SCRIPT `MOD2S1B`: MAXIMUM LIKELIHOOD ESTIMATION WITH ANALYTICAL GRADIENT AND HESSIAN

Set basic R-options upfront and load all required R packages:

1. Load and describe data

This is our wage data set from script `mod1_2b`. We originally estimated a CLRM model based on these data via OLS. Now we'll do the same via MLE. Here we use `R` 's built-in `optim` function to find the MLE solution.

Since I saved the data in `R` format last time, I can call it in via the "load" command. To be specific, I saved it as: `save(data, file = "c:/Klaus/AAEC5126/R/data/wage1000.rda")`. This means that the object I saved was "data" (the actual data set), and the destination was workspace "wage1000" in my "c:/.../data/" folder. A workspace is almost like a special `R` folder within a windows folder. It can hold many objects of different type under a single name. Accordingly, the `load` command is as follows:

```
R> load("c:/klaus/AAEC5126/R/data/wage1000.rda")
```

Recall the variable definitions:

Table 1. Variable description

pos.	variable	description
1	wage	hourly wage (1995 dollars)
2	gender	(1= worker = female)
3	race	(1= worker = non-white)
4	union	(1 = worker = unionized)
5	education	years of education
6	experience	years of work experience
7	age	age in years

First, define all components that don't need to be updated during MLE optimization upfront to conserve on computing time.

```
R> names(data)[1]<-"wage"
R> names(data)[2]<-"gender"
R> names(data)[3]<-"race"
R> names(data)[4]<-"unionmember"
R> names(data)[5]<-"education"
R> names(data)[6]<-"experience"
R> names(data)[7]<-"age"
R> attach(data)
R> n<-nrow(data)
R> X<-cbind(rep(1,n),gender,race,unionmember,education,experience)
R> k<-ncol(X)
```

```
R> y<-matrix(log(wage))
R>
```

2. Define Components for Optimization

Next, we will define the log-likelihood function, the analytical gradient, and the Hessian. This function (let's call it `CLRMllfan`) will take a vector of parameters as input (call it `x`) and return the values of the sample log-likelihood, gradient, and Hessian at these parameters.

```
R> CLRMllfan<-function(x,y,X,n,k){
+
+ bm<-x[1:k]
+ sig2<-x[k+1]^2    #square to keep positive
+
+ llf<- -(n/2)*log(2*pi)-(n/2)*log(sig2)-((1/(2*sig2))*t(y-X%*%bm)%*%(y-X%*%bm))
+ #sample log-lh for the CLRM
+
+ #Gradient
+ g1<- (t(X)%*%(y-X%*%bm))/sig2
+ g2<- -(n/(2*sig2))+((t(y-X%*%bm)%*%(y-X%*%bm))/((2*sig2^2)))
+ g<-  rbind(g1,g2)
+
+ #Hessian
+ H1<- -(t(X)%*%X)/sig2
+ H2<- -(t(X)%*%(y-X%*%bm))/(sig2^2)
+ H3<- t(H2)
+ H4<- n/(2*sig2^2)-(t(y-X%*%bm)%*%(y-X%*%bm)/sig2^3)
+ H<-rbind(cbind(H1, H2),cbind(H3, H4))
+
+
+ return (list(llf,g,H))
+ }
```

Now we need starting values (collected in vector `x0` for all of our model parameters, i.e. the slope coefficients and the error variance. We could use the OLS results, but that would make it "too easy" for the MLE optimization routine (since the OLS and MLE solutions are asymptotically equivalent). Instead, I will use "perturbed" OLS starters.

```
R> bols<-solve((t(X)) %*% X) %*% (t(X) %*% y)# compute OLS estimator
R> e<-y-X%*%bols # Get residuals.
R> SSR<-(t(e)%*%e)#sum of squared residuals - should be minimized
R> s2<-(t(e)%*%e)/(n-k) #get the regression error (estimated variance of "eps").
R> Vb<-s2[1,1]*solve((t(X))%*%X)
R> # get the estimated variance-covariance matrix of bols
R> se=sqrt(diag(Vb)) # get the standard erros for your coefficients;
R> tval=bols/se # get your t-values.
R> x0<-0.7*c(bols,s2)
```

3. Optimization

Now we set the optimization parameters and start our optimization routine:

```
R> cri<-10        #initial setting for convergence criterion
R> cri1<-0.0001    #convergence criterion
R> # (here for the sum of the absolute values of the elements in the gradient)
R> maxiter<-1000   #max. number of allowed iterations
R> stsz<-0.1      #step size, something between 0.1 and 1
R> b<-x0
R> jj<-0
R> while ((cri>cri1) & (jj<maxiter))   {
+       jj=jj+1
+
+       int<-CLRMllfan(b,y,X,n,k)
+       llf<-int[[1]]
+       g<-int[[2]]
+       H<-int[[3]]
+
+       cri<-sum(abs(g)) #evaluate convergence criterion
+       db=solve(-H)%*%g; #get directional vector
+
+       b<- b+stsz*db; #update b
+
+       iter<-c(jj, llf, cri)
+       print(iter) #send iteration results to R's command window
+
+       if (jj==maxiter)  {
+       "Maximum number of iterations reached"
+       break
+       }
+
+  } #end of "while"-loop
```

Output analysis and comparison of results:

```
R> bm<-b #this includes sigma
R> sig2<-bm[k+1]^2
R> sem<-sqrt(diag(solve(-H))) #note here we need the negative H
R> tm<- bm/sem
R> ttols<-data.frame(col1=
+   c("constant","gender","race","unionmember","education","experience"),
+                col2=bols,
+                col3=se,
+                col4=tval)
R> colnames(ttols)<-c("variable","estimate","s.e.","t")
R> ttmle<-data.frame(col1=
+   c("constant","gender","race","unionmember","education","experience","sigma"),
+                col2=bm,
+                col3=sem,
+                col4=tm)
R> colnames(ttmle)<-c("variable","estimate","s.e.","t")
```

```
R> ttolsx<- xtable(ttols,caption="OLS output")
R> digits(ttolsx)<-3   #decimals to be shown for each column
R> ttmlex<- xtable(ttmle,caption="MLE output")
R> digits(ttmlex)<-3
R> print(ttolsx,include.rownames=FALSE,
+ latex.environment="center", caption.placement="top",table.placement="!h")
```

TABLE 2. OLS output

variable	estimate	s.e.	t
constant	0.831	0.083	10.067
gender	-0.230	0.030	-7.630
race	-0.140	0.043	-3.274
unionmember	0.163	0.041	3.931
education	0.106	0.005	19.814
experience	0.013	0.001	10.055

```
R> print(ttmlex,include.rownames=FALSE,
+ latex.environment="center", caption.placement="top",table.placement="!h")
```

TABLE 3. MLE output

variable	estimate	s.e.	t
constant	0.831	0.082	10.097
gender	-0.230	0.030	-7.653
race	-0.140	0.043	-3.284
unionmember	0.163	0.041	3.943
education	0.106	0.005	19.873
experience	0.013	0.001	10.085
sigma	0.473	0.010	47.245

The estimated error variance for the OLS model is 0.225.
The estimated error variance for the MLE model is 0.224.
The value of the log-likelihood function at convergence is -670.885

```
R> #save key results:
R> save(b,llf,g,H,
+ file = "c:/Klaus/AAEC5126/R/data/mod2s1b.rda")

R> proc.time()-tic
   user  system elapsed
   2.46    0.17    2.64
```