```
#===========================================================================
 =>
# 1.2 Econometric models using R packages
#===========================================================================
 =>
# R is rich in packages contributed by individuals from all over the world.
 R
# packages comprise a set of functions executing tasks more or less related.
# R packages are easy to write and they offer an ordered way of access

#===========================================================================
 =>
## Installing and using R packages

# Command line code usable at the console to install a package is
# install.packages("package name")

# Let us install the package AER - You need internet connection for this
 step
install.packages("AER")

# Let us load the package into memory
library("AER")

#===========================================================================
 =>
## Ordinary least squares

# This is just a revision of 1.1. Let us reproduce the code here and take a
# closer look at our results. Recall, the essence lies in the
 interpretation/
# meaning of our results. Codes just facilitator.

# a prototypical call of a linear regression looks like fm <- lm(formula,
 data, ...)

# An example from the mroz data:
# Dependent variable - nonwife income
# Independent variables - age, age^2, education, experience
# We allow for a quadratic term to allow for decreasing/increasing returns
# to age

dat<- read.csv("dat.csv",header = T,sep = " ") # load data set

reg1<- lm(dat$nonwife~dat$age + I(dat$age^2)+dat$education+dat$experience)
summary(reg1)

# Interpretation of the results?
#----------------------------------------------->
# An example from the AER package
# load data from the package
data("Journals",package = "AER") #load data Journals into memory from the
#package "AER
names(Journals)
```

```r
# Explore the data
journals <- Journals[, c("subs", "price")] # create a data frame of two
# variables subs, price
journals$citeprice <- Journals$price/Journals$citations # generate new
# variable citeprice and add it to journals. Notice the $ sign
summary(journals) # a summary of the data frame

# The goal is to estimate the effect of the price per citation on the
# number of library subscriptions.

plot(log(subs) ~ log(citeprice), data = journals) # visualise relation-
#ship between variables.

# run a linear regression
jour_lm <- lm(log(subs) ~ log(citeprice), data = journals) # create
# regression object
summary(jour_lm)
abline(jour_lm) # add plot of fitted values to scatter plot

#===========================================================================
 =>
## Binary response models
#---------------------------------------------->
# Probit model
# we begin with a probit regression:
data("SwissLabor") # load data from the AER package
swiss_probit <- glm(participation ~ . + I(age^2),
                   data = SwissLabor, family = binomial(link = "probit"))
# The use of ~. implies regress participation on all other variables in the
# data set then add a quadratic term of age
summary(swiss_probit) # summary of results
#---------------------------------------------->
# visualisation (using a spinogram)
plot(participation ~ age, data = SwissLabor, ylevels = 2:1)
# By setting ylevels =   2:1, the order of participation levels is
# reversed, highlighting participation (rather than non-participation).
#---------------------------------------------->
# Interpretation of results using the average partial effects
favp <- mean(dnorm(predict(swiss_probit, type = "link")))
ape_p = favp * coef(swiss_probit) # these give the average partial effects
ape_p

#--------------------->
# Discussion:
# How would you calculate their standard errors and/or asymptotic
# distributions?
#---------------------------------------------->
# Logit model
swiss_logit <- glm(participation ~ . + I(age^2),
                    data = SwissLabor, family = binomial(link = "logit"))
summary(swiss_logit)
favl <- mean(dlogis(predict(swiss_logit, type = "link")))
ape_l = favl * coef(swiss_logit) # these give the average partial effects
ape_l
```

```
# compare average partial effects:
rbind(ape_p,ape_l)
# what do you say?

#============================================================================
=>
## Regression models for Count data
# These models have discrete dependent variables

# Poisson regression:
# We begin with the standard model for count data, a Poisson regression.
# Fitting is as simple as
#--------------------------------------------->
data("RecreationDemand") # load data from AER package
rd_pois <- glm(trips ~ ., data = RecreationDemand,
                   family = poisson)
# The dependent variable is the number of trips
summary(rd_pois)
# Interpretation of results?

#--------------------------------------------->
# Negative binomial regression
# In R, tools for negative binomial regression are provided by the MASS
# package (Venables and Ripley 2002).
# Let us re-run the count data model using the negative binomial

library("MASS") # this package is in-built. you don't need to install it
rd_nb <- glm.nb(trips ~ ., data = RecreationDemand)
summary(rd_nb)

# compare poisson and negative binomial results:
rbind(rd_pois$coefficients,rd_nb$coefficients)
# observations?

#============================================================================
=>
## Quantile Regression

# The package to use is rq
# install.packages("quantreg")
require("quantreg") # the command require() is synonymous to library()

# To get documentation on a function fun in a package that is installed,
# type ?fun

# look up the function rq in quantreg
?rq
# skim through the documentation and run the example codes.
# you can do so by copying them into the editor and then run them or run
# them straight in the console.
data(stackloss)
rq(stack.loss ~ stack.x,tau=.5)  # LAD is a special case of quantreg
rq(stack.loss ~ stack.x,tau=.25)

#--------------------------------------------->
```

```
# Interpretation of the results?

#==========================================================================
 =>
## Bootstrapping a linear regression
# Conventional regression output relies on asymptotic approximations to the
# distributions of test statistics, which are often not very reliable in
# small samples or models with substantial nonlinearities. A possible remedy
# is to employ bootstrap methodology.

# For ease of reference, we reproduce the basic regression
data("Journals")
journals <- Journals[, c("subs", "price")]
journals$citeprice <- Journals$price/Journals$citations
jour_lm <- lm(log(subs) ~ log(citeprice), data = journals)

# Let us write a function to bootstrap

refit <- function(data, i) coef(lm(log(subs) ~ log(citeprice), data =
 data[i,]))

library("boot") #load package for bootstapping
set.seed(123) # set seed for reproducibility
jour_boot <- boot(journals, refit, R = 999)

# A comparison with the standard regression output
coeftest(jour_lm)

boot.ci(jour_boot, index = 2, type = "basic") # compute confidence intervals

# while its classical counterpart is
confint(jour_lm, parm = 2)
# How do they compare?
#==========================================================================
 =>

# Exercises:
# use data set "women" in the pre-loaded package "datasets"
# Use "height" as the outcome and "weight" as the covariate

# 1. Run a poisson regression: outcome - floor(height), covariate - weight
# 2. Run a negative binomial regression : outcome - floor(height), covariate
 - weight
# 3. Run a quantile regression model at the following quantiles
#    tau = c(0.2, 0.4, 0.5, 0.6, 0.8)
# 4. Run the gamma regression  of "height" on "weight".
# Hint: use the "Gamma" family in glm( )
```