

S-PLUS PROGRAM FOR CALCULATING THE ADJUSTED WELCH CONFIDENCE INTERVAL

The adjusted Welch confidence interval is described in section 3 of the following paper:

Kabaila, P. (2005). Assessment of a preliminary F-test solution to the Behrens-Fisher problem. *Communications in Statistics - Theory and Methods*, **34**, 291-302.

The following functions may be used freely, provided that the user makes reference to this paper.

The S-PLUS program to calculate the value of c such that the confidence interval (3.2) of this paper has minimum coverage $1-\alpha$ is

opt.adj.welch

An S-PLUS program to plot the coverage probability of the confidence interval (3.2) for a given value of c is

plot.covpr.welch.adj

This program may be used to check the result obtained from opt.adj.welch.

These programs make use of the following S-PLUS programs which are listed below.

z.to.r

rhs1.welch.adj

rhs2.welch

integrand.welch.adj

covpr.welch.adj

min.covpr.welch.adj

opt.adj.welch

Document name: splus_progs_adj_welch.doc

Date: 6 July 2005

```

z.to.r
function(z, rho, f1, f2)
{
# Written by P.Kabaila for the following paper:
#
# Kabaila, P. (2005). Assessment of a preliminary
# F-test solution to the Behrens-Fisher problem.
# Communications in Statistics - Theory and Methods,
# 34, 291-302.
#
# This function may be used freely, provided the
# user makes reference to this paper.
#
# This function converts z to r.
#
# The notation used in this function is the same as
# in this paper and is as follows.
#
# f1 is a degrees of freedom
# f2 is a degrees of freedom
# rho is sigma1.squared / sigma2.squared
#
#
    num <- rho * f2 * (f2 + 1.)
    denom <- f1 * (f1 + 1.) * ((1./z) - 1.)
    r <- num/denom
    r
}

```

```

rhs1.welch.adj
function(z, rho, f1, f2, alpha, const)
{
# Written by P.Kabaila for the following paper:
#
# Kabaila, P. (2005). Assessment of a preliminary
# F-test solution to the Behrens-Fisher problem.
# Communications in Statistics - Theory and Methods,
# 34, 291-302.
#
# This function may be used freely, provided the
# user makes reference to this paper.
#
# This function calculates
#
#  $c C_{\{\alpha/2\}}(r)$ 
#
# expressed as a function of z.
#
# The notation used in this function is the same as
# in this paper and is as follows.
#
# alpha :- the nominal coverage probability is 1-alpha
# const is a value of c
# dof is the degrees of freedom of the t-distribution
#   calculated using (2.1) of Wang (1971)
# f1 is a degrees of freedom
# f2 is a degrees of freedom
# rho is sigma1.squared / sigma2.squared
#
#
#   r <- z.to.r(z, rho, f1, f2)
#   denom1 <- ((r/(1. + r))^2.)/f1
#   denom2 <- ((1./(1. + r))^2.)/f2
#
#
#   denom <- denom1 + denom2
#   dof <- 1./denom
#   out <- const * qt(1. - (alpha/2.), dof)
#   out
}

```

```

rhs2.welch
function(z, rho, f1, f2)
{
# Written by P.Kabaila for the following paper:
#
# Kabaila, P. (2005). Assessment of a preliminary
# F-test solution to the Behrens-Fisher problem.
# Communications in Statistics - Theory and Methods,
# 34, 291-302.
#
# This function may be used freely, provided the
# user makes reference to this paper.
#
# This function calculates the second term
# making up b_1(z).
#
# The notation used in this function is the same as
# in this paper and is as follows.
#
# alpha :- nominal coverage probability is 1-alpha
# f1 is a degrees of freedom
# f2 is a degrees of freedom
# rho is sigma1.squared / sigma2.squared
#
#
  r <- z.to.r(z, rho, f1, f2)
  num1 <- rho * (f1 + 1.) * (f2 + 1.)
  num2 <- (f1 + f2) * (1. + r)
#
#
  num <- num1 * num2
  denom1 <- rho * (f2 + 1.) + (f1 + 1.)
  denom2 <- r * f1 * (f1 + 1.) + f2 * (f2 + 1.) * rho
  denom <- denom1 * denom2
  out <- sqrt(num/denom)
  out
}

```

```

integrand.welch.adj
function(z, rho, f1, f2, alpha, const)
{
# Written by P.Kabaila for the following paper:
#
# Kabaila, P. (2005). Assessment of a preliminary
# F-test solution to the Behrens-Fisher problem.
# Communications in Statistics - Theory and Methods,
# 34, 291-302.
#
# This function may be used freely, provided the
# user makes reference to this paper.
#
# This function calculates the integrand
# of the integral which satisfies
# coverage probability = 1 - integral
# for the adjusted Welch APDF CI.
# Z has a Beta(f1/2,f2/2) distribution.
#
# The notation used in this function is the same as
# in this paper and is as follows.
#
# alpha is nominal significance level
# const is a value of c
# f1 is a degrees of freedom
# f2 is a degrees of freedom
# rho is sigma1.squared / sigma2.squared
#
  tmp1 <- rhs1.welch.adj(z, rho, f1, f2, alpha, const)
  tmp2 <- rhs2.welch(z, rho, f1, f2)
  rhs <- tmp1 * tmp2
  term1 <- 1. - pt(rhs, f1 + f2)
  term2 <- dbeta(z, (f1/2.), (f2/2.))
  out <- term1 * term2
  out
}

```

```

covpr.welch.adj
function(rho, f1, f2, alpha, const)
{
# Written by P.Kabaila for the following paper:
#
# Kabaila, P. (2005). Assessment of a preliminary
# F-test solution to the Behrens-Fisher problem.
# Communications in Statistics - Theory and Methods,
# 34, 291-302.
#
# This function may be used freely, provided the
# user makes reference to this paper.
#
# This function calculates the coverage
# probability of the adjusted Welch APDF
# CI.
#
# The notation used in this function is the same as
# in this paper and is as follows.
#
# alpha :- nominal coverage probability is 1-alpha
# const is a value of c
# f1 is a degrees of freedom
# f2 is a degrees of freedom
# rho is sigma1.squared / sigma2.squared
#
  newrel.tol <- (.Machine$double.eps^0.25)/10000.
  tmp <- integrate(integrand.welch.adj, 0., 1., keep.xy = T,
    rel.tol = newrel.tol, rho = rho, f1 = f1, f2 = f2,
    alpha = alpha, const = const)
  out <- 1. - 2. * tmp$integral
  out
}

```

```

min.covpr.welch.adj
function(const, f1, f2, alpha, ngrid)
{
# Written by P.Kabaila for the following paper:
#
# Kabaila, P. (2005). Assessment of a preliminary
# F-test solution to the Behrens-Fisher problem.
# Communications in Statistics - Theory and Methods,
# 34, 291-302.
#
# This function may be used freely, provided the
# user makes reference to this paper.
#
# This function calculates the coverage
# probability of the adjusted Welch APDF CI
# for a grid of psi values psigrd to carry
# out an initial search for the minimum of the
# cov. probability(psi), followed by
# a hill-descent search.
#
# The notation used in this function is the same as
# in this paper and is as follows.
#
# alpha :- nominal coverage probability is 1-alpha
# const is a value of c
# delta is used in defining the grid of psi values
# f1 is a degrees of freedom
# f2 is a degrees of freedom
# interval is an interval of rho values
# ngrid is a number of psi gridpoints in the initial
#   grid search
# psi = rho / (1+rho), so that rho = psi / (1-psi)
# psigrd is a grid of psi values
# rho.interval is an interval of rho values corresponding
#   to the interval of psi values interval.
#
# The search is expressed in terms of psi.
#
#
covpr.welch.adj.psi <- function(psi, f1, f2, alpha, const)
{
rho <- psi/(1. - psi)
out <- covpr.welch.adj(rho, f1, f2, alpha, const)
out
}

```

```

}
delta <- 1./(2. * ngrid)
psigrid <- seq(delta, 1. - delta, 2. * delta)
prgrid <- rep(0., ngrid)
for(i in c(1.:ngrid)) {
    psi <- psigrid[i]
    prgrid[i] <- covpr.welch.adj.psi(psi, f1, f2, alpha, const)
}
tmp1 <- order(prgrid)
#
mindex <- tmp1[1.]
if(mindex == 1.)
    interval <- c(1e-009, psigrid[2.])
else if(mindex == ngrid)
    interval <- c(psigrid[ngrid - 1.], 1. - 1e-009)
else interval <- c(psigrid[mindex - 1.], psigrid[mindex + 1.])
tmp <- nlminb(start = psigrid[mindex],
    objective = covpr.welch.adj.psi, f1 = f1,
    f2 = f2, alpha = alpha, const = const, lower = interval[1.],
    upper = interval[2.])
#
#
minimum <- tmp$objective
minimum
}

```

```

opt.adj.welch
function(f1, f2, alpha, ngrid)
{
# Written by P.Kabaila for the following paper:
#
# Kabaila, P. (2005). Assessment of a preliminary
# F-test solution to the Behrens-Fisher problem.
# Communications in Statistics - Theory and Methods,
# 34, 291-302.
#
# This function may be used freely, provided the
# user makes reference to this paper.
#
# This function finds the value of c such that
# the adjusted Welch APDF CI has minimum coverage
# probability equal to 1-alpha.
#
# The notation used in this function is the same as
# in this paper and is as follows.
#
# alpha :- nominal coverage probability is 1-alpha
# f1 is a degrees of freedom
# f2 is a degrees of freedom
# ngrid is the number of psi gridpoints in the initial
#       grid search for the minimum coverage prob.
#
  zero.funct <- function(const, f1, f2, alpha, ngrid)
  {
    tmp <- min.covpr.welch.adj(const, f1, f2, alpha, ngrid)
    out <- tmp - (1. - alpha)
    out
  }
  tmp <- uniroot(zero.funct, lower = 0.5, upper = 2., f1 = f1,
    f2 = f2, alpha = alpha, ngrid = ngrid)
#
#
  out <- tmp$root
  out
}

```

```

plot.covpr.welch.adj
function(psi.grid, f1, f2, alpha, const)
{
# Written by P.Kabaila for the following paper:
#
# Kabaila, P. (2005). Assessment of a preliminary
# F-test solution to the Behrens-Fisher problem.
# Communications in Statistics - Theory and Methods,
# 34, 291-302.
#
# This function calculates the coverage
# probability of the adjusted Welch APDF CI
# for a grid of psi values psi.grid and then plots
# these as a function of psi.
#
# This function finds the value of c such that
# the adjusted Welch APDF CI has minimum coverage
# probability equal to 1-alpha.
#
# The notation used in this function is the same as
# in this paper and is as follows.
#
# alpha is nominal significance level
# f1 is a degrees of freedom
# f2 is a degrees of freedom
#  $\psi = \rho / (1 + \rho)$ , so that  $\rho = \psi / (1 - \psi)$ ,  $\psi$  is in  $[0, 1]$ 
# rho.grid is a grid of  $\sigma_1^2 / \sigma_2^2$  values
#
  rho.grid <- psi.grid/(1. - psi.grid)
  m <- length(rho.grid)
  pr.grid <- rep(0., m)
  for(i in c(1.:m)) {
    pr.grid[i] <- covpr.welch.adj(rho.grid[i], f1, f2, alpha,
      const)
  }
  plot(psi.grid, pr.grid, type = "b", xlab = "psi",
    ylab = "coverage probability(psi)",
    xaxs = "i", xlim = c(0., 1.))
  abline(1. - alpha, 0.)
  mtext(paste("f1=", f1, ", f2=", f2, ", c=", const,
    ", nom. cov. prob.=", 1. - alpha,
    ", min. cov. prob.=", min(pr.grid)))
}

```