Department of Economics & Finance
University of Guelph
Econ 3740 Introduction to Econometrics
Instructor: D Prescott

R-Notes: Conditional Mean Plots

Define a Function to Compute Data for Plots

Task 2 of Report I includes a plot of the conditional mean of house prices for alternative size categories. A sequence of sample conditional means is a nonparametric estimate of the population mean regression of price given size.
The Chart will also include a plot of the conditional median price.

The recommended method is to create a function in R that will calculate the conditional means and medians for a series of size categories or bins. Such a function is described below – the function is called "conmeanp" which is short for "conditional means of price", although the function computes more than just conditional means.

```
conmeanp <- function(data,minsize,width,step,nsteps) {
left <- minsize
x <- matrix(nrow=nsteps,ncol=5)
for (i in 1:nsteps) {
bin <- subset(data,size>=left & size<left+width)
x[i,1] <- mean(bin$size)
x[i,2] <- mean(bin$price)
x[i,3] <- median(bin$price)
x[i,4] <- var(bin$price)^0.5
x[i,5] <- length(bin$size)
left=left+step
}
return(x)
}
```

The first line defines the function name and its 5 arguments. The first argument is the name of the dataset (use the name of the data frame that you have attached in the current session.) Note the first line ends with "{" which opens parentheses that will contain the details of the function.

The second argument is the minimum house size that will define the left side of the first size bin. It will be be something like 700 or 800 square feet. This argument is used in the second line to define the variable "left". The third argument "width" is the width of the size bins that are used to compute conditional means and conditional medians. The fourth argument is "step" which is the distance the bins

are shifted rightwards. The fifth argument, "nsteps" is the number of steps i.e., the total number of bins or points that will be plotted. In R, a typical application of the function would look like this:

```
x <- conmeanp(mls,800,200,50,40)
```

where X is a matrix that contains the results of the calculations that the function performs. Note that line 3 of the function creates the matrix X and specifies that it has as many rows as there are bins (or steps). X has 5 columns.

Line 4 defines a loop. The function must calculate conditional means etc for every one of the bins and there are "nstep" bins. Hence the commands within the loop will be executed "nstep" times. The range of commands in the loop is defined by the parentheses { }. On the first pass through the loop the value of the index i is 1. On the second pass, i becomes 2 and so on.

Inside the loop, the first command defines a bin that contains a subset of the data. The subsets are defined by the size variable. The first bin contains observation on houses whose size is between "left" and "left + width". On the first pass, "left" is equal to the argument "minsize" which might be 800 and since "width" is 200, the first bin would be `800 <= size < 1000`.

The next 5 statements define the values that are to be stored in the five columns of the matrix X (on the first pass, the first row of X will be completed since i = 1. Note that the fifth column stores the number of observations in the bin.

The statement "left = left + step" shifts the left boundary of the bin a distance of "step" square feet. If "step" is 50, the bins are shifted right 50 square feet every step. Note that the loop has "nstep" iterations so the bins move a total distance of nstep*step square feet or 40*50 = 2000 square feet. The left-most boundary starts at 800 square feet and at the end of the loop finishes at 2,800 square feet.

The loop is terminated by "}"

The final command of the function is "return x" which defines the output of the function – it is a matrix. The final "}" marks the end of the definition of the function.

How to Create, Save, Edit and Execute User-Defined Function in R

This section explains how to execute a user-defined function in R. It also explains how a function can be edited and saved in a text file.

User-defined functions can be stored as text files and pasted into R. In Windows, Notepad is ideal for creating, editing and saving text files. Follow these steps to use conmeanp in your R session.

Step 1: Open R and attach your data

Step 2: Set your working directory to your location of choice.

Step 3: Check your data are attached by typing (for example): head(data), where "data" is the name of your data frame .

Step 4: Copy the conmeanp function and paste it into R (then tap “enter”).

Step 5: Type ls() to list objects. You should find that conmeanp is listed.

Step 6: Type “conmeanp” to view the function itself.

Step 7: Execute conmeanp by typing a line such as:

```
X <- conmeanp(data,700,200,50,40)
```

Review the explanation of how this function works to be sure you understand the role of each of the arguments. Note that instead of “data” you must use the name of your own data frame.

Step 8: Type X to examine the matrix of data created by the function conmeanp.

When you quit R using q(), you will be asked if you would like to save the workspace, which you would normally do. All objects created in the current session will be saved in the workspace, including conmeanp.

The workspace does not save the individual line commands that you have entered but these can be saved in a history file. Type :

savehistory(“session1.Rhistory”)

to save all the session’s line commands. The first part of the file name is arbitrary, here it is chosen to be “session1”. The file is a text file (it can be opened by Notepad) that will be saved in the current working directory. This file can also be opened from within R (in the Console window, open the “File” drop-down menu.) After opening the history file you can scroll through the history of commands. You can also see all the commands in the console’s text editor window by entering:

history(Inf)

Note “Inf” is capitalised.

Creating Plots

The following 8 lines of code will produce a scatter plot and add three lines (the conditional mean, the conditional median and the major axis.

```
plot(size,price,pch=16,main="Chart 5")
subdat <- subset(mls,size <=2500)
lines(x[,1],x[,2],col="red",lwd=3)
lines(x[,1],x[,3],col=3,lwd=3)
lines(x[,1],x[,3],col=3,lwd=3)
b <- (var(price)/var(size))^.5
a <- mean(price) - b*mean(size)
abline(a,b,lwd=3,col=4)
```

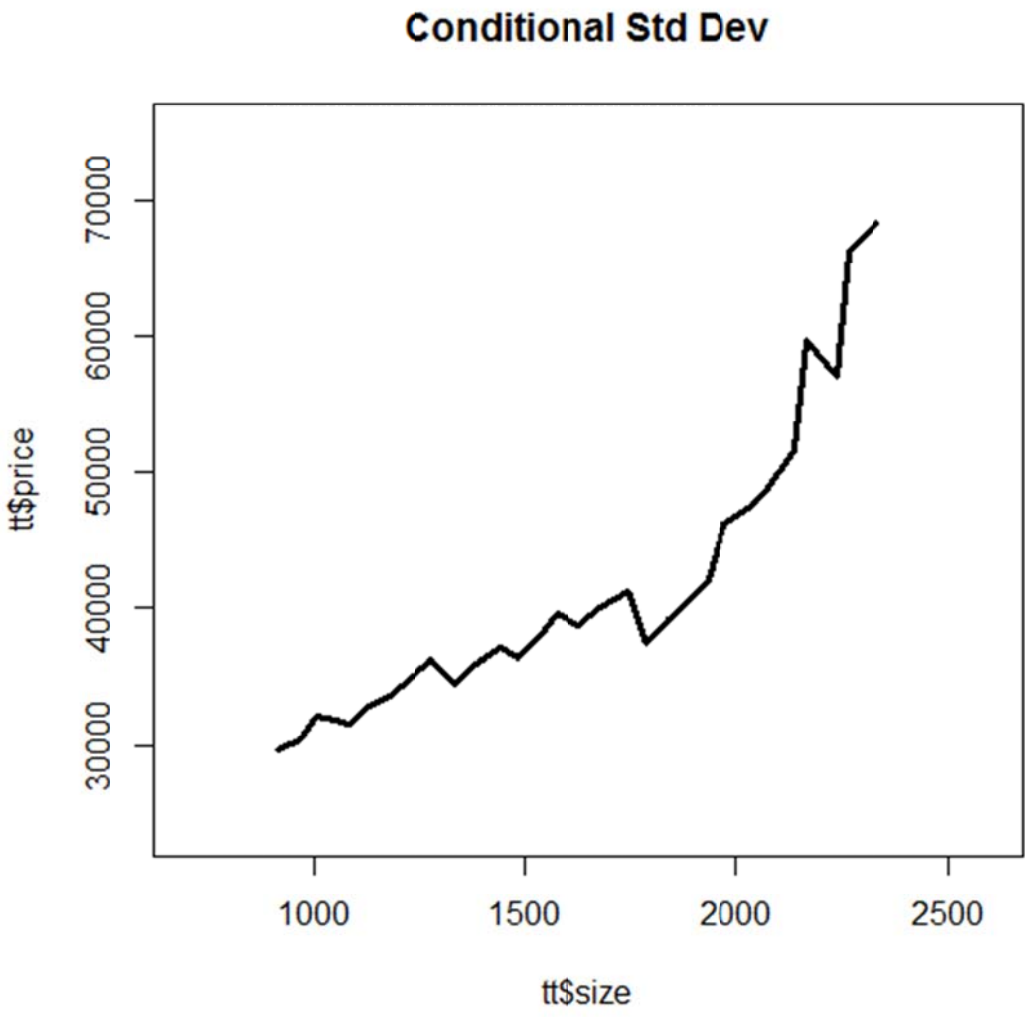
Having created the plot, save it in a form of your choice. To embed the chart in a Word document, you could use the png format. Make the plot window the active window. Then from the File drop-down menu save the file with the png file extension. The graphic file will be saved in the working directory. In Word "insert" the plot as an image.

The conditional standard deviation of price is unlikely to exceed $90,000 so when you plot the conditional standard deviation you will need to modify the Y-axis accordingly. Also, you will not want to scatter plot shown in this chart. The following code defines a subset of data. The scatter plot of price and size creates the plot frame but by using

type = "n"

the markers are suppressed (not shown.) When the conditional standard deviation is added to the plot, the result is a plot like the one that appears below the code:

```
tt <- subset(mls,price<75000 & size < 3000)
plot(tt$size,tt$price,main="Conditional Std Dev", type="n")
lines(x[,1],x[,4],lwd=3)
```


Conditional Std Dev
tt$price
30000
40000
50000
60000
70000
1000
1500
2000
2500
tt$size