homework-3

```
library(bis557)
devtools::load_all()
#> Loading bis557
```

Problem 1

Constructing a well-conditioned Hessian along with an ill-conditioned logistic Hessian.

```
X <- c(-3.5, -3, -3.6, 6, 8)/2
X <- cbind(rep(1,5),X)
H <- t(X) %*% X
kappa(H)
#> [1] 7.123973
```

```
betas <- c(0.1,100)
X <- c(-3.5, -3, -3.6, 6, 8)/2
X <- cbind(rep(1,5),X)
mu <- 1 / (1+exp(-X %*% betas))
D <- diag(x=as.vector(mu), nrow=5, ncol=5)
H <- t(X) %*% D %*% X
kappa(H)
#> [1] 863.9918
```

As seen, the D matrix inflates the condition number by more than 100 times.

Problem 2

First generate some random data for testing.

```
n <- 1000
p <- 3
betas <- c(0.2,2,1)
X <- cbind(1, matrix(rnorm(n*(p-1)), ncol=p-1))
mu <- 1 / (1 + exp(-X %*% betas))
y <- as.numeric(runif(n) > mu)
```

I first built a GLM fitting function using gradient descent (without hessian), optimizing log-likelihood.

```
fit <- GLMgradient(X=X,y=y,mu_fun=function(eta) 1/(1+exp(-eta)), T_fun=identity, lrate=0.01,
        maxiter=10000, tol=1e-4)
print(fit)
#>             [,1]
#> [1,] -0.2061319
#> [2,] -1.8153240
#> [3,] -0.9266144
```

The fitting is reasonably good.

Next I use optimization with momentum.

```
fit <- GLMgradientMomentum(X=X,y=y,mu_fun=function(eta) 1/(1+exp(-eta)), T_fun=identity,
        mom=0.2, lrate=0.01, maxiter=10000, tol=1e-4)
print(fit)
#>             [,1]
#> [1,] -0.2091865
#> [2,] -1.8455678
#> [3,] -0.9420365
```

Here the fitting accuracy is a little better than the one with constant step size.

Problem 3

I implemented a softmax regression to generalize logistic regression for classifying more than 2 classes.

First generate some random data for testing.

```
library(ramify)  # call argmax function
#>
#> Attaching package: 'ramify'
#> The following object is masked from 'package:graphics':
#>
#>     clip
n <- 1000
p <- 3
betas <- matrix(c(0.2,2,1,-1,0.2,0.5,0.5,1.0,-0.5), nrow=3, ncol=3)
X <- cbind(1, matrix(rnorm(n*(p-1)), ncol=p-1))
z <- exp(X %*% betas)
z <- z/colSums(z)
y <- argmax(z)
```

Then I fit data using the implemented softmaxreg function.

```
fit <- softmaxreg(X=X,y=y,lrate=0.001, maxiter=10000, tol=1e-4)
print(fit)
#>           [,1]       [,2]       [,3]
#> [1,] 3.5201063  2.2829310  4.1571335
#> [2,] 0.6117602 -1.2743148  0.3160039
#> [3,] 0.5709003  0.7576756 -1.6284941
```

There is an issue.