Statistical Inference week4

Power

the probability of rejecting the null hypothesis when false 1 - beta

beta is 2nd error.

```
## [1] 0.63876
```

```
## [1] 0.05
```

```
## [1] 0.63876
```

```
## Error in FUN(X[[i]], ...): no such symbol rs_createUUID
```

Usually you calculate for beta or n, to make a study, or make sure the study will be meaningful in a certain dataset.

t-test power just use power.t.test (non-central t distruibution)

```
power.t.test(n = 16, delta = 2/4, sd = 1, type = "one.sample", alt = "one.sided")$power
```

```
## [1] 0.6040329
```

```
power.t.test(n = 16, delta = 2, sd = 4, type = "one.sample", alt = "one.sided")$power
```

```
## [1] 0.6040329
```

```
power.t.test(n = 16, delta = 100, sd = 200, type = "one.sample", alt = "one.sided")$power
```

```
## [1] 0.6040329
```

```
power.t.test(power = 0.8, delta = 2/4, sd = 1, type = "one.sample", alt = "one.sided")$n
```

```
## [1] 26.13751
```

```
power.t.test(power = 0.8, delta = 2, sd = 4, type = "one.sample", alt = "one.sided")$n
```

```
## [1] 26.13751
```

```
power.t.test(power = 0.8, delta = 100, sd = 200, type = "one.sample", alt = "one.sided")$n
```

```
## [1] 26.13751
```

you need n = 27 to detect a 0.5 differnece in the mean

Multiple comparisons

type1 : false positive
type2 : false negative

	$\beta = 0$	$\beta \neq 0$	HYPOTHESES
Claim $\beta = 0$	U	T	m-R
Claimβ $\neq$0	V	S	R
Claims	m_0	$m - m_0$	m

- false positive rate:
$$E\left[\frac{V}{m_0}\right]$$
- familiy wise error rate: Pr(V>=1) the probabily of at least one false positive
- false discovery rate :
$$E\left[\frac{V}{R}\right]$$

controlling false positive

Bonferroni correction set
$$\alpha_{fwer} = \alpha/m$$
where m is the number of tests. This is easy to calculate, but may be very conservative.

Controlling false discovery rate

- most popular in genomics, etc.
- do m tests
- order p values from smallest to largest
$$P_{(1)}, \ldots, P_{(m)}$$
- call any
$$P_{(i)} \leq \alpha \times \frac{i}{m}$$
significant
- this will allow for more false positives

Example with 10 p-values

Adjust the p values

- not p values anymore!
- easier than adjusting alpha
- eg. p values P_1, . . . , P_m
$$P_i^{fwer} = max\ m \times P_i, 1$$
for each P-value.
- Then if you call all
$$P_i^{fwer} < \alpha$$
significant you will control the FWER.

```r
set.seed(1010093)
pValues <- rep(NA, 1000)
for(i in 1:1000){
        y <- rnorm(20)
        x <- rnorm(20)
        pValues[i] <- summary(lm(y~x))$coeff[2,4]
}
# control false positive rate
sum(pValues<0.05)
```

```
## [1] 51
```

```r
sum(p.adjust(pValues, method = "bonferroni")<0.05)
```

```
## [1] 0
```

```r
sum(p.adjust(pValues, method = "BH")<0.05)
```

```
## [1] 0
```

```r
set.seed(1010093)
pValues <- rep(NA,1000)
for(i in 1:1000){
        x <- rnorm(20)
        #first 500, beta = 0, last 500 beta = 2
        if(i<=500){y <- rnorm(20)}else{y <- rnorm(20,mean=2*x)}
        pValues[i] <- summary(lm(y~x))$coeff[2,4]
}
trueStatus <- rep(c("zero","not zero"), each = 500)
table(pValues < 0.05, trueStatus)
```

```
##        trueStatus
##         not zero zero
##   FALSE        0  476
##   TRUE       500   24
```

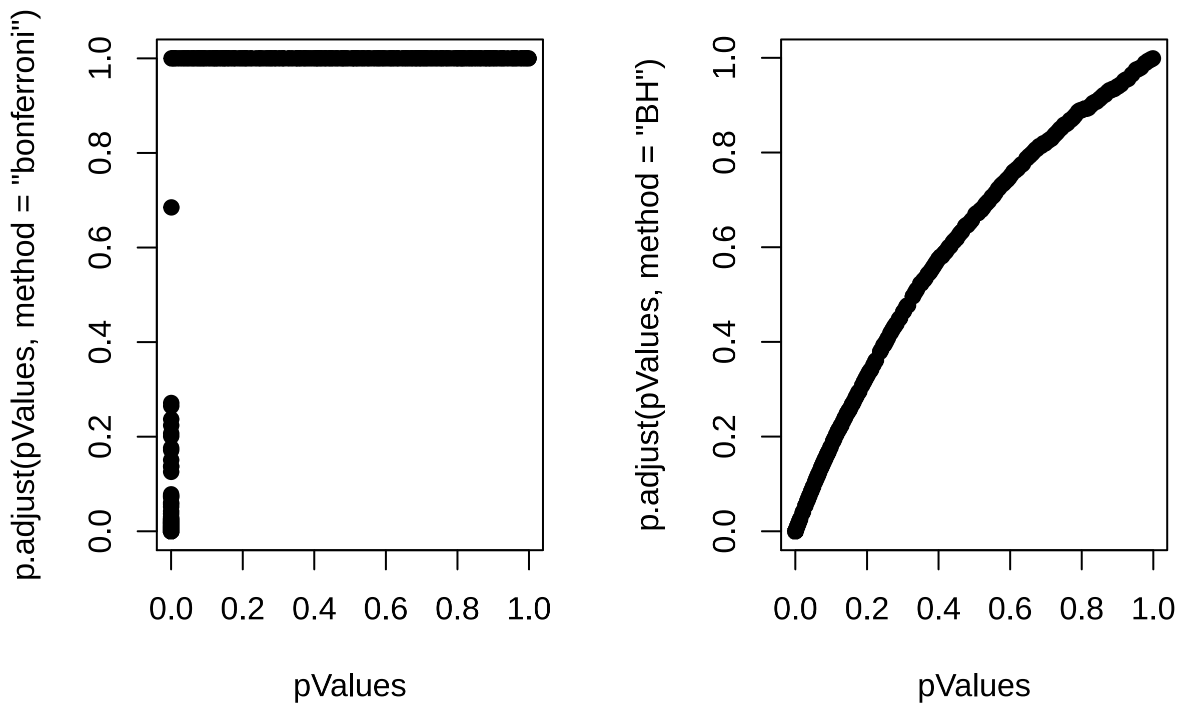
```r
table(p.adjust(pValues, method = "bonferroni") < 0.05, trueStatus)
```

```
##        trueStatus
##         not zero zero
##   FALSE       23  500
##   TRUE       477    0
```

```r
table(p.adjust(pValues, method = "BH")<0.05, trueStatus)
```

```
##        trueStatus
##         not zero zero
##   FALSE        0  487
##   TRUE       500   13
```

```
par(mfrow = c(1,2))
plot(pValues, p.adjust(pValues, method = "bonferroni"), pch = 19)
plot(pValues, p.adjust(pValues, method = "BH"), pch = 19)
```



maybe method = "BY" is good too

Bootstrapping

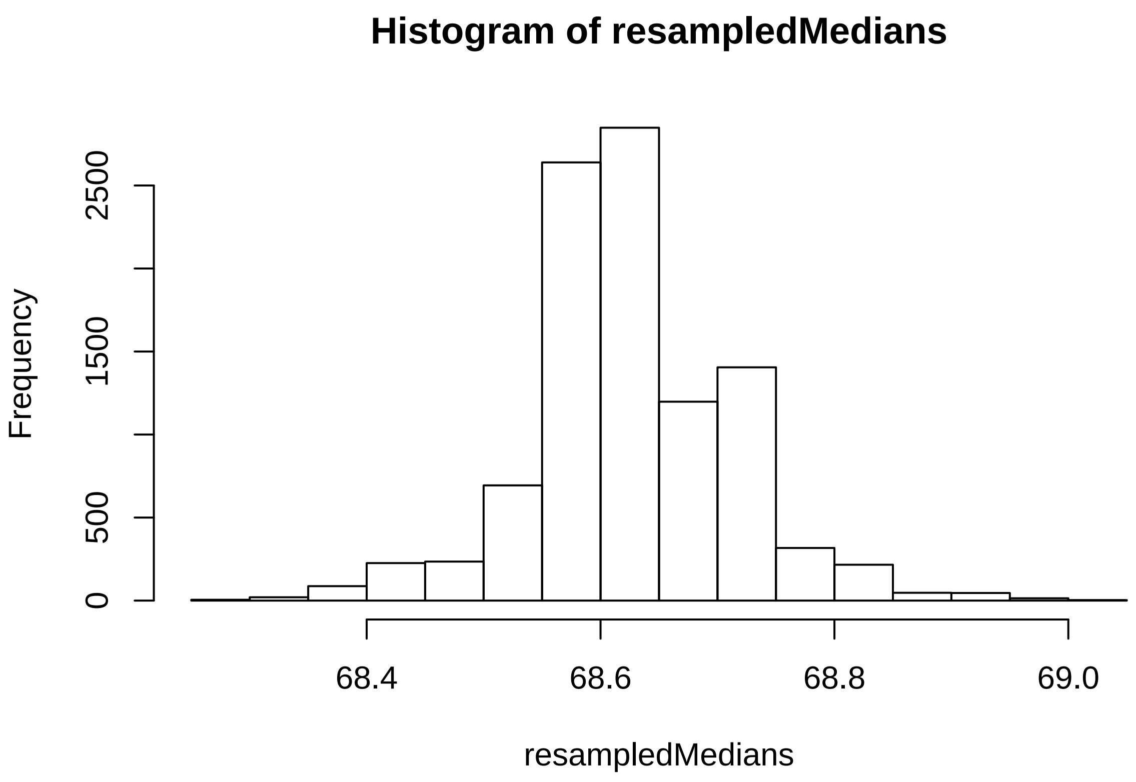
Basic idea: when you have only one distruibution, use that over and over to sample randomly, to figure what kind of charestiristics it has.

```
library(UsingR)
```

```
## Loading required package: MASS
## Loading required package: HistData
## Loading required package: Hmisc
## Loading required package: grid
## Loading required package: lattice
## Loading required package: survival
## Loading required package: Formula
## Loading required package: ggplot2
##
## Attaching package: 'Hmisc'
##
## The following objects are masked from 'package:base':
##
##     format.pval, round.POSIXt, trunc.POSIXt, units
##
##
```

```
## Attaching package: 'UsingR'
##
## The following object is masked from 'package:ggplot2':
##
##     movies
##
## The following object is masked from 'package:survival':
##
##     cancer
```

```r
data(father.son)
x <- father.son$sheight
n <- length(x)
B <- 10000
resamples <- matrix(sample(x, n*B, replace = TRUE), B, n)
resampledMedians <- apply(resamples, 1, median)
hist(resampledMedians)
```



How to do it

1. resample the whole data
2. take the mean
3. do it B times

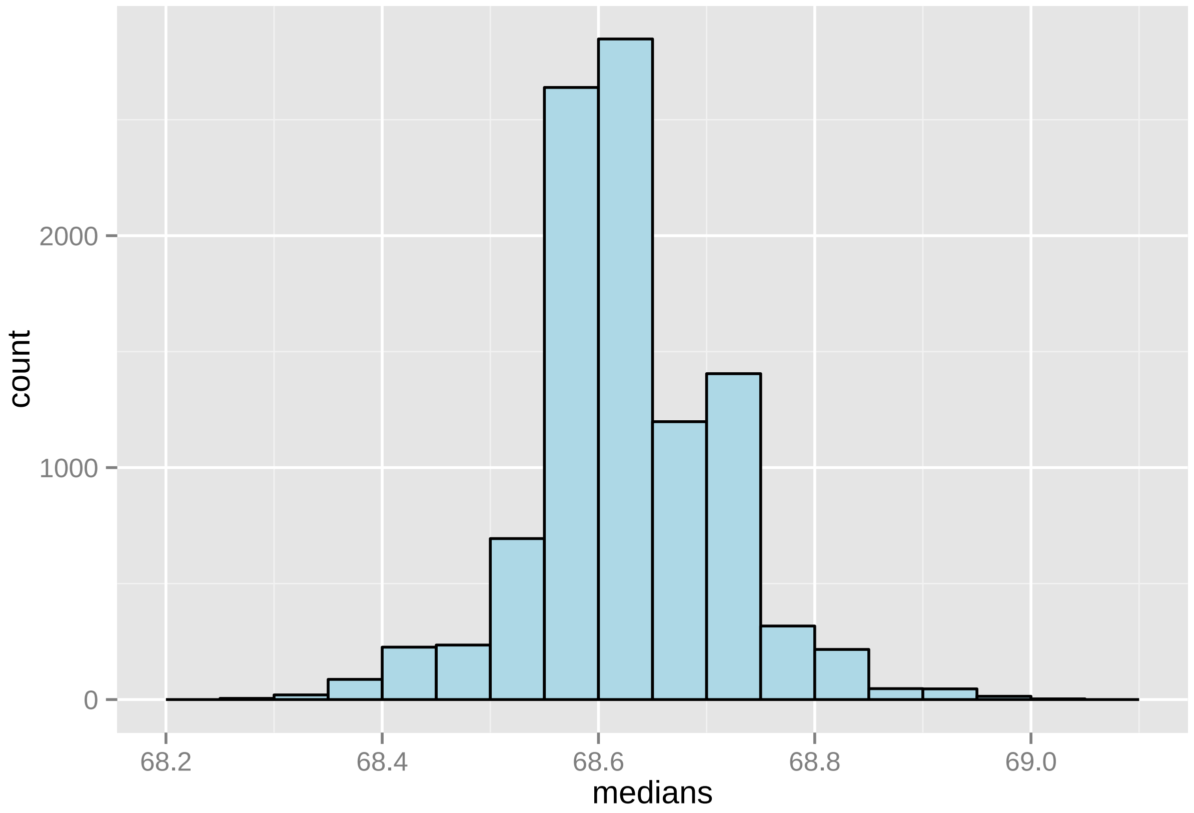
```r
sd(resampledMedians)
```

```
## [1] 0.08473704
```

```
quantile(resampledMedians, c(0.025, 0.975))
```

```
##     2.5%    97.5%
## 68.42972 68.81461
```

```
library(ggplot2)
g = ggplot(data.frame(medians = resampledMedians), aes(x = medians))
g = g + geom_histogram(color = "black", fill = "lightblue", binwidth = 0.05)
g
```



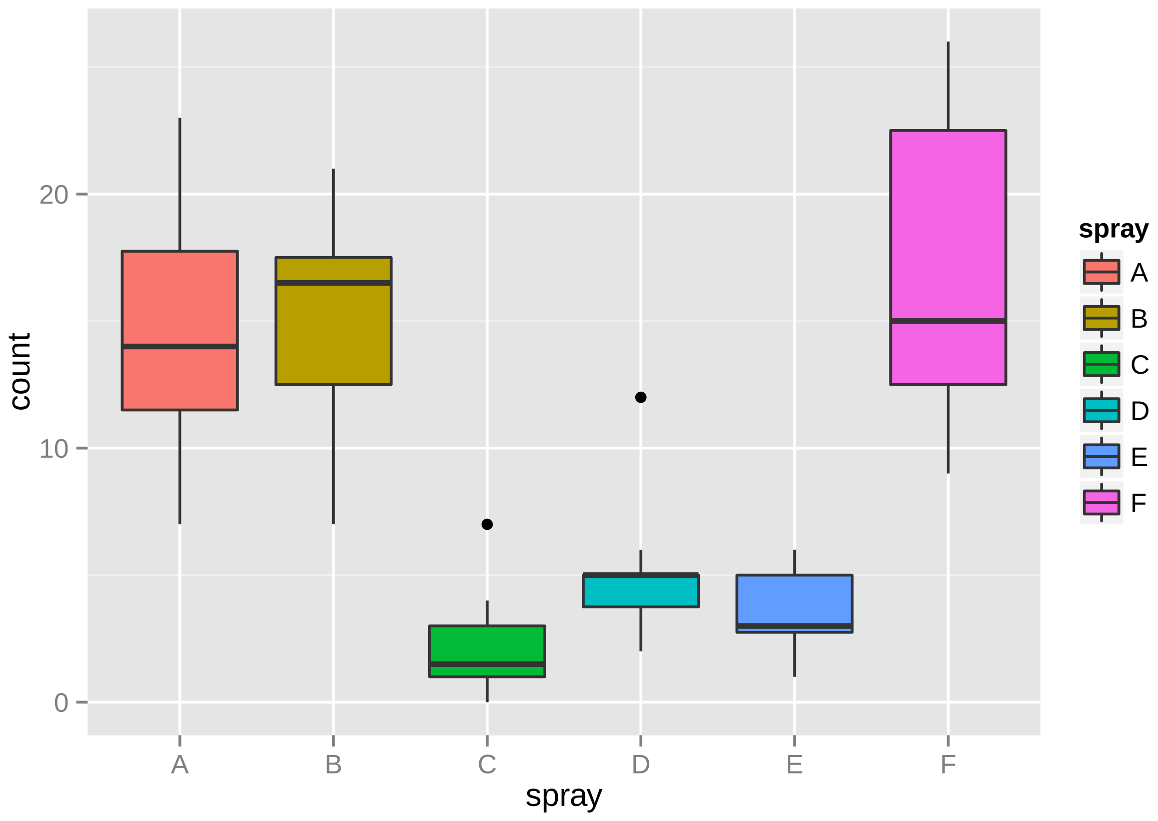
- better to take a bootstrap confidence intervals correct for bias
- "An introduction to bootstrap" is a good place to start

Permutation tests

This is for comparing groups

- take a data frame with groups
- sample without the groups
- was the sample more extreme?

```
data(InsectSprays)
g = ggplot(InsectSprays, aes(spray, count, fill = spray))
g = g + geom_boxplot()
g
```



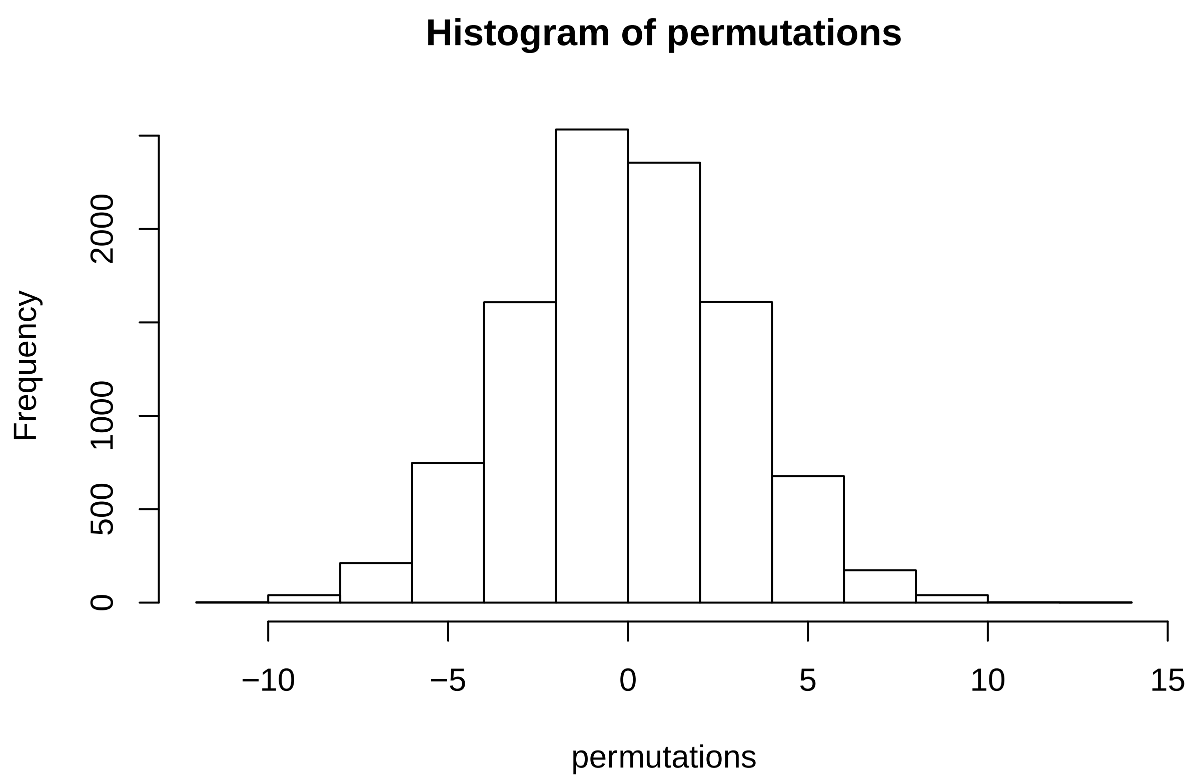
```
subdata <- InsectSprays[InsectSprays$spray %in% c("B","C"),]
y <- subdata$count
group <- as.character(subdata$spray)
teststat <- function(w,g) mean(w[g=="B"]) - mean(w[g=="C"])
observedStat <- teststat(y, group)
permutations <- sapply(1:10000, function(i) teststat(y, sample(group)))
observedStat
```

```
## [1] 13.25
```

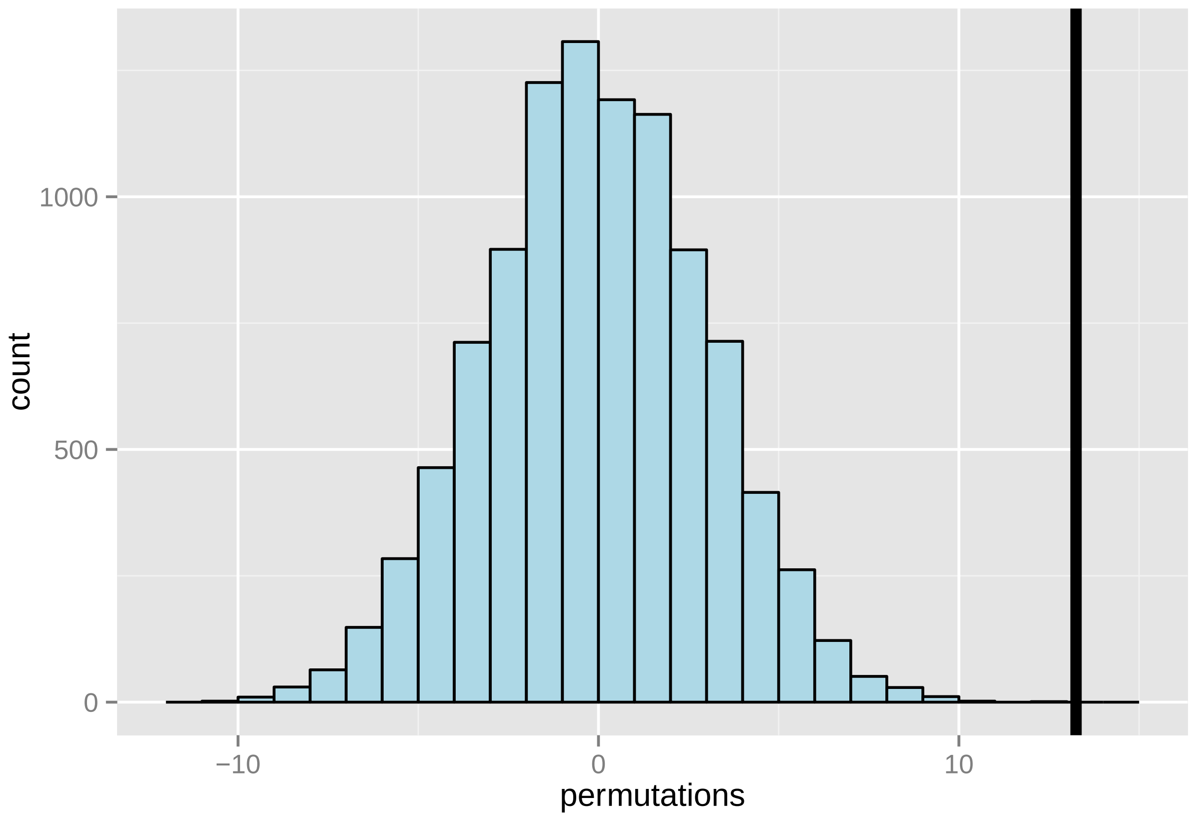
```
mean(permutations > observedStat)
```

```
## [1] 0
```

```
hist(permutations)
```



```
g = ggplot(data.frame(permutations = permutations),
           aes(permutations))
g = g + geom_histogram(fill = "lightblue", color = "black", binwidth = 1)
g = g + geom_vline(xintercept = observedStat, size = 2)
g
```



Quiz

Subject Baseline Week 2

a <- scan() 1 140 132 b <- scan() 2 138 135 c <- scan() 3 150 151 d <- scan() 4 148 146 e <- scan() 5 135 130

data <- rbind(a,b,c,d,e) data <- data.frame(data) names(data) <- c("subject", "baseline", "week2") data t.test(data$baseline - data$week2)

```
n = 9
mean = 1100
sd = 30
mean + c(-1,1)* qt(0.975,n-1) * sd
```

```
## [1] 1030.82 1169.18
```

```
a <- c(0,1,1,1)
t.test(a)
```

```
##
##  One Sample t-test
##
## data:  a
## t = 3, df = 3, p-value = 0.05767
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
```

```
##  -0.04561158  1.54561158
## sample estimates:
## mean of x
##      0.75
```

```
a <- a-1/2
t.test(a)
```

```
##
##  One Sample t-test
##
## data:  a
## t = 1, df = 3, p-value = 0.391
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  -0.5456116  1.0456116
## sample estimates:
## mean of x
##      0.25
```

```
?binom.test
binom.test(3, 4, p = 0.5, alternative = "g")
```

```
##
##  Exact binomial test
##
## data:  3 and 4
## number of successes = 3, number of trials = 4, p-value = 0.3125
## alternative hypothesis: true probability of success is greater than 0.5
## 95 percent confidence interval:
##  0.2486046 1.0000000
## sample estimates:
## probability of success
##                   0.75
```

```
n = 10
d = 1787
b = 1/100
p = n/d
?pnorm
?ppois
ppois(p,b, lower.tail = F)
```

```
## [1] 0.009950166
```

```
poisson.test(n, T = d, r = b, alternative = "l")
```

```
##
##  Exact Poisson test
##
## data:  n time base: d
## number of events = 10, time base = 1787, p-value = 0.03237
## alternative hypothesis: true event rate is less than 0.01
## 95 percent confidence interval:
##  0.000000000 0.009492009
## sample estimates:
##  event rate
## 0.005595971
```