t-Test and the Normal Distribution in R

MTH 3220

1 The t-Test

We'll use the following data:

```
data("Puromycin")
Puromycin

##    conc rate     state
## 1  0.02   76   treated
## 2  0.02   47   treated
## 3  0.06   97   treated
## 4  0.06  107   treated
## 5  0.11  123   treated
## 6  0.11  139   treated
## 7  0.22  159   treated
## 8  0.22  152   treated
## 9  0.56  191   treated
## 10 0.56  201   treated
## 11 1.10  207   treated
## 12 1.10  200   treated
## 13 0.02   67 untreated
## 14 0.02   51 untreated
## 15 0.06   84 untreated
## 16 0.06   86 untreated
## 17 0.11   98 untreated
## 18 0.11  115 untreated
## 19 0.22  131 untreated
## 20 0.22  124 untreated
## 21 0.56  144 untreated
## 22 0.56  158 untreated
## 23 1.10  160 untreated

attach(Puromycin)
```

We can perform the *t*-test as follows:

```
t.test(rate ~ state)

##
##  Welch Two Sample t-test
##
## data:  rate by state
## t = 1.6375, df = 19.578, p-value = 0.1175
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -8.504864 70.216985
## sample estimates:
##   mean in group treated mean in group untreated
##                141.5833                110.7273
```

The above output is simply displayed, but if we want to access the resulting values later, we may choose to assign the output to an object:

```
mytest <- t.test(rate ~ state)
mytest$p.value

## [1] 0.1174984

mytest$conf.int

## [1] -8.504864 70.216985
## attr(,"conf.level")
## [1] 0.95
```

You can also run something like `t.test(x, y)` for a two-sample *t*-test where `x` and `y` are the vectors containing the observations of two samples. Check out the help file to see how to change the alternative hypothesis and other arguments. For example, if samples are matched or paired (e.g. before/after), use argument `paired = TRUE`.

2 The Normal Distribution

Apparently R developers don't care for the z-test because there isn't one built in. No worries, we can use the function `pnorm()` to find p-values:

```
mu <- 5
se <- .5
xbar <- 4
z <- (xbar - mu)/se
pnorm(z)

## [1] 0.02275013

## OR use arguments mean and sd
pnorm(xbar, mean = mu, sd = se)

## [1] 0.02275013
```

Check out `?pnorm` for other useful functions regarding the normal distribution. What do you think the function `pt()` is used for?