SCRIPT `MOD3S2B`: HYPOTHESIS TESTING IN MAXIMUM LIKELIHOOD ESTIMATION

INSTRUCTOR: KLAUS MOELTNER

1. Load data & run OLS

This data set comprises 553 observations of NV annual fishing license holders' per-trip expenditures to destinations along the Truckee, Carson, and Walker Rivers in 2005 and 2006.

```
R> data<- read.table('c:/Klaus/AAEC5126/R/data/fishing.txt', sep="\t", header=FALSE)
R> save(data, file = "c:/Klaus/AAEC5126/R/data/fishing.rda")
R> #
R> #assign variable names
R> names(data)[1]<-"runid"
R> names(data)[2]<-"expenditure"
R> names(data)[3]<-"gender"
R> names(data)[4]<-"age"
R> names(data)[5]<-"age2"
R> names(data)[6]<-"fishyrs"
R> names(data)[7]<-"flyfish"
R> names(data)[8]<-"lninc"
R> #
R> attach(data)
```

Variable definitions:

Table 1. Variable description for fishing data

pos.	variable	description
1	runid	runnig id (1-553)
2	expenditure	average per-trip expenditure in 2005 dollars
3	gender	1 = male
4	age	in years
5	age2	age squared
6	fishyrs	years of fishing experience
7	flyfish	1= primarily fly-fishes (as opposed to spin-casting)
8	lninc	log of annual income (2005 dollars)

2. Unconstrained Model

Assume you want to test if experience and income have a different effect on expenditures for fly-fishers than spin casters. This requires the specification of an unconstrained model that allows the coeffcients for these 2 variables to differ across the two groups (similar to a partial Chow test).

We can use the output from this model to perform a Wald test.

2.1. **Data preparation.**

```
R> #prepare data
R> y<-expenditure
R> sel<-y==0
R> y[sel]<-1
R> y<-log(y)
R> n<-nrow(data)
R> fly <-matrix(flyfish,nrow=n )
R> spin<-matrix(1-(flyfish))
R> # Compose X such that the common coefficients come first,
R> # followed by the type-specific stuff
R> int1<-matrix(rep(fly,2),nrow=n)*cbind(fishyrs,lninc)
R> int2<-matrix(rep(spin,2),nrow=n)*cbind(fishyrs,lninc)
R> X<-cbind(rep(1,n),gender,age,age2,fly,int1,int2)
R> k<-ncol(X)
```

2.2. **optimization.** Define function for optimization components (llf, g, H).

```
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))
 #
 #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))
 }
```

Define starting values.

```
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) # 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)
```

Set the optimization parameters and start optimization routine:

```
R> cri<-10        #initial setting for convergence criterion
R> cri1<-0.0001    #convergence criterion
R> maxiter<-1000   #max. number of allowed iterations
R> stsz<-0.1      #step size, something between 0.1 and 1
R> #
R> b<-x0
R> jj<-0
R> #
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
```

Capture output:

```
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> #
R> ttmle<-data.frame(col1=c("constant","gender","age","age2","fly","fishyrs*fly",
                          "lninc*fly","fishyrs*spin","lninc*spin","sigma"),
                col2=bm,
                col3=sem,
                col4=tm)
R> colnames(ttmle)<-c("variable","estimate","s.e.","t")
R> llf_u<-llf #capture llf for unconstrained model for LR test below
```

The estimated error variance for the unconstrained MLE model is 0.58.
The value of the log-likelihood function at convergence is -634.057

Table 2. MLE output, unconstrained model

variable	estimate	s.e.	t
constant	2.524	0.740	3.412
gender	0.059	0.096	0.616
age	0.016	0.015	1.120
age2	-0.000	0.000	-1.849
fly	0.771	1.053	0.733
fishyrs*fly	0.006	0.003	1.882
lninc*fly	0.012	0.071	0.174
fishyrs*spin	0.006	0.003	2.083
lninc*spin	0.082	0.064	1.275
sigma	0.762	0.035	21.834

2.3. **Wald test.** All restrictions are linear, so we can use the ”F-test” approach, though the test statistic is a Wald.

```
R> invH<-solve(-H)
R> Vb<-invH[1:k,1:k]  #we can use any of the ususal estimators for Vb for the Wald test;
R> b<-bm[1:k]                 # here we'll stick to the inverted Hessian
R> Rmat1<-matrix(c(0, 0, 0, 0, 0, 1, 0, -1, 0),nrow=1)
R> Rmat2<-matrix(c(0, 0, 0, 0, 0, 0, 1, 0, -1),nrow=1)
R> Rmat<-rbind(Rmat1,Rmat2)
R> q<- matrix(c(0,0),nrow=2)
R> J<-nrow(Rmat)
R> W<-t(Rmat %*% b-q) %*% solve(Rmat %*% Vb %*% t(Rmat)) %*% (Rmat%*%b-q)
R> pval=1-pchisq(W,J)
```

The Wald-statistic for this test is 0.552. The corresponding p-value is 0.759.

3. Constrained Model

The constrained model will ”force” identical marginal effects of experience and income for both fishing types. We can use the output from this model to perform an LM and an LR test.

3.1. **Data preparation.**

```
R> #prepare data
R> X<-cbind(rep(1,n),gender,age,age2,fly,fishyrs,lninc)
R> k<-ncol(X)
R> #
R> #Define starting values.
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) # 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)
```

Set the optimization parameters and start optimization routine:

```
R> cri<-10        #initial setting for convergence criterion
R> cri1<-0.0001    #convergence criterion
R> maxiter<-1000   #max. number of allowed iterations
R> stsz<-0.1      #step size, something between 0.1 and 1
R> #
R> b<-x0
R> jj<-0
R> #
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
```

Capture output:

```
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> #
R> ttmle<-data.frame(col1=c("constant","gender","age","age2","fly","fishyrs",
                          "lninc","sigma"),
                col2=bm,
                col3=sem,
                col4=tm)
R> colnames(ttmle)<-c("variable","estimate","s.e.","t")
R> llf_c<-llf #capture llf for constrained model for LR test below
```

The estimated error variance for the constrained MLE model is 0.581.
The value of the log-likelihood function at convergence is -634.333

Table 3. MLE output, constrained model

variable	estimate	s.e.	t
constant	2.863	0.573	4.996
gender	0.067	0.095	0.698
age	0.016	0.015	1.107
age2	-0.000	0.000	-1.844
fly	-0.001	0.068	-0.021
fishyrs	0.006	0.003	2.470
lninc	0.051	0.048	1.044
sigma	0.762	0.035	21.823

3.2. **Lagrange Multiplier Test.**

```
R> invH<-solve(-H)
R> Vb<-invH
R> J<-2
R> LM<-t(g) %*% invH %*% g #recall g is the sample gradient
R> pval=1-pchisq(LM,J)
```

The LM-statistic for this test is 0. The corresponding p-value is 1.

3.3. **Likelihood-Ratio test.**

```
R> LR<-2*(llf_u-llf_c)
R> pval=1-pchisq(LR,J)
```

The LR-statistic for this test is 0.5517. The corresponding p-value is 0.7589.

```
R> proc.time()-tic
   user  system elapsed
   0.53    0.04    0.87
```