Assignment 3. undergraduate

Problem 1.

code:

```r
N = 10000
B = 1000
theta = (qt(0.75, 3)-qt(0.25, 3))/1.34
len_a = vector("numeric", N)
len_b = vector("numeric", N)
len_c = vector("numeric", N)
cov_a = 0
cov_b = 0
cov_c = 0

for (i in 1:N) {
    x = rt(25, 3)
    jack = jackknife(x, function(x){(quantile(x,0.75)-quantile(x,0.25))/1.34})
    ci_a = mean(jack$jack.values)+c(-1,1)*qnorm(0.975)*jack$jack.se
    len_a[i] = ci_a[2] - ci_a[1]
    if(theta > ci_a[1] && theta < ci_a[2]) cov_a = cov_a + 1

    boots = boot(x, function(x, ind) {(quantile(x[ind], 0.75)-quantile(x[ind], 0.25))/1.34}, B,
                 parallel = "multicore", ncpus = 8)
    ci_b = boot.ci(boots, 0.95, "norm")
    len_b[i] = ci_b$normal[3] - ci_b$normal[2]
    if(theta > ci_b$normal[2] && theta < ci_b$normal[3]) cov_b = cov_b + 1

    boots = boot(x, function(x, ind) {(quantile(x[ind], 0.75)-quantile(x[ind], 0.25))/1.34}, B,
                 parallel = "multicore", ncpus = 8)
    ci_c = boot.ci(boots, 0.95, "perc")
    len_c[i] = ci_c$percent[5] - ci_c$percent[4]
    if(theta > ci_c$percent[4] && theta < ci_c$percent[5]) cov_c = cov_c + 1
}
```

```
> mean(len_a)
[1] 1.238188

> mean(len_b)
[1] 1.343335

> mean(len_c)
[1] 1.323244

> cov_a
[1] 8617

> cov_b
[1] 9149

> cov_c
[1] 9751
```

a) coverage for jackknife is 8617/10000

length is 1.238188.

b) Coverage for bootstrap is 9149/10000.

length is 1.343335.

c) Coverage for bootstrap percentile is 9751/10000

length is 1.323244

Conclusion: jackknife gives the smallest length.

bootstrap percentile gives the best coverage.

Problem 2.

a). $\theta = (p, \lambda_1, \lambda_2)$. $X_{com} = (X_{obs}, X_{mis})$

$X_{obs} = x_1, \cdots, x_n$ is the observed data.

$X_{mis} = \begin{cases} 0 & \text{with prob } p. \\ 1 & \text{with prob } (1-p) \end{cases}$

$$f_{mis}(x_{mis}; \theta) = p^{x_{mis}} (1-p)^{1-x_{mis}}$$

$$f_{obs|mis}(x_i | x_{mis}; \theta) = (f(x_i|\lambda_1))^{x_{mis}} (f(x_i|\lambda_2))^{1-x_{mis}}$$

$$L_{com} = \prod_{i=1}^{n} f_{obs|mis}(x_i | x_{mis}) \cdot f_{mis}(x_{mis})$$

$$\ell_{com}(\theta; X_{com}) = \sum_{i=1}^{n} \left[x_{mis} \cdot \log\left(\frac{\lambda_1^{x_i} e^{-\lambda_1}}{x_i!} \right) + (1-x_{mis}) \log\left(\frac{\lambda_2^{x_i} e^{-\lambda_2}}{x_i!} \right) \right]$$

$$+ \sum_{i=1}^{n} \left[x_{mis} \cdot \log(p) + (1-x_{mis}) \log(1-p) \right]$$

E: $Q(\theta; \theta^{(k)}) = E_{mis|obs;\theta^{(k)}}(\ell_{com}(\theta; X_{com}) | x_{obs})$

set $w_i^{(k)} = E_{mis|obs;\theta^{(k)}}[X_{mis} | X_{obs} = x_i]$

$$Q(\theta; \theta^{(k)}) = \sum_{i=1}^{n} w_i^{(k)} [x_i \log(\lambda_1) - \lambda_1 - \log(x_i!)] + ((1-w_i^{(k)}) [x_i \log(\lambda_2) - \lambda_2 - \log(x_i!)]$$

$$+ w_i^{(k)} \log(p) + ((1-w_i^{(k)}) \log(1-p)$$

M: $\hat{p}^{(k+1)} = \frac{\sum_{i=1}^{n} w_i^{(k)}}{n}$.

$$\hat{\lambda}_1^{(k+1)} = \sum_{i=1}^{n} \frac{x_i}{w_i^{(k)}}$$

$$\hat{\lambda}_2^{(k+1)} = \sum_{i=1}^{n} \frac{x_i}{(1-w_i^{(k)})}$$

↓ next page.

$$w_i^{(k)} = E_{mis|obs;\theta^{(k)}}(X_{mis} \mid X_{obs}=x_i) = P(X_{mis}=1 \mid X_{obs}=x_i;\theta^{(k)})$$

$$= \frac{p^{(k)} f(x_i \mid \lambda_1^{(k)})}{p^{(k)} \cdot f(x_i \mid \lambda_1^{(k)}) + (1-p^{(k)}) \cdot f(x_i \mid \lambda_2^{(k)})}$$

repeat EM step until converge.

b)

```r
# hyper-param
p = 0.5
lambda1 = 1
lambda2 = 1

init_value = c(p, lambda1, lambda2)

# obs
x_obs = c(24,18,21,5,5,11,11,17,6,7,20,13,4,16,19,21,4,22,8,17)

P2EM = function(init_value){
    p = init_value[1]
    lambda1 = init_value[2]
    lambda2 = init_value[3]
    n = length(x_obs)
    w = vector("numeric", n)
    for (i in 1:n){
        a = p * dpois(x_obs[i], lambda1)
        b = (1-p) * dpois(x_obs[i], lambda2)
        w[i] = a/(a+b)
    }
    # update hyper-param
    p = sum(w/n)
    lambda1 = sum(x_obs/w)
    lambda2 = sum(x_obs/(1-w))

    return(c(p, lambda1, lambda2))
}

temp = init_value
for (i in 1:10000) {
    temp = P2EM(temp)
}
```

c).

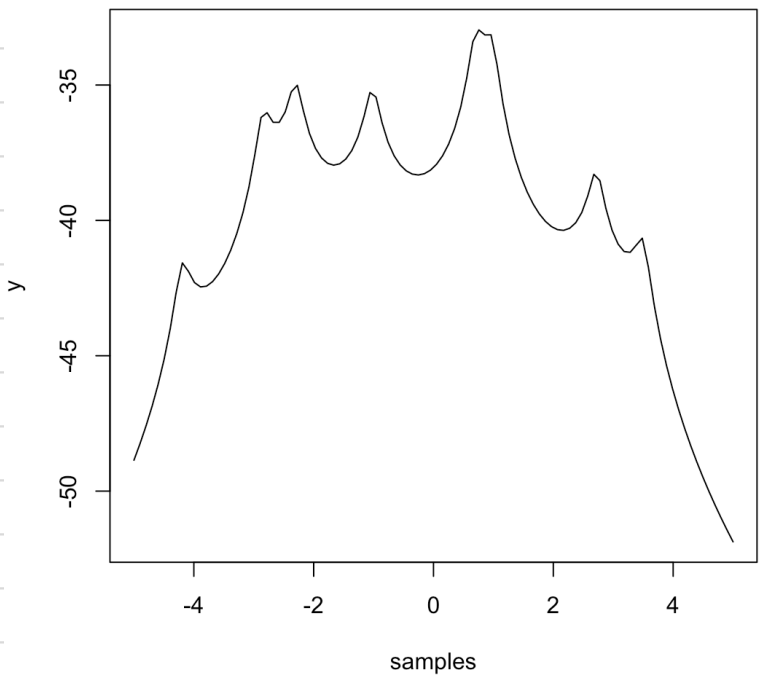
```
temp          num [1:3] 0.5 538 538
```

the function doesn't converge to desire value.

Problem 3.

a).

```r
x = c(-4.2,-2.85,-2.3,-1.02,0.7,0.98,2.72,3.5)
n = length(x)

log_likelihood = function(alpha, beta=0.1) {
    n*log(beta)-n*log(pi)-sum(log(beta^2+(x-alpha)^2))
}

samples = seq(-5,5, length.out = 100)
y = NA
for (i in 1:length(samples)) {
    y[i] = log_likelihood(samples[i])
}

plot(samples, y, 'l')
```



b) set $T_0 = 10$, $T_f = 10^{-7}$, $\Rightarrow T_i = T_0 \cdot \left(\frac{T_f}{T_0}\right)^{(i-1)/n}$ $n = 10^6$

by course note:

```
16 T_fn = function(i) {
17     T0 = 10
18     Tf = 10^-7
19     n = 10^6
20     return(T0*(Tf/T0)^((i-1)/n))
21 }
22 
23 simulated_annealing = function(alpha_init, n) {
24     alpha = alpha_init
25     for (i in 1:n) {
26         alpha_new = rnorm(1, alpha, 1)
27         r = exp((log_likelihood(alpha_new) - log_likelihood(alpha))/T_fn(i))
28         u = runif(1)
29         if (r >= u) {
30             alpha = alpha_new
31         }
32     }
33     return(alpha)
34 }
```

```
> simulated_annealing(-2.5, 10^6)
[1] 0.7327815

> simulated_annealing(0, 10^6)
[1] 0.7328221

> simulated_annealing(-0.30875, 10^6)
[1] 0.7327794
```

they all converge to 0.7328

C)

```
GA_10 = ga("real-valued", log_likelihood, lower = -4, upper = 4, popSize = 10)
GA_20 = ga("real-valued", log_likelihood, lower = -4, upper = 4, popSize = 20)
GA_30 = ga("real-valued", log_likelihood, lower = -4, upper = 4, popSize = 30)
```

```
> GA_10@solution
           x1
[1,] 0.7327702
> GA_20@solution
           x1
[1,] 0.7327716
> GA_30@solution
           x1
[1,] 0.732772
```

by code above. the function converge to 0.7327

Problem 4.

a). $\frac{\partial h}{\partial x_1} = x_1^3 - x_2 + 1 = 0$.

$\Rightarrow$ $(1,2)$ $(0,1)$ $(-1,0)$ are the stationary pts.

$\frac{\partial h}{\partial x_2} = x_2 - x_1 - 1 = 0$.

$\frac{\partial^2 h}{\partial x_1^2} = 3x_1^2$ $\quad$ $\frac{\partial^2 h}{\partial x_1 \partial x_2} = -1$

$\frac{\partial^2 h}{\partial x_2 \partial x_1} = -1$ $\quad$ $\frac{\partial^2 h}{\partial x_2^2} = 1$

for $x = (1,2)$. $H = \begin{bmatrix} 3 & -1 \\ -1 & 1 \end{bmatrix}$ has both eigen value > 0.
$(-1,0)$ ie: PD.

$\therefore$ $x = (1,2), (-1,0)$ are both local min.

for $x = (0,1)$ $H = \begin{bmatrix} 0 & -1 \\ -1 & 1 \end{bmatrix}$ is indefinite. $\lambda_1 = 1.61$
$\lambda_2 = -0.618$.

$\therefore$ $x = (0,1)$ is saddle point.

b). by course note

```r
h = function(x1,x2) {
    1/4*(x1^4) + 1/2*(x2^2) - x1*x2 +x1 - x2
}

h_gradient = function(x1, x2) {
    a = x1^3-x2+1
    b = x2-x1-1
    return(c(a,b))
}


FR_CG = function(init_value, n) {
    x1 = init_value[1]
    x2 = init_value[2]
    temp1 = h(x1,x2)
    p = -h_gradient(x1,x2)
    for (i in 1:n) {
        alpha = 0
        if(!isTRUE(all.equal(p, c(0,0), tolerance = 1e-3))) {
            alpha = optimize(function(x) {h(x1 + x*p[1], x2 + x*p[2])}, c(-20,20))$minimum
        }
        else {break}
        old_x1 = x1
        old_x2 = x2
        x1 = x1 + (alpha*p)[1]
        x2 = x2 + (alpha*p)[2]
        beta = (h_gradient(x1,x2) %*% h_gradient(x1,x2)) /
            (h_gradient(old_x1,old_x2) %*% h_gradient(old_x1,old_x2))
        p = -h_gradient(x1,x2) + c(beta*p[1], beta*p[2])
    }
    return(c(x1,x2))
}
```

```r
FR_CG(c(5,10), 100)
FR_CG(c(-5,-10), 100)

> FR_CG(c(1,2), 100)
[1] 1 2

> FR_CG(c(-1,0), 100)
[1] -1  0

> FR_CG(c(5,10), 100)
[1] 1.000882 2.001866

> FR_CG(c(-5,-10), 100)
[1] -1.001005635 -0.002144307
```

And the result did converge to min depends on the init value.

Problem 5.

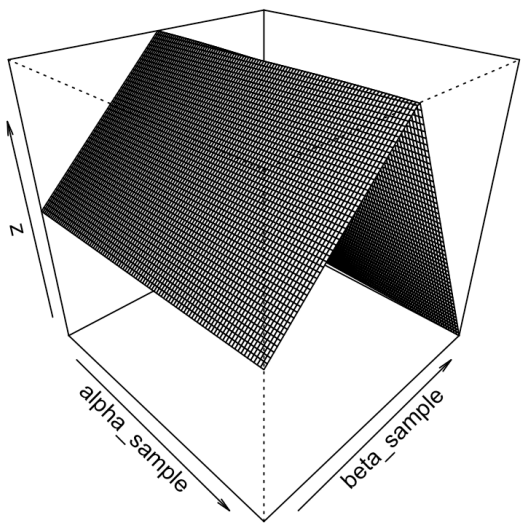
a). $L(\alpha,\beta \mid (x_i,y_i)) = \prod_{i=1}^{n} \left[\frac{e^{\alpha+\beta x_i}}{1+e^{\alpha+\beta x_i}}\right]^{y_i} \left[\frac{1}{1+e^{\alpha+\beta x_i}}\right]^{1-y_i}$

log likelihood: $\ell(\alpha,\beta \mid (x_i,y_i)) = \sum_{i=1}^{n} -\ln(1+e^{\alpha+\beta x_i}) + y_i(\alpha+\beta x_i)$

score: $U(\alpha,\beta,(x_i,y_i)) = \begin{bmatrix} \frac{\partial \ell}{\partial \alpha} \\ \frac{\partial \ell}{\partial \beta} \end{bmatrix} = \begin{bmatrix} \sum\left[-\frac{e^{\alpha+\beta x_i}}{1+e^{\alpha+\beta x_i}} + y_i\right] \\ \sum\left[-\frac{e^{\alpha+\beta x_i}}{1+e^{\alpha+\beta x_i}} \cdot x_i + y_i x_i\right] \end{bmatrix}$

```r
x = c(53,57,58,63,66,67,67,67,68,69,70,70,70,70,72,73,75,75,76,76,78,79,81)
y = c(1,1,1,1,0,0,0,0,0,0,0,0,1,1,0,0,0,2,0,0,0,0,0)

p5log = function(alpha, beta) {
    sum(-log(1+exp(alpha+beta*x)) + y*(alpha+beta*x))
}

alpha_sample = seq(-5, 5, length.out = 100)
beta_sample = seq(-5, 5, length.out = 100)
z = matrix(nrow = length(alpha_sample), ncol = length(beta_sample))

for(i in 1:length(alpha_sample)) {
    for(j in 1:length(beta_sample))
        z[i,j] = p5log(alpha_sample[i], beta_sample[j])
}

persp(alpha_sample, beta_sample, z, theta = 45, phi = 30)
```

log-likelihood.



b)

```
25 p5b = optim(c(0,0), p5log_vector, method= "Nelder-Mead", control=list(fnscale=-1))
26 summary(p5b)
```

```
> summary(p5b)
            Length Class  Mode
par         2      -none- numeric
value       1      -none- numeric
counts      2      -none- numeric
convergence 1      -none- numeric
message     0      -none- NULL
> p5b$par
[1] 11.0608283 -0.1698155
> p5b$value
[1] -12.15341
```

results is (11.06, -0.17)

c)

```
p5c = newtonRaphson(p5log_vector, c(0,0), maxiter = 50000)
summary(p5c)
```

```
p5c           List of 4
 root : num [1:2] -0.039 -0.039
 f.root : num -22.1
 niter : num 50001
 estim.prec: num 0.0432
```

result is (-0.04, -0.04)

which is different from b)
may because the init value.

e)

```
31 p5e = optim(c(0,0), p5log_vector, method= "BFGS", control=list(fnscale=-1))
32 summary(p5e)
```

```
p5e           List of 5
 par : num [1:2] 11.06 -0.17
 value : num -12.2
 counts : Named int [1:2] 97 31
 ..- attr(*, "names")= chr [1:2] "function" "gradi…
 convergence: int 0
 message : NULL
```

result = (11.06, -0.17)
matches result in b)