R Functions for Statistics

Chris Parrish

May 8, 2015

Contents

one proportion

estimation

check sample-size conditions for CI for one proportion

```
#  check sample-size conditions for CI for one proportion
#  p is unknown, so use p.hat
conditions.ci.one.proportion <- function(p.hat, n){
  category <- c("p.hat * n", "(1 - p.hat) * n")
  count <- c(p.hat * n, (1 - p.hat) * n)
  condition <- count >= 10
  results <- data.frame(category, count, condition)
  return(results)
}
```

confidence interval for a proportion

```
ci.proportion <- function(p.hat, n, alpha){
  z.star <- qnorm(1 - alpha/2)
  se <- sqrt(p.hat * (1 - p.hat) / n)
  ci <- p.hat + z.star * se * c(-1, 1)
  return(list(p.hat=p.hat, z.star=z.star, se=se, ci=ci))
}
```

draw.ci.proportion

```
draw.ci.proportion <- function(a, b, center, ci, title){
  xs <- seq(from=a, to=b, length.out=200)
  ys <- rep(0, 200)
  par(oma=c(0,0,0,2))
  plot(xs, ys,
    col="darkred", type="l", yaxt="n",
    xlab="proportion", ylab="",
    ylim=c(-1.6, 1), main=title)
  lines(x=c(center, center), y=c(-0.5, 0.5), lwd=2, col="orange")
  lines(x=c(ci[1], ci[2]), y=c(0, 0), lwd=3, col="orangered")
  lines(x=c(ci[1], ci[1]), y=c(-0.25, 0.25), lwd=2, col="orangered")
  lines(x=c(ci[2], ci[2]), y=c(-0.25, 0.25), lwd=2, col="orangered")
  text(x=center, y=-1, labels=expression(hat(p)))
  text(x=ci[1], y=-1, labels=expression(hat(p)-z^{"*"}*SE))
  text(x=ci[2], y=-1, labels=expression(hat(p)+z^{"*"}*SE))
}
```

hypothesis test

check sample-size conditions for HT for one proportion

```
#  check sample-size conditions for HT for one proportion
#  p0 is determined by the null hypothesis
conditions.ht.one.proportion <- function(p0, n){
  category <- c("p0 * n", "(1 - p0) * n")
  count <- c(p0 * n, (1 - p0) * n)
  condition <- count >= 10
  results <- data.frame(category, count, condition)
  return(results)
}
```

hypothesis test for a proportion

```
#  alternative = "two.sided", "less", "greater"
ht.proportion <- function(p.hat, n, p0, alternative){
  se <- sqrt(p0 * (1 - p0) / n)
  z <- (p.hat - p0) / se
  if (alternative=="two.sided")
    p.value <- 2 * (1 - pnorm(abs(z)))
  else if (alternative=="greater")
     p.value <- 1 - pnorm(z)
  else if (alternative=="less")
    p.value <- pnorm(z)
  return(list(p.hat=p.hat, se=se, z=z, p.value=p.value))
}
```

draw.normal

We use a function called `normTail` to illustrate regions under the standard normal bell curve. The function is provided by OpenIntro, but we will redefine it so that it just produces an image and not any accompanying numerical data.

```
library(openintro)
```

Key elements in a hypothesis test based on a normal distribution are

- the value of the z test statistic,
- the mean, μ, and standard deviation, σ, of the appropriate normal distribution, and
- the associated p-value.

Write a function called `draw.normal` to illustrate these values. We assume a two-tailed test.

```
draw.normal <- function(x.max, y.max, z, mu, sigma, p.value, title){
  library(openintro)
  normTail(L=-z, U=z,
           ylim=c(-0.1 * y.max, y.max),
```

```
          xlim=c(-x.max, x.max), col="lightblue",
          main=title)
  #  z
  text(x=z, y=-0.03,
       labels=bquote(z == .(z)),
       col="slateblue")
  lines(x=c(z, z), y=c(-0.01, dnorm(z)),
        lwd=2, col="darkred")
  #  p.value
  arrow.x <- max(z + 0.1, 0.8 * x.max)
  text(x=arrow.x, y=0.08, labels=bquote(p == .(p.value)), col="slateblue")
  arrows(x0=arrow.x, y0=0.02, x1=arrow.x, y1=0.06, code=1, angle=10, length=0.1, lwd=1, col="black")
  #  N(mu, sigma^2)
  text(x=0.75 * x.max, y=0.8 * y.max,
       labels=paste("N(", bquote(.(mu)), ", ", bquote(.(sigma))^2, ")", sep=""),
       col="slateblue")
}
```

two proportions

estimation

check sample-size conditions

Check sample-size conditions for CI and HT for the difference of two proportions

```
conditions.two.proportions <- function(p1.hat, n1, p2.hat, n2){
  category <- c("p1.hat * n1", "(1 - p1.hat) * n1",
                "p2.hat * n2", "(1 - p2.hat) * n2")
  count <- c(p1.hat * n1, (1 - p1.hat) * n1,
             p2.hat * n2, (1 - p2.hat) * n2)
  condition <- count >= 10
  results <- data.frame(category, count, condition)
  return(results)
}
```

confidence interval for the difference of two proportions

```
ci.two.proportions <- function(p1.hat, n1, p2.hat, n2, alpha){
  p.hat.diff <- p1.hat - p2.hat
  se <- sqrt(p1.hat * (1 - p1.hat) / n1 + p2.hat * (1 - p2.hat) / n2)
  z.star <- qnorm(1 - alpha/2)
  ci <- p.hat.diff + z.star * se * c(-1, 1)
  return(list(p.hat.diff=p.hat.diff, se=se, z.star=z.star, ci=ci))
}
```

draw.ci.two.proportions

```
draw.ci.two.proportions <- function(a, b, center, ci, title){
  xs <- seq(from=a, to=b, length.out=200)
  ys <- rep(0, 200)
  par(oma=c(0,0,0,2))
  plot(xs, ys,
    col="darkred", type="l", yaxt="n",
    xlab="difference of proportions", ylab="",
    ylim=c(-1.6, 1), main=title)
  lines(x=c(center, center), y=c(-0.5, 0.5), lwd=2, col="orange")
  lines(x=c(ci[1], ci[2]), y=c(0, 0), lwd=3, col="orangered")
  lines(x=c(ci[1], ci[1]), y=c(-0.25, 0.25), lwd=2, col="orangered")
  lines(x=c(ci[2], ci[2]), y=c(-0.25, 0.25), lwd=2, col="orangered")
  text(x=center, y=-1, labels=expression(hat(p)[1]-hat(p)[2]))
  text(x=ci[1], y=-1, labels=expression(hat(p)[1]-hat(p)[2]-z^{"*"}*SE))
  text(x=ci[2], y=-1, labels=expression(hat(p)[1]-hat(p)[2]+z^{"*"}*SE))
}
```

hypothesis test

hypothesis test for the difference of two proportions

```
#  alternative = "two.sided", "less", "greater"

ht.two.proportions <- function(p1.hat, n1, p2.hat, n2, alternative){
  p.hat.diff <- p1.hat - p2.hat
  p.pooled <- (p1.hat * n1 + p2.hat * n2) / (n1 + n2)
  se.diff <- sqrt(p.pooled * (1 - p.pooled) * (1 / n1 + 1 / n2))
  z <- p.hat.diff / se.diff
  if (alternative=="two.sided")
    p.value <- 2 * (1 - pnorm(abs(z)))
  else if (alternative=="greater")
     p.value <- 1 - pnorm(z)
  else if (alternative=="less")
    p.value <- pnorm(z)
  return(list(p.hat.diff=p.hat.diff, se.diff=se.diff, z=z, p.value=p.value))
}
```

one mean

estimation

confidence interval for a mean

The appropriate formula for a confidence interval for a population mean depends on whether the population standard deviation, σ, is known or not, so write two functions corresponding to those two cases.

CI for a mean when sigma is known

```r
ci.mean.sigma.known <- function(x.bar, sigma, n, alpha){
  z.star <- qnorm(1 - alpha/2)
  se <- sigma / sqrt(n)
  ci <- x.bar + z.star * se * c(-1, 1)
  return(list(x.bar=x.bar, z.star=z.star, se=se, ci=ci))
}
```

CI for a mean when sigma is not known

```r
ci.mean.sigma.unknown <- function(x.bar, s, n, alpha){
  t.star <- qt(1 - alpha/2, df=n-1)
  se <- s / sqrt(n)
  ci <- x.bar + t.star * se * c(-1, 1)
  return(list(x.bar=x.bar, t.star=t.star, df=n-1, se=se, ci=ci))
}
```

draw.ci.mean

```r
draw.ci.mean <- function(a, b, center, ci, title){
  xs <- seq(from=a, to=b, length.out=200)
  ys <- rep(0, 200)
  par(oma=c(0,0,0,2))
  plot(xs, ys,
    col="darkred", type="l", yaxt="n",
    xlab="x", ylab="",
    ylim=c(-1.6, 1), main=title)
  lines(x=c(center, center), y=c(-0.5, 0.5), lwd=2, col="orange")
  lines(x=c(ci[1], ci[2]), y=c(0, 0), lwd=3, col="orangered")
  lines(x=c(ci[1], ci[1]), y=c(-0.25, 0.25), lwd=2, col="orangered")
  lines(x=c(ci[2], ci[2]), y=c(-0.25, 0.25), lwd=2, col="orangered")
  text(x=center, y=-1, labels=expression(bar(x)))
  text(x=ci[1], y=-1, labels=expression(bar(x)-t^{"*"}*SE))
  text(x=ci[2], y=-1, labels=expression(bar(x)+t^{"*"}*SE))
}
```

hypothesis test

hypothesis test for a mean

```r
#  alternative = "two.sided", "less", "greater"

ht.mean <- function(x.bar, s, n, mu0, alternative){
  se <- s / sqrt(n)
  t <- (x.bar - mu0) / se
  if (alternative=="two.sided")
    p.value <- 2 * (1 - pt(abs(t), df=n - 1))
  else if (alternative=="greater")
     p.value <- 1 - pt(t, df=n - 1)
  else if (alternative=="less")
```

```
    p.value <- pt(t, df=n - 1)
  return(list(x.bar=x.bar, se=se, t=t, p.value=p.value))
}
```

draw t

We use a function called `normTail` to illustrate regions under the t distribution. The function is provided by OpenIntro, but we will redefine it so that it just produces an image and not any accompanying numerical data.

```
library(openintro)
```

Key elements in a hypothesis test based on a t distribution are

- the value of the t test statistic,
- the degrees of freedom, df, of the appropriate t distribution, and
- the associated p-value.

Write a function called `draw.t` to illustrate these values. We assume a one-tailed test.

```
# U or L; max for U, -max for L

draw.t <- function(x.max, y.max, t, df, p.value, title){
  library(openintro)
  normTail(U=t, df=df,                          # U or L
    ylim=c(-0.1 * y.max, y.max),
    xlim=c(-x.max, x.max), col="wheat",
    main=title)
  #  t
  text(x=t, y=-0.03,
    labels=bquote(t == .(t)),
    col="wheat4")
  lines(x=c(t, t), y=c(-0.01, dt(t, df=df)),
    lwd=2, col="darkred")
  #  p.value
  arrow.x <- max(t + 0.1, 0.8 * x.max)          # max for U, -max for L
  text(x=arrow.x, y=0.08, labels=bquote(p == .(p.value)), col="wheat4")
  arrows(x0=arrow.x, y0=0.02, x1=arrow.x, y1=0.06, code=1, angle=10, length=0.1, lwd=1, col="black")
  #  t(df)
  text(x=0.75 * x.max, y=0.8 * y.max,
    labels=paste("t(df=", bquote(.(df)), ")", sep=""),
    col="wheat4")
}
```

two means

estimation

Welch-Satterthwaite formula for t degrees of freedom

```
t.df <- function(s1, n1, s2, n2){
  v1 <- s1^2 / n1
  v2 <- s2^2 / n2
  numer <- (v1 + v2)^2
  denom <- v1^2 / (n1 - 1) + v2^2 / (n2 - 1)
  df <- floor(numer / denom)
  return(df)
}
```

confidence interval for difference of two means (independent samples)

```
t.ci.two.means <- function(x1.bar, s1, n1, x2.bar, s2, n2, alpha){
  x.bar.diff <- x1.bar - x2.bar
  se.diff <- sqrt(s1^2 / n1 + s2^2 / n2)
  df <- t.df(s1, n1, s2, n2)
  t.star <- qt(c(alpha/2, 1 - alpha/2), df=df)
  ci <- x.bar.diff + t.star * se.diff
  return(list(x.bar.diff=x.bar.diff, se.diff=se.diff, df=df, t.star=t.star, ci=ci))
}
```

confidence interval for difference of two means (paired data)

```
ci.two.means.paired.data <- function(x.d.bar, s.d, n, alpha){
  t.star <- qt(1 - alpha/2, df=n-1)
  se <- s.d / sqrt(n)
  ci <- x.d.bar + t.star * se * c(-1, 1)
  return(list(x.d.bar=x.d.bar, t.star=t.star, df=n-1, se=se, ci=ci))
}
```

draw.ci.two.means

For independent samples or paired data.

```
draw.ci.two.means <- function(a, b, center, ci, title){
  xs <- seq(from=a, to=b, length.out=200)
  ys <- rep(0, 200)
  par(oma=c(0,0,0,2))
  plot(xs, ys,
    col="darkred", type="l", yaxt="n",
    xlab="x", ylab="",
    ylim=c(-1.6, 1), main=title)
```

```
  lines(x=c(center, center), y=c(-0.5, 0.5), lwd=2, col="orange")
  lines(x=c(ci[1], ci[2]), y=c(0, 0), lwd=3, col="orangered")
  lines(x=c(ci[1], ci[1]), y=c(-0.25, 0.25), lwd=2, col="orangered")
  lines(x=c(ci[2], ci[2]), y=c(-0.25, 0.25), lwd=2, col="orangered")
  text(x=center, y=-1, labels=expression(bar(x)[1] - bar(x)[2]))
  text(x=ci[1], y=-1, labels=expression(bar(x)[1] - bar(x)[2] - t^{"*"}*SE))
  text(x=ci[2], y=-1, labels=expression(bar(x)[1] - bar(x)[2] + t^{"*"}*SE))
}
```

hypothesis test

t test for difference of two means (independent samples)

```
#  alternative = "two.sided", "less", "greater"

t.ht.two.means <- function(x1.bar, s1, n1, x2.bar, s2, n2, alternative){
  x.bar.diff <- x1.bar - x2.bar
  se.diff <- sqrt(s1^2 / n1 + s2^2 / n2)
  df <- t.df(s1, n1, s2, n2)
  t <- x.bar.diff / se.diff
  if (alternative=="two.sided")
    p.value <- 2 * (1 - pt(abs(t), df=df))
  else if (alternative=="greater")
     p.value <- 1 - pt(t, df=df)
  else if (alternative=="less")
    p.value <- pt(t, df=df)
  return(list(x.bar.diff=x.bar.diff, se.diff=se.diff, df=df, t=t, p.value=p.value))
}
```

t test for difference of two means (paired data)

```
#  alternative = "two.sided", "less", "greater"

ht.mean.paired.data <- function(x.d.bar, s.d, n, mu0, alternative){
  se <- s.d / sqrt(n)
  t <- (x.d.bar - mu0) / se
  if (alternative=="two.sided")
    p.value <- 2 * (1 - pt(abs(t), df=n - 1))
  else if (alternative=="greater")
     p.value <- 1 - pt(t, df=n - 1)
  else if (alternative=="less")
    p.value <- pt(t, df=n - 1)
  return(list(x.d.bar=x.d.bar, se=se, t=t, p.value=p.value))
}
```

chi-square

univariate categorical data

chi-square goodness-of-fit test

Use `chisq.test`

draw.chisq

Key elements in a χ^2 test are

- the value of the χ^2 test statistic,
- the degree of freedom specifying the appropriate χ^2 distribution, and
- the associated p-value.

Write a function called `draw.chisq` to illustrate these relationships.

```
draw.chisq <- function(x.max, y.max, chisq.val, chisq.df, chisq.p.value, title){
  curve(dchisq(x, df=chisq.df),
    from=-0.1, to=x.max, axes=TRUE,
    col="darkred", xlab="x", n=999,
    ylab="Density", ylim=c(-0.1 * y.max, y.max),
    main=title)
  text(x=0.80 * x.max, y=0.9 * y.max,
    col="orangered",
    labels=bquote(chi^2 * (df == .(chisq.df))))
  abline(h=0)
  lines(x=c(chisq.val, chisq.val),
    y=c(-0.04 * y.max, dchisq(chisq.val, df=chisq.df)), col="orange", lwd=2)
  text(x=chisq.val, y=-0.08 * y.max,
    labels=bquote(chi^2 == .(chisq.val)),
    col="orangered")
  xs <- seq(from=chisq.val, to=x.max, length.out=100)
  ys <- dchisq(xs, df=chisq.df)
  polygon(c(chisq.val, xs, x.max), y=c(0, ys, 0),
    col="peachpuff", border="saddlebrown")
  x.arrow <- 0.85 * x.max
  y.arrow <- 0.15 * y.max
  text(x=x.arrow, y=y.arrow, col="orangered",
    labels=(paste("p-value = ", bquote(.(chisq.p.value)), sep="")))
  arrows(x0=x.arrow, y0=0.2 * y.arrow, x1=x.arrow, y1=0.8 * y.arrow,
    code=1, angle=10, length=0.1, lwd=1, col="sandybrown")
}
```

bivariate categorical data

chi-square test for homogeneity

Compare several populations or treatments (rows) on the basis of a single categorical variable (columns).

Use `chisq.test`

chi-square test for independence

Test two categorical variables for independence.

Use `chisq.test`

regression

For regression, use `lm` and `predict`

For Pearson's sample correlation coefficient, r, use `cor`

correlation.strength

```
correlation.strength <- function(r){
  if (abs(r) < 0.5) {
    strength <- "weak"
  } else if (abs(r) < 0.8) {
    strength <- "moderate"
  } else {
    strength <- "strong"
  }
  return(strength)
}
```

ANOVA

For ANOVA, use `anova`

For multiple comparisons, use `TukeyHSD`

SSTr, SSE, MSTr, MSE, F, p

The following example shows how to do ANOVA "by hand"

memory

reference: Peck, 1/e, 17.11

```
soccer <- c(29.90, 3.73, 86)
non.soccer <- c(30.94, 5.14, 95)
comparison <- c(29.32, 3.78, 53)
data <- data.frame(rbind(soccer, non.soccer, comparison))
names(data) <- c("mean", "s", "n")
data
```

```
##             mean    s  n
## soccer     29.90 3.73 86
## non.soccer 30.94 5.14 95
## comparison 29.32 3.78 53
```

```
grand.mean <- 30.19                # grand mean = T / N
alpha <- 0.05
```

SSTr

```
SSTr <- function(data, grand.mean){
  x.bar <- data[ , 1]
  s <- data[ , 2]
  n <- data[ , 3]
  sstr <- sum(n * (x.bar - grand.mean)^2)
  return(sstr)
}
sstr <- SSTr(data, grand.mean)
```

SSE

```
SSE <- function(data){
  s <- data[ , 2]
  n <- data[ , 3]
  sse <- sum((n - 1) * s^2)
  return(sse)
}
sse <- SSE(data)
```

MSTr, MSE

```
N <- sum(data[ ,3])
k <- nrow(data)
mstr <- sstr / (k - 1)
mse <- sse / (N - k)
F <- mstr / mse
F
```

```
## [1] 2.640205
```

```
p <- 1 - pf(F, df1 = k - 1, df2 = N - k)
p
```

```
## [1] 0.07349915
```

draw.F

Key elements in an F test are

- the value of the F test statistic,
- two degrees of freedom specifying the appropriate F distribution, and
- the associated p-value.

Write a function called `draw.F` to illustrate these relationships.

```
draw.F <- function(x.max, y.max, f.val, f.df1, f.df2, f.p.value, title){
  curve(df(x, df1=f.df1, df2=f.df2),
    from=-0.1, to=x.max, axes=TRUE,
    col="darkred", xlab="x", n=999,
    ylab="Density", ylim=c(-0.1 * y.max, y.max),
    main=title)
  text(x=0.80 * x.max, y=0.9 * y.max,
    col="orangered",
    labels=bquote(paste('F(', df[1] == .(f.df1), ', ', df[2]  == .(f.df2), ')')))
  abline(h=0)
  lines(x=c(f.val, f.val),
    y=c(-0.04 * y.max, df(f.val, df1=f.df1, df2=f.df2)),
    col="orange", lwd=2)
  text(x=f.val, y=-0.08 * y.max,
    labels=bquote(F == .(f.val)),
    col="orangered")
  xs <- seq(from=f.val, to=x.max, length.out=100)
  ys <- df(xs, df1=f.df1, df2=f.df2)
  polygon(c(f.val, xs, x.max), y=c(0, ys, 0),
    col="peachpuff", border="saddlebrown")
  x.arrow <- 0.85 * x.max
  y.arrow <- 0.15 * y.max
  text(x=x.arrow, y=y.arrow, col="orangered",
    labels=(paste("p-value = ", bquote(.(f.p.value)), sep="")))
  arrows(x0=x.arrow, y0=0.2 * y.arrow, x1=x.arrow, y1=0.8 * y.arrow,
    code=1, angle=10, length=0.1, lwd=1, col="sandybrown")
}
```

supplementary

z.star

```
z.star <- function(alpha){
  return(qnorm(1 - alpha / 2))
}
```

t.star

```
z.star <- function(alpha, df){
  return(qt(1 - alpha / 2, df=df))
}
```

required sample size for a proportion

```
required.sample.size.proportion <- function(me, alpha){
  p <- 0.5
  z.star <- qnorm(1 - alpha/2)
```

```
  n <- z.star^2 * p * (1-p) / me^2
  return(ceiling(n))
}
```

pooled standard deviation

```
pooled.standard.deviation <- function(s1, n1, s2, n2){
    numer <- (n1 - 1) * s1^2 + (n2 - 1) * s2^2
    denom <- n1 + n2 - 2
    return(sqrt(numer / denom))
}
```

EXAMPLES

one proportion

estimation

check sample-size conditions

```
p.hat <- 0.21
n <- 20

conditions.ci.one.proportion(p.hat, n)
```

```
##          category count condition
## 1       p.hat * n   4.2     FALSE
## 2 (1 - p.hat) * n  15.8      TRUE
```

confidence interval for a proportion

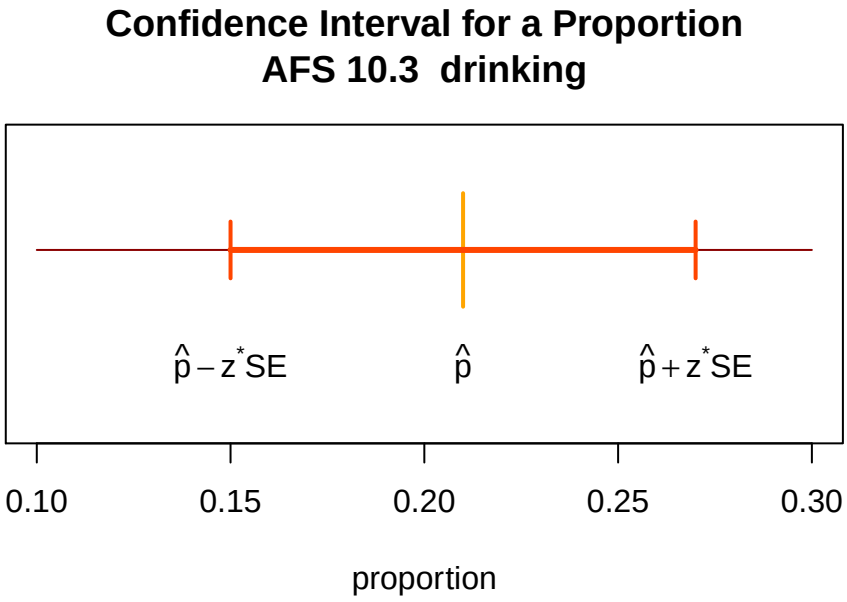
```
p.hat <- 0.21
n <- 40
alpha <- 0.05

ci.proportion(p.hat, n, alpha)
```

```
## $p.hat
## [1] 0.21
##
## $z.star
## [1] 1.959964
##
## $se
## [1] 0.06440109
```

```
##
## $ci
## [1] 0.08377619 0.33622381
```

draw.ci.proportion

```
a <- 0.10
b <- 0.30
center <- 0.21
ci <- c(0.150, 0.270)
title <- "Confidence Interval for a Proportion\nAFS 10.3  drinking"

draw.ci.proportion(a, b, center, ci, title)
```



hypothesis test

hypothesis test for a proportion

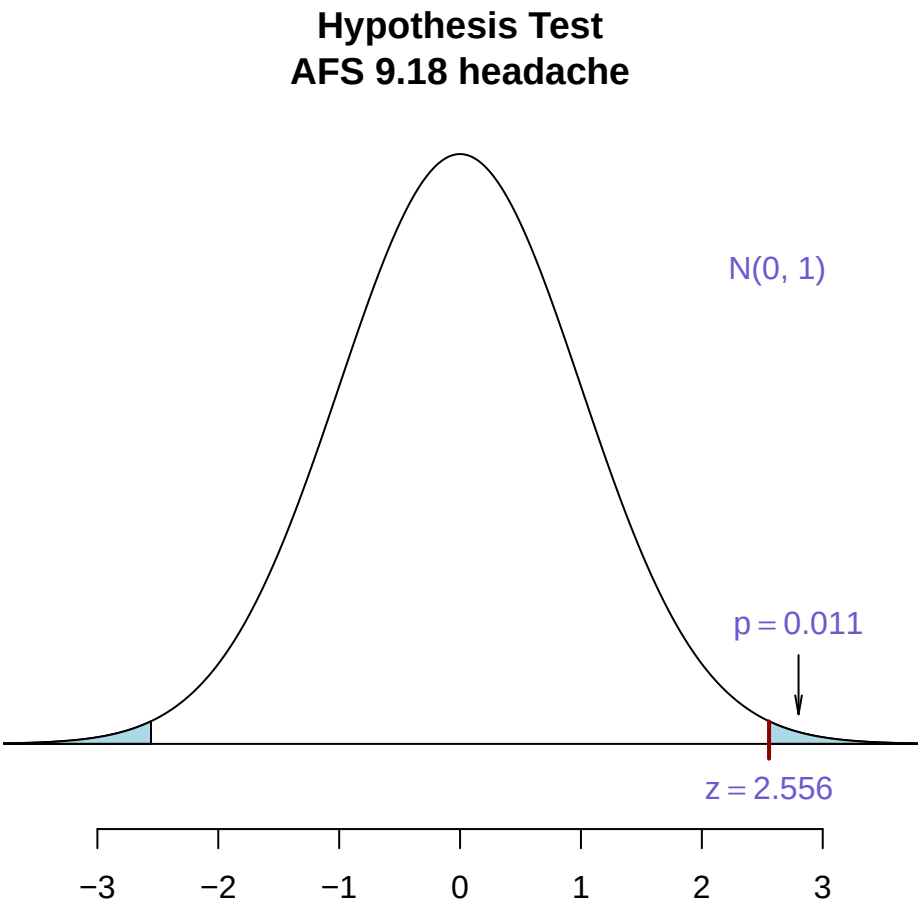
```
x <- 0.23
n <- 20
p0 <- 0.5
alternative <- "two.sided"

ht.proportion(x, n, p0, alternative)
```

```
## $p.hat
## [1] 0.23
##
## $se
## [1] 0.1118034
```

```
##
## $z
## [1] -2.414953
##
## $p.value
## [1] 0.01573722
```

draw.normal

```
x.max <- 3.5
y.max <- 0.4
z <- 2.556
mu <- 0
sigma <- 1
p.value <- 0.011
title <- "Hypothesis Test\nAFS 9.18 headache"

draw.normal(x.max, y.max, z, mu, sigma, p.value, title)
```



two proportions

estimation

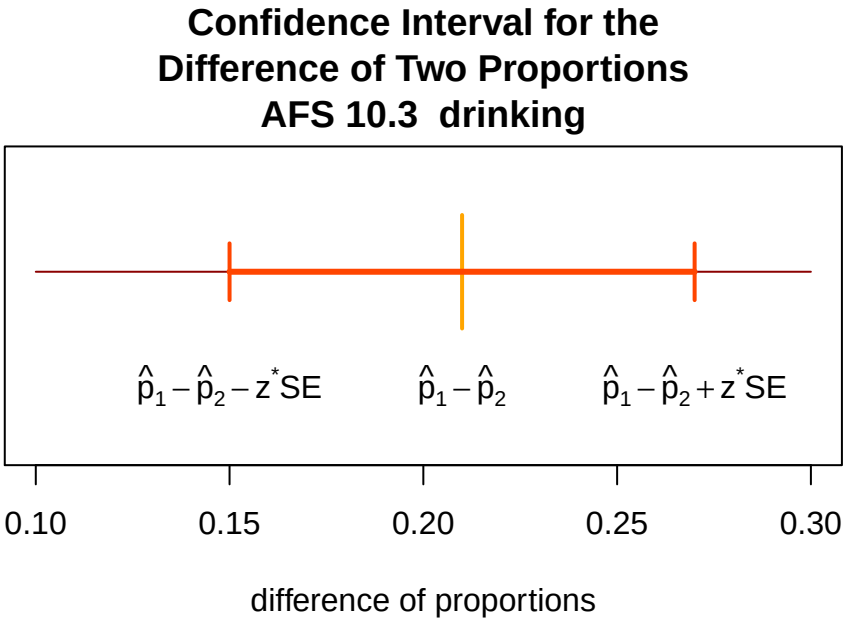
confidence interval for the difference of two proportions

```
#  AFS 10.3 drinking

p1.hat <- 0.206
n1 <- 27603

p2.hat <- 0.256
n2 <- 42000

alpha <- 0.01

ci.two.proportions(p1.hat, n1, p2.hat, n2, alpha)
```

```
## $p.hat.diff
## [1] -0.05
##
## $se
## [1] 0.003234261
##
## $z.star
## [1] 2.575829
##
## $ci
## [1] -0.0583309 -0.0416691
```

draw.ci.two.proportions

```
a <- 0.10
b <- 0.30
center <- 0.21
ci <- c(0.150, 0.270)
title <- "Confidence Interval for the\nDifference of Two Proportions\nAFS 10.3  drinking"

draw.ci.two.proportions(a, b, center, ci, title)
```



hypothesis test

hypothesis test for the difference of two proportions

```
#  AFS 10.8 aspirin

p1.hat <- 347 / 11535
n1 <- 11535
p2.hat <- 327 / 14035
n2 <- 14035
alternative <- "greater"

conditions.two.proportions(p1.hat, n1, p2.hat, n2)
```

```
##            category count condition
## 1       p1.hat * n1   347      TRUE
## 2 (1 - p1.hat) * n1 11188      TRUE
## 3       p2.hat * n2   327      TRUE
## 4 (1 - p2.hat) * n2 13708      TRUE
```

```
ht.two.proportions(p1.hat, n1, p2.hat, n2, alternative)
```

```
## $p.hat.diff
## [1] 0.006783462
##
## $se.diff
## [1] 0.002013327
##
## $z
## [1] 3.36928
```

```
##
## $p.value
## [1] 0.0003768246
```

one mean

estimation

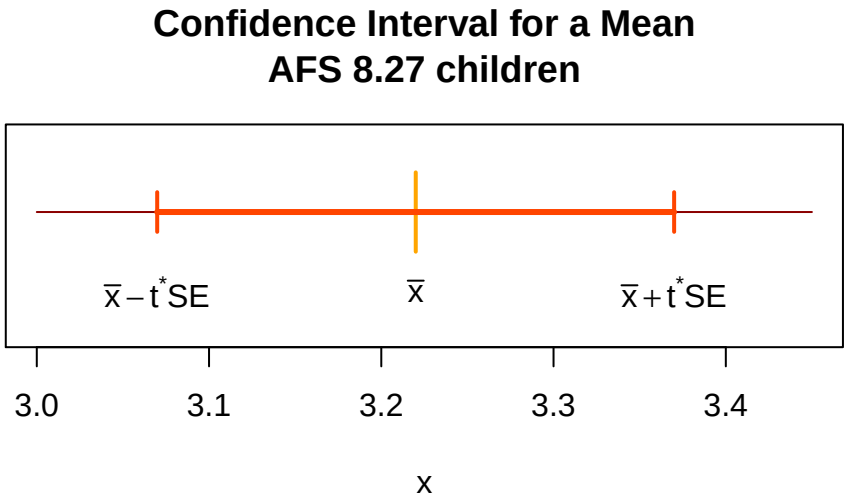
confidence interval for a mean

```
#  AFS 8.27 children

x.bar <- 3.22
s <- 1.99
n <- 678
alpha <- 0.05

ci.mean.sigma.unknown(x.bar, s, n, alpha)
```

```
## $x.bar
## [1] 3.22
##
## $t.star
## [1] 1.963474
##
## $df
## [1] 677
##
## $se
## [1] 0.07642549
##
## $ci
## [1] 3.069941 3.370059
```

draw.ci.mean

```
a <- 3.0
b <- 3.45
center <- 3.22
ci <- c(3.0699, 3.370)
title <- "Confidence Interval for a Mean\nAFS 8.27 children"

draw.ci.mean(a, b, center, ci, title)
```



hypothesis test

hypothesis test for a mean

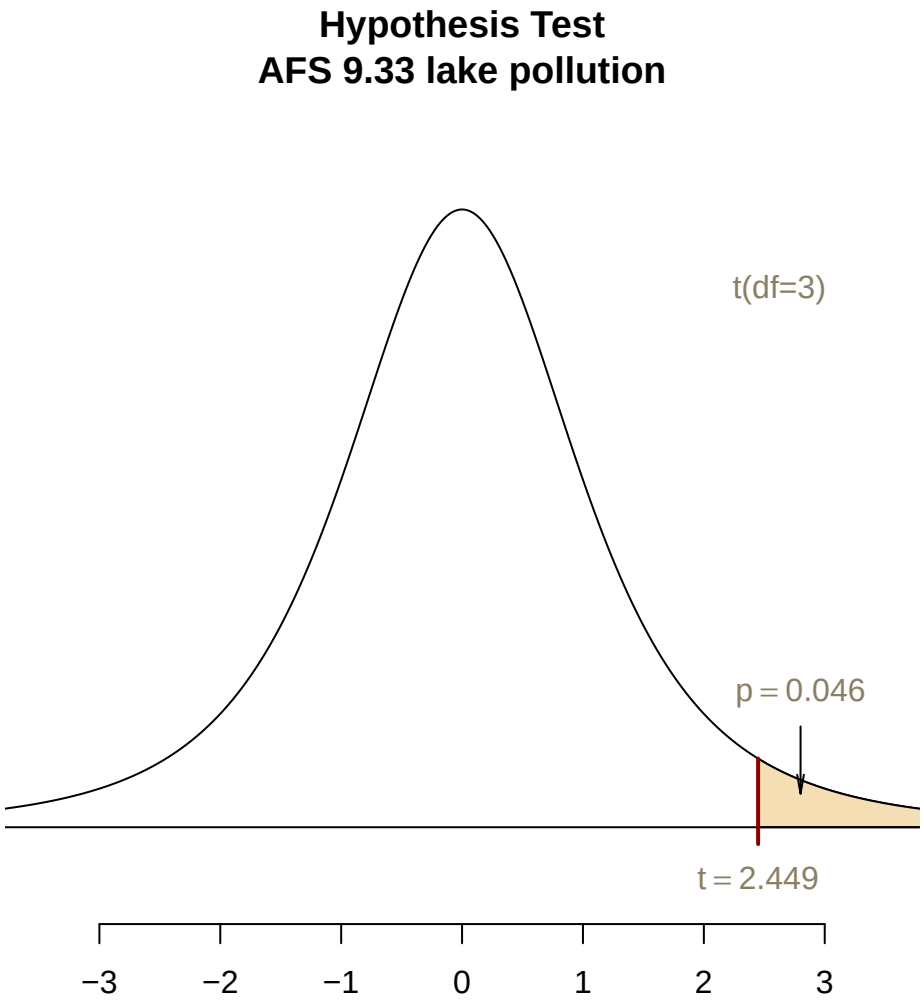
```
#  Peck, 1/e, ex. 12.44
x.bar <- 4.5425
s <- 1.3467
n <- 8
mu0 <- 3.57
alternative <- "greater"

ht.mean(x.bar, s, n, mu0, alternative)
```

```
## $x.bar
## [1] 4.5425
##
## $se
## [1] 0.4761304
##
## $t
## [1] 2.042508
##
## $p.value
## [1] 0.04020735
```

draw t

```
x.max <- 3.5
y.max <- 0.4
t <- 2.449
df <- 3
p.value <- 0.046
title <- "Hypothesis Test\nAFS 9.33 lake pollution"
```

```
draw.t(x.max, y.max, t, df, p.value, title)
```



two means

estimation

Welch-Satterthwaite formula for t degrees of freedom

```
#  AFS 10.25  nicotine

s1 <- 3.6
n1 <- 150

s2 <- 2.9
n2 <- 182

t.df(s1, n1, s2, n2)
```

```
## [1] 284
```

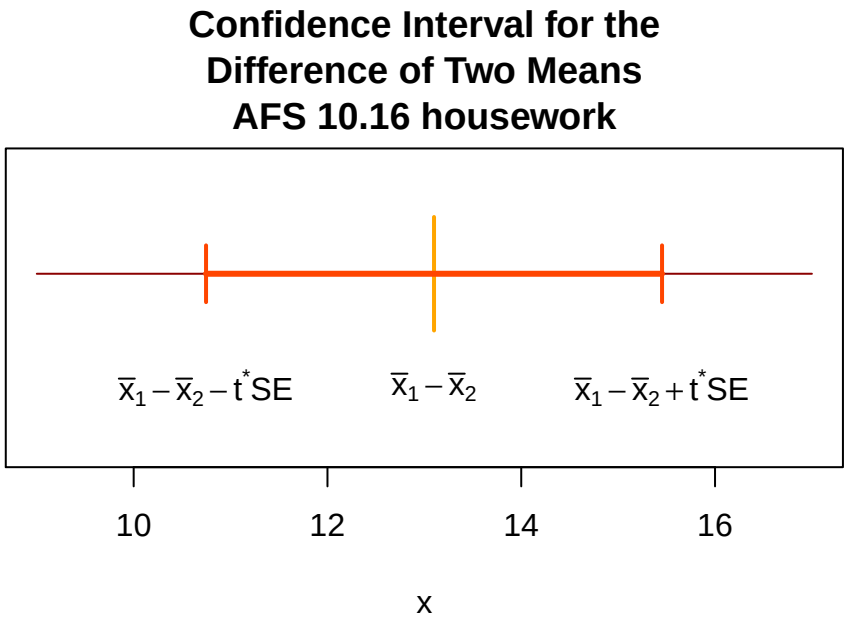
confidence interval for difference of two means

```
#  AFS 10.16  housework

x1.bar <- 33.0
s1 <- 21.9
n1 <- 476

x2.bar <- 19.9
s2 <- 14.6
n2 <- 496

alpha <- 0.05

t.ci.two.means(x1.bar, s1, n1, x2.bar, s2, n2, alpha)
```

```
## $x.bar.diff
## [1] 13.1
##
## $se.diff
## [1] 1.198892
##
## $df
## [1] 822
##
## $t.star
## [1] -1.962854  1.962854
##
## $ci
## [1] 10.74675 15.45325
```

draw.ci.two.means

```
a <- 9.0
b <- 17.0
center <- 13.1
ci <- c(10.747, 15.453)
title <- "Confidence Interval for the\nDifference of Two Means\nAFS 10.16 housework"

draw.ci.two.means(a, b, center, ci, title)
```

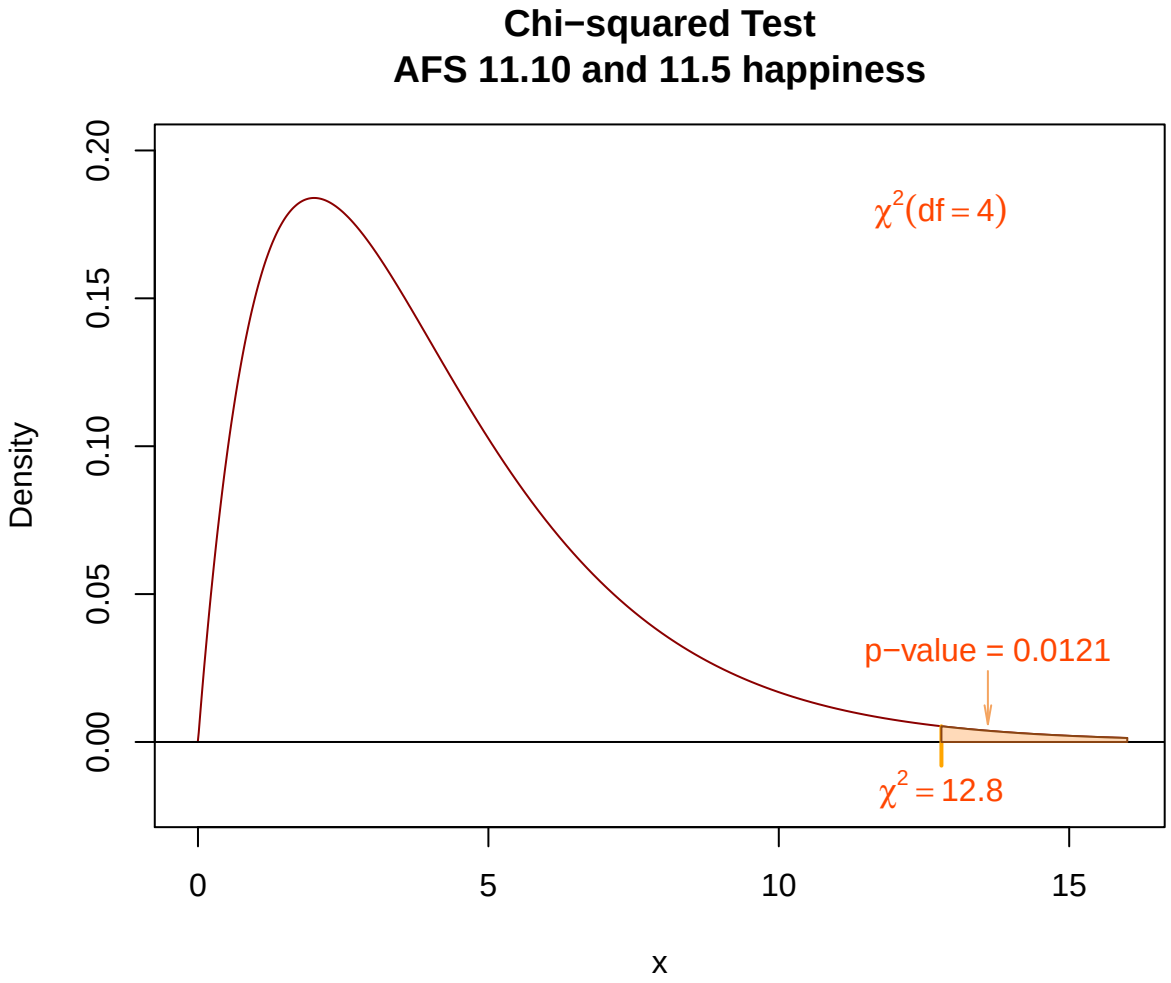


hypothesis test

t test for difference of two means

chi-square

```
x.max <- 16
y.max <- 0.2
chisq.val <- 12.8
chisq.df <- 4
chisq.p.value <- 0.0121
title <- "Chi-squared Test\nAFS 11.10 and 11.5 happiness"

draw.chisq(x.max, y.max, chisq.val, chisq.df, chisq.p.value, title)
```

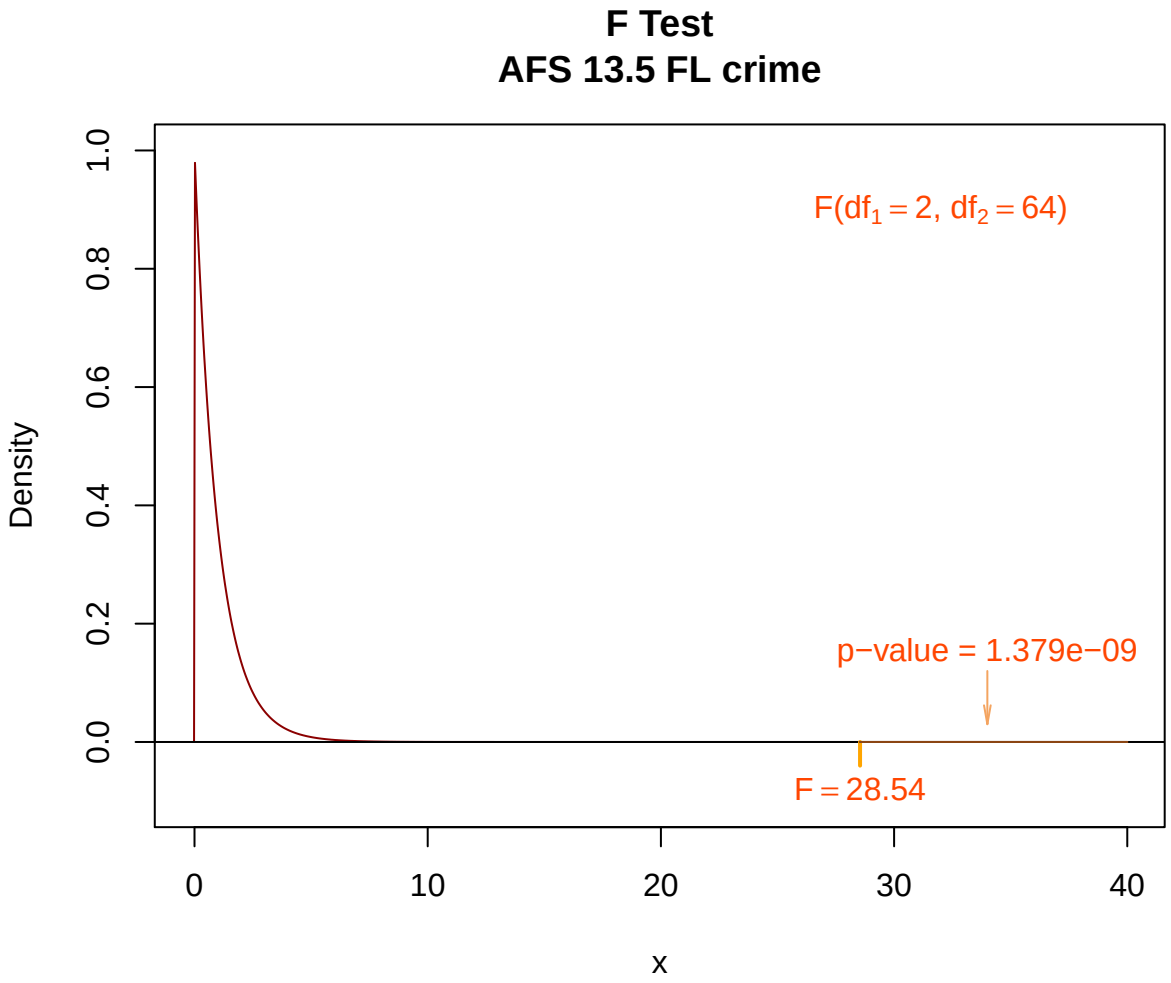


regression

ANOVA

draw.F

```
n <- 67   #  n observations
g <- 3    #  g parameters : b_0, b_1, b_2

x.max <- 40
y.max <- 1.0
f.val <- 28.54
f.df1 <- g - 1
f.df2 <- n - g
f.p.value <- 1.379e-09
title <- "F Test\nAFS 13.5 FL crime"

draw.F(x.max, y.max, f.val, f.df1, f.df2, f.p.value, title)
```



supplementary

z.star

t.star

required sample size for a proportion

pooled standard deviation