qbs120_ps8_correction_gibran

Gibran Erlangga

11/9/2021

Question 1

a.

```
library(pwr)
```

```
## Warning: package 'pwr' was built under R version 4.1.1
```

```
res = pwr.t.test(d = 0.5, sig.level = 0.05, power = 0.8, alternative = "two.sided")

(n = ceiling(2*res$n))
```

```
## [1] 128
```

b. value of n will be the same because DE genes are assumed to have the same distribution in M0 and M1 tumors for both experiment A and B.
c. my original solution was correct.
d. my original solution was correct.
e. my original solution was correct.
f. my original solution was correct.

Question 2

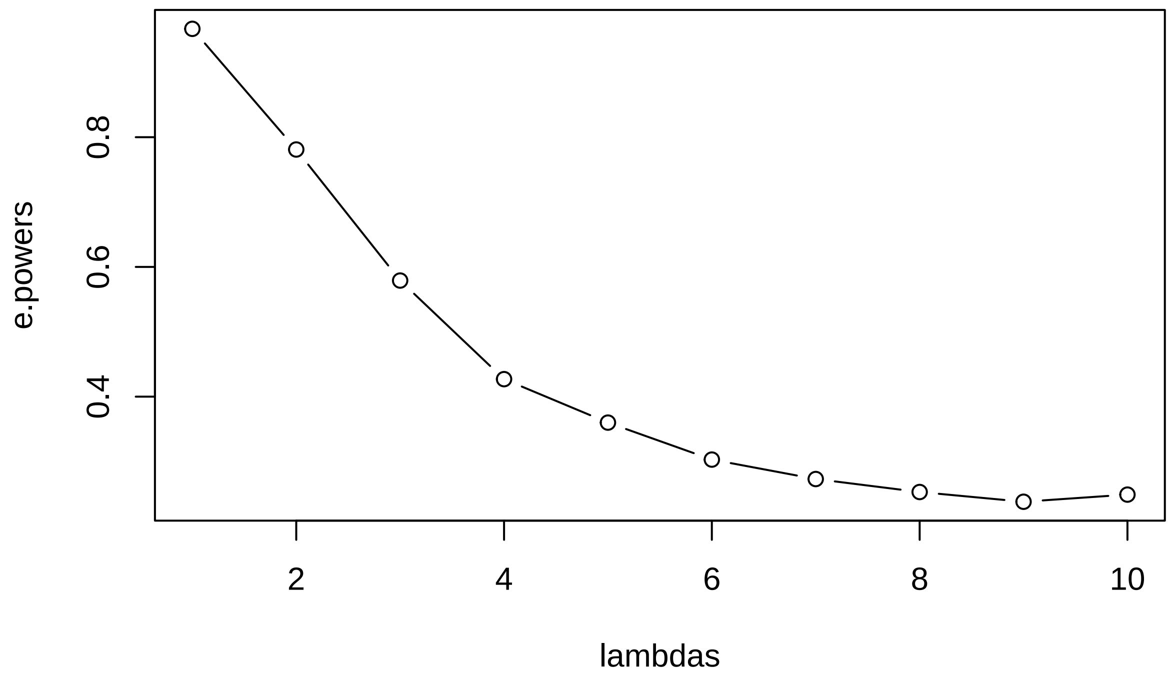
functions from Problem Set 6:

```
# add helper functions
biased.sd = function(x) {
  biased.var = mean((x-mean(x))^2)
  return (sqrt(biased.var))
}

coef.of.skewness = function(x) {
  b.1 = mean((x - mean(x))^3)/biased.sd(x)^3
```

```r
  return (b.1)
}

B = 10000
n = 100

sim.data = matrix(rnorm(B*n), nrow=B, ncol=n)
sim.b.1 = apply(sim.data, 1, coef.of.skewness)
ranked.sim.b.1 = sort(sim.b.1)

simPVal = function(x, ranked.sim.values) {
  n = length(ranked.sim.values)
  smaller.vals =which(ranked.sim.values <= x)
  if (length(smaller.vals) == 0) {
    alpha.low = 0
    } else{
      alpha.low = length(smaller.vals)/n
    }
  larger.vals = which(ranked.sim.values >= x)
  if (length(larger.vals) == 0) {
    alpha.hi = 0
  } else {
    alpha.hi = length(larger.vals)/n
  }
  p.val = 2*min(alpha.low, alpha.hi)
  return(p.val)
}
```

a.

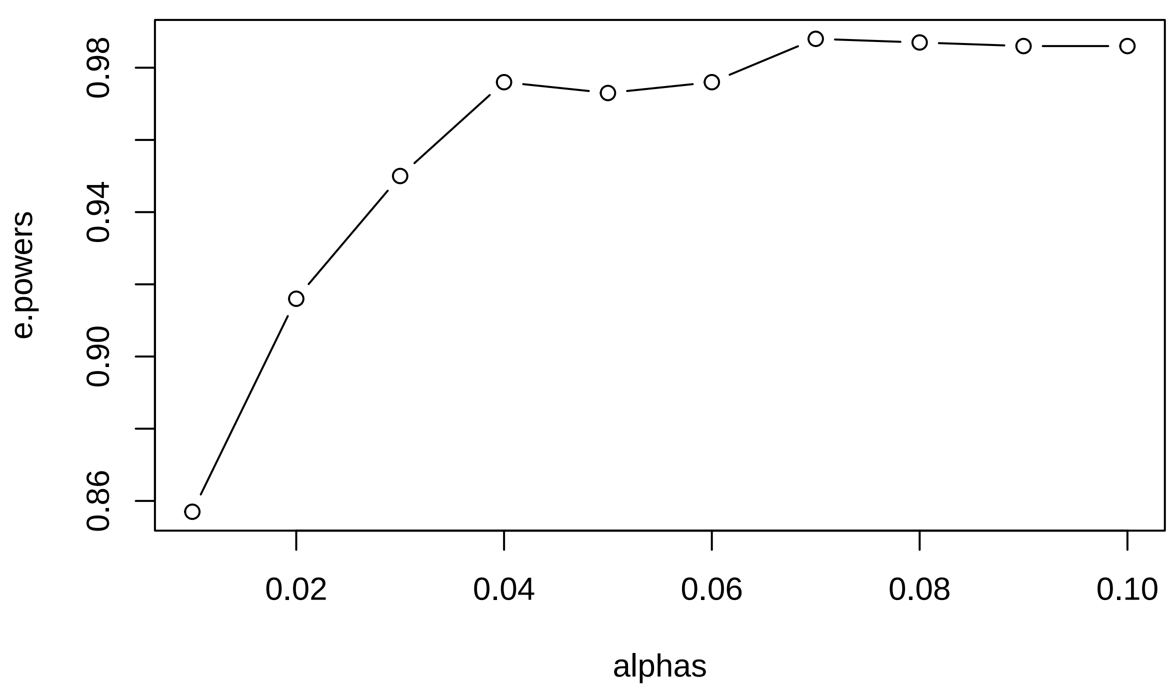
```r
computeEmpiricalPower = function(alpha=0.05, lambda=1, n=1000) {
  test.data = matrix(rpois(n*100, lambda = lambda), nrow=n, ncol=100)
  test.b.1 = apply(test.data, 1, coef.of.skewness)
  p.values = sapply(test.b.1, function(x) simPVal(x, ranked.sim.b.1))
  e.power = length(which(p.values < alpha))/n
  return (e.power)
}

lambdas = 1:10
e.powers = rep(0, 10)
for (i in 1:10) {
  e.powers[i] = computeEmpiricalPower(0.05, lambdas[i])
}

plot(lambdas, e.powers, type="b")
```



as λ increases, the Poisson distribution converges to normal distribution based on CLT - power will decrease.

b.

```
alphas = seq(0.01, 0.1, 0.01)
e.powers = rep(0, 10)
for (i in 1:10) {
  e.powers[i] = computeEmpiricalPower(alphas[i], 1)
}

plot(alphas, e.powers, type="b")
```



as shown above, as α increases, power increases.

Question 3

my original solution was correct.

Question 4

my original solution was correct.

Question 5

my original solution was correct.