sum_{i \in \{AA, Aa, aa\}} \frac{(X_i - n\hat{P}_i)^2}{n\hat{P}_i}. \tag{3}$$

c) Hva er den tilnærmede fordelingen til D under nullhypotesen at populasjonen er i Hardy-Weinberglikevekt og hvilke(n) verdi(er) har parameteren(e) i fordelingen?

Forklar hvordan D kan brukes for å teste om populasjonen er i Hardy-Weinberglikevekt.

Det oppgis at $D = 0.64$ for det observerte utvalget. Hva blir da testens konklusjon?

Skriv også et R-uttrykk som beregner testen p-verdi. Forklar kort forskjellen på testens p-verdi og sannsynligheten omtalt i punkt a).

d) Anta at standardfeilen til estimatet av p_A beregnet i punkt b) er 0.035. Bruk dette til å finne den tilnærmede standardfeilen til estimatet av frekvensen av genotypen aa i populasjonen, alstå parameteren $P_{aa} = (1 - p_A)^2$.

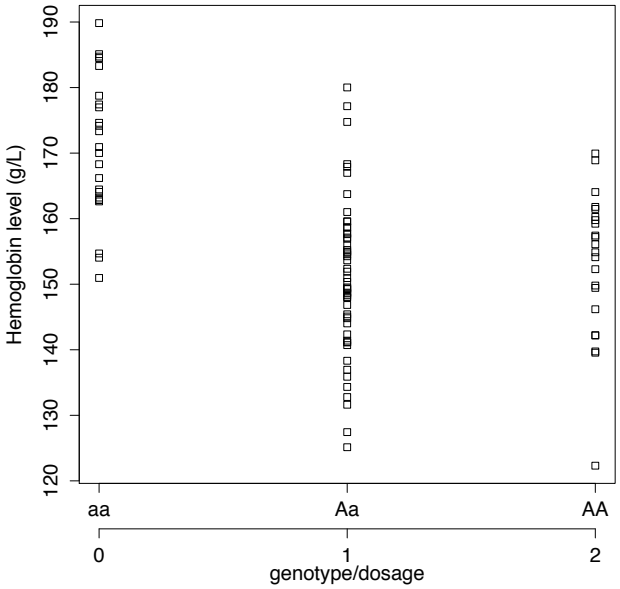
I resten av oppgaven skal vi studere hvordan de tre genotypene påvirker nivået av hemoglobin i blodet til de ulike individene i populasjonen. Vi mistenker at genvarianten (allelet) a disponerer for forhøyede hemoglobinnivå (variabelen `hemoglobin` nedenfor) og at denne effekten muligens kun uttrykkes recessivt. For å modellere dette lager vi to alternative kategoriske forklaringsvariabler; `genotype` med tre nivå (`aa`,`Aa` og `AA`) og `genotyperecessive` med to nivå (`aa` og `A-` hvor nivået `A-` representerer en sammenslåing av `Aa` og `AA`)).

Setter vi dataene sammen i en dataframe blir denne da (delvis) seende ut som følger.

```
> data
    genotype genotyperecessive dosage hemoglobin
1         AA                A-      2      142.2
2         AA                A-      2      150.5
3         AA                A-      2      138.6
.
.
.
21        AA                A-      2      143.6
22        AA                A-      2      166.7
23        Aa                A-      1      147.3
24        Aa                A-      1      147.7
25        Aa                A-      1      148.7
.
.
.
74        Aa                A-      1      149.0
75        Aa                A-      1      145.8
76        Aa                A-      1      149.5
77        aa                aa      0      179.8
78        aa                aa      0      195.9
79        aa                aa      0      182.8
.
.
99        aa                aa      0      175.1
100       aa                aa      0      165.2
```

Et plot av dataene er vist i figur 1.

Vi tilpasser først følgende modell.



Figur 1: Hemoglobinnivået til hvert individ versus variablene `genotype`/`dosage`.

```
> fullmodel <- lm(hemoglobin ~ genotype)
> summary(fullmodel)

Call:
lm(formula = hemoglobin ~ genotype)

Residuals:
    Min      1Q  Median      3Q     Max
-21.910  -8.451  -1.066   7.412  30.870

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  170.954      2.444  69.951  < 2e-16 ***
genotypeAa   -22.974      2.937  -7.822 6.41e-12 ***   (i)
genotypeAA   -19.185      3.534  -5.429 4.19e-07 ***   (ii)
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 11.97 on 97 degrees of freedom
Multiple R-squared:  0.3917,Adjusted R-squared:  0.3792
F-statistic: 31.23 on 2 and 97 DF,  p-value: 3.381e-11
```

e) Skriv opp denne modellen i matematisk notasjon og gjør rede for modellantakelsene. Hvilke ukjente parametere er estimert ovenfor? Hva blir forventet hemoglobinnivå for hver av de

tre genotypene aa, Aa og AA basert på denne modellen? Hvilke hypoteser er testet i linjene merket (i) og (ii) under overskriften "Coefficients"?

Vi tilpasser så en alternativ modell ved i stedet å inkluderer `genotyperecessive` som forklaringsvariabel.

```
> recessive <- lm(hemoglobin ~ genotyperecessive)
> summary(recessive)

Call:
lm(formula = hemoglobin ~ genotyperecessive)

Residuals:
     Min       1Q   Median       3Q      Max
-29.2379  -7.5311   0.2705   7.0902  28.4661

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         170.750      2.302  74.162  < 2e-16 ***
genotyperecessiveA-  -19.196      2.641  -7.268 8.91e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 11.28 on 98 degrees of freedom
Multiple R-squared:  0.3503,Adjusted R-squared:  0.3436
F-statistic: 52.83 on 1 and 98 DF,  p-value: 8.906e-11


> anova(recessive,fullmodel,test="F")
Analysis of Variance Table

Model 1: hemoglobin ~ genotyperecessive
Model 2: hemoglobin ~ genotype
  Res.Df   RSS Df Sum of Sq      F Pr(>F)
1     98 12468
2     97 12391  1    77.048 0.6032 0.4393
```

f) Forklar hvorfor modellen `recessive` er nøstet i modellen `fullmodel` tilpasset ovenfor. Hvilken av de to modellene er å foretrekke ut i fra testresultatene ovenfor?

En tredje alternativ hypotese er at hemoglobin-nivået er lineært påvirket av antall kopier av allelet A som hvert individ bærer (den numeriske forklaringsvariabelen `dosage` i dataframen vist tidligere). Tilpasser vi en slik modell får vi følgende resultat.

```
> linear <- lm(hemoglobin ~ dosage)
> summary(linear)

Call:
```

```
lm(formula = hemoglobin ~ dosage)

Residuals:
     Min       1Q   Median       3Q      Max
-30.8466  -7.4550  -0.7421   9.2804  24.7874

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  165.048      2.204  74.882  < 2e-16 ***
dosage        -9.068      1.850  -4.903 3.75e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 12.54 on 98 degrees of freedom
Multiple R-squared:  0.197,Adjusted R-squared:  0.1888
F-statistic: 24.04 on 1 and 98 DF,  p-value: 3.749e-06


> anova(linear,fullmodel,test="F")
Analysis of Variance Table

Model 1: hemoglobin ~ dosage
Model 2: hemoglobin ~ genotype
  Res.Df   RSS Df Sum of Sq      F    Pr(>F)
1     98 15409
2     97 12391  1    3018.6 23.631 4.495e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

g) Skriv også opp denne modellen i matematisk notasjon. Forklar videre hvordan denne modellen eventuelt er nøstet i modellene `fullmodel` og `recessive` og hvorvidt den er å foretrekke fremfor noen av disse modellene basert på testresultatene. Ser endelig valgt modell ut til å gi mening gitt dataene i figur 1?

h) Anta at vi trekker et nytt tilfeldig valgt individ fra populasjonen ovenfor. Vi får vite at hemoglobinnivået Y i dette individet er målt til å være større enn 170 g/L. Finn sannsynligheten for at dette individet er av genotype *aa* gitt punktestimatene av parameterene til modellen du har valgt ovenfor samt estimatene av populasjonens genotypefrekvenser i punkt b). Hint: Bestem først bl.a. $P(Y > 170|AA)$.

```
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'.

     This is conventionally interpreted as the number of 'successes' in
     'size' trials.

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 <= x],
          otherwise, P[X > x].

Details:

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

                       p(x) = choose(n, x) p^x (1-p)^(n-x)

     for x = 0, ..., 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) >=
     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.

     The length of the result is determined by 'n' for 'rbinom', 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:

     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
     'dnbinom' 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)

------------------------------------------------------------------------
Chisquare               package:stats                R Documentation

The (non-central) Chi-Squared Distribution

Description:

     Density, distribution function, quantile function and random
     generation for the chi-squared (chi^2) distribution with 'df'
     degrees of freedom and optional non-centrality parameter 'ncp'.

Usage:

     dchisq(x, df, ncp = 0, log = FALSE)
     pchisq(q, df, ncp = 0, lower.tail = TRUE, log.p = FALSE)
     qchisq(p, df, ncp = 0, lower.tail = TRUE, log.p = FALSE)
     rchisq(n, df, ncp = 0)

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.

      df: degrees of freedom (non-negative, but can be non-integer).

     ncp: non-centrality parameter (non-negative).

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 chi-squared distribution with 'df'= n >= 0 degrees of freedom
     has density

            f_n(x) = 1 / (2^(n/2) Gamma(n/2))  x^(n/2-1) e^(-x/2)

     for x > 0.  The mean and variance are n and 2n.

     The non-central chi-squared distribution with 'df'= n degrees of
     freedom and non-centrality parameter 'ncp' = lambda has density

     f(x) = exp(-lambda/2)
            * SUM_{r=0}^Inf ((lambda/2)^r / r!) dchisq(x, df + 2r)

     for x >= 0.  For integer n, this is the distribution of the sum of
     squares of n normals each with variance one, lambda being the sum
     of squares of the normal means; further,

     E(X) = n + lambda, Var(X) = 2(n + 2*lambda), and E((X - E(X))^3) =
     8(n + 3*lambda).
```

```
     Note that the degrees of freedom 'df'= n, can be non-integer, and
     also n = 0 which is relevant for non-centrality lambda > 0, see
     Johnson _et al_ (1995, chapter 29).

     Note that 'ncp' values larger than about 1e5 may give inaccurate
     results with many warnings for 'pchisq' and 'qchisq'.

Value:

     'dchisq' gives the density, 'pchisq' gives the distribution
     function, 'qchisq' gives the quantile function, and 'rchisq'
     generates random deviates.

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

     The length of the result is determined by 'n' for 'rchisq', 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.

Note:

     Supplying 'ncp = 0' uses the algorithm for the non-central
     distribution, which is not the same algorithm used if 'ncp' is
     omitted.  This is to give consistent behaviour in extreme cases
     with values of 'ncp' very near zero.

     The code for non-zero 'ncp' is principally intended to be used for
     moderate values of 'ncp': it will not be highly accurate,
     especially in the tails, for large values.

Source:

     The central cases are computed via the gamma distribution.

     The non-central 'dchisq' and 'rchisq' are computed as a Poisson
     mixture central of chi-squares (Johnson _et al_, 1995, p.436).

     The non-central 'pchisq' is for 'ncp < 80' computed from the
     Poisson mixture of central chi-squares and for larger 'ncp' _via_
     a C translation of

     Ding, C. G. (1992) Algorithm AS275: Computing the non-central
     chi-squared distribution function. _Appl.Statist._, *41* 478-482.

     which computes the lower tail only (so the upper tail suffers from
     cancellation and a warning will be given when this is likely to be
     significant).

     The non-central 'qchisq' is based on inversion of 'pchisq'.

References:

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

     Johnson, N. L., Kotz, S. and Balakrishnan, N. (1995) _Continuous
     Univariate Distributions_, chapters 18 (volume 1) and 29 (volume
     2). Wiley, New York.

See Also:

     Distributions for other standard distributions.

     A central chi-squared distribution with n degrees of freedom is
     the same as a Gamma distribution with 'shape' a = n/2 and 'scale'
     s = 2.  Hence, see 'dgamma' for the Gamma distribution.

Examples:

     require(graphics)

     dchisq(1, df = 1:3)
     pchisq(1, df =  3)
     pchisq(1, df =  3, ncp = 0:4)  # includes the above

     x <- 1:10
     ## Chi-squared(df = 2) is a special exponential distribution
     all.equal(dchisq(x, df = 2), dexp(x, 1/2))
     all.equal(pchisq(x, df = 2), pexp(x, 1/2))

     ## non-central RNG -- df = 0 with ncp > 0:  Z0 has point mass at 0!
     Z0 <- rchisq(100, df = 0, ncp = 2.)
     graphics::stem(Z0)

     ## visual testing
     ## do P-P plots for 1000 points at various degrees of freedom
     L <- 1.2; n <- 1000; pp <- ppoints(n)
     op <- par(mfrow = c(3,3), mar = c(3,3,1,1)+.1, mgp = c(1.5,.6,0),
               oma = c(0,0,3,0))
     for(df in 2^(4*rnorm(9))) {
       plot(pp, sort(pchisq(rr <- rchisq(n, df = df, ncp = L), df = df,
            ncp = L)),
            ylab = "pchisq(rchisq(.),.)", pch = ".")
       mtext(paste("df = ", formatC(df, digits = 4)), line =  -2,
             adj = 0.05)
       abline(0, 1, col = 2)
     }
     mtext(expression("P-P plots : Noncentral  "*
                      chi^2 *"(n=1000, df=X, ncp= 1.2)"),
           cex = 1.5, font = 2, outer = TRUE)
     par(op)

     ## "analytical" test
     lam <- seq(0, 100, by = .25)
     p00 <- pchisq(0,      df = 0, ncp = lam)
     p.0 <- pchisq(1e-300, df = 0, ncp = lam)
     stopifnot(all.equal(p00, exp(-lam/2)),
               all.equal(p.0, exp(-lam/2)))

------------------------------------------------------------------------
Multinom                package:stats                R Documentation

The Multinomial Distribution

Description:

     Generate multinomially distributed random number vectors and
     compute multinomial probabilities.

Usage:

     rmultinom(n, size, prob)
     dmultinom(x, size = NULL, prob, log = FALSE)

Arguments:

       x: vector of length K of integers in '0:size'.

       n: number of random vectors to draw.

    size: integer, say N, specifying the total number of objects that
          are put into K boxes in the typical multinomial experiment.
          For 'dmultinom', it defaults to 'sum(x)'.

    prob: numeric non-negative vector of length K, specifying the
          probability for the K classes; is internally normalized to
          sum 1. Infinite and missing values are not allowed.

     log: logical; if TRUE, log probabilities are computed.

Details:

     If 'x' is a K-component vector, 'dmultinom(x, prob)' is the
     probability

      P(X[1]=x[1], ... , X[K]=x[k]) = C * prod(j=1 , ..., K) p[j]^x[j]

     where C is the 'multinomial coefficient' C = N! / (x[1]! * ... *
     x[K]!)  and N = sum(j=1, ..., K) x[j].

     By definition, each component X[j] is binomially distributed as
     'Bin(size, prob[j])' for j = 1, ..., K.

     The 'rmultinom()' algorithm draws binomials X[j] from Bin(n[j],
     P[j]) sequentially, where n[1] = N (N := 'size'), P[1] = p[1] (p
     is 'prob' scaled to sum 1), and for j >= 2, recursively, n[j] = N
     - sum(k=1, ..., j-1) X[k] and P[j] = p[j] / (1 - sum(p[1:(j-1)])).

Value:

     For 'rmultinom()', an integer K x n matrix where each column is a
     random vector generated according to the desired multinomial law,
     and hence summing to 'size'.  Whereas the _transposed_ result
     would seem more natural at first, the returned matrix is more
     efficient because of columnwise storage.

Note:
```

'dmultinom' is currently _not vectorized_ at all and has no C interface (API); this may be amended in the future.

See Also:

Distributions for standard distributions, including 'dbinom' which is a special case conceptually.

Examples:

```
rmultinom(10, size = 12, prob = c(0.1,0.2,0.8))

pr <- c(1,3,6,10) # normalization not necessary for generation
rmultinom(10, 20, prob = pr)

## all possible outcomes of Multinom(N = 3, K = 3)
X <- t(as.matrix(expand.grid(0:3, 0:3))); X <- X[, colSums(X) <= 3]
X <- rbind(X, 3:3 - colSums(X)); dimnames(X) <- list(letters[1:3], NULL)
X
round(apply(X, 2, function(x) dmultinom(x, prob = c(1,2,5))), 3)
```

FDist package:stats R Documentation

The F Distribution

Description:

Density, distribution function, quantile function and random generation for the F distribution with 'df1' and 'df2' degrees of freedom (and optional non-centrality parameter 'ncp').

Usage:

```
df(x, df1, df2, ncp, log = FALSE)
pf(q, df1, df2, ncp, lower.tail = TRUE, log.p = FALSE)
qf(p, df1, df2, ncp, lower.tail = TRUE, log.p = FALSE)
rf(n, df1, df2, ncp)
```

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.

df1, df2: degrees of freedom. 'Inf' is allowed.

ncp: non-centrality parameter. If omitted the central F is assumed.

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 F distribution with 'df1 =' n1 and 'df2 =' n2 degrees of freedom has density

```
f(x) = Gamma((n1 + n2)/2) / (Gamma(n1/2) Gamma(n2/2))
    (n1/n2)^(n1/2) x^(n1/2 - 1)
    (1 + (n1/n2) x)^-(n1 + n2)/2
```

for x > 0.

It is the distribution of the ratio of the mean squares of n1 and n2 independent standard normals, and hence of the ratio of two independent chi-squared variates each divided by its degrees of freedom. Since the ratio of a normal and the root mean-square of m independent normals has a Student's t_m distribution, the square of a t_m variate has a F distribution on 1 and m degrees of freedom.

The non-central F distribution is again the ratio of mean squares of independent normals of unit variance, but those in the numerator are allowed to have non-zero means and 'ncp' is the sum of squares of the means. See Chisquare for further details on non-central distributions.

Value:

'df' gives the density, 'pf' gives the distribution function 'qf' gives the quantile function, and 'rf' generates random deviates.

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

The length of the result is determined by 'n' for 'rf', 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.

Note:

Supplying 'ncp = 0' uses the algorithm for the non-central distribution, which is not the same algorithm used if 'ncp' is omitted. This is to give consistent behaviour in extreme cases with values of 'ncp' very near zero.

The code for non-zero 'ncp' is principally intended to be used for moderate values of 'ncp': it will not be highly accurate, especially in the tails, for large values.

Source:

For the central case of 'df', computed _via_ a binomial probability, code contributed by Catherine Loader (see 'dbinom'); for the non-central case computed _via_ 'dbeta', code contributed by Peter Ruckdeschel.

For 'pf', _via_ 'pbeta' (or for large 'df2', _via_ 'pchisq').

For 'qf', _via_ 'qchisq' for large 'df2', else _via_ 'qbeta'.

References:

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

Johnson, N. L., Kotz, S. and Balakrishnan, N. (1995) _Continuous Univariate Distributions_, volume 2, chapters 27 and 30. Wiley, New York.

See Also:

Distributions for other standard distributions, including 'dchisq' for chi-squared and 'dt' for Student's t distributions.

Examples:

```
## Equivalence of pt(.,nu) with pf(.^2, 1,nu):
x <- seq(0.001, 5, len = 100)
nu <- 4
stopifnot(all.equal(2*pt(x,nu) - 1, pf(x^2, 1,nu)),
          ## upper tails:
          all.equal(2*pt(x,     nu, lower=FALSE),
                    pf(x^2, 1,nu, lower=FALSE)))

## the density of the square of a t_m is 2*dt(x, m)/(2*x)
# check this is the same as the density of F_{1,m}
all.equal(df(x^2, 1, 5), dt(x, 5)/x)

## Identity:  qf(2*p - 1, 1, df)) == qt(p, df)^2)  for  p >= 1/2
p <- seq(1/2, .99, length = 50); df <- 10
rel.err <- function(x, y) ifelse(x == y, 0, abs(x-y)/mean(abs(c(x,y))))
quantile(rel.err(qf(2*p - 1, df1 = 1, df2 = df), qt(p, df)^2), .90)  # ~= 7e
```