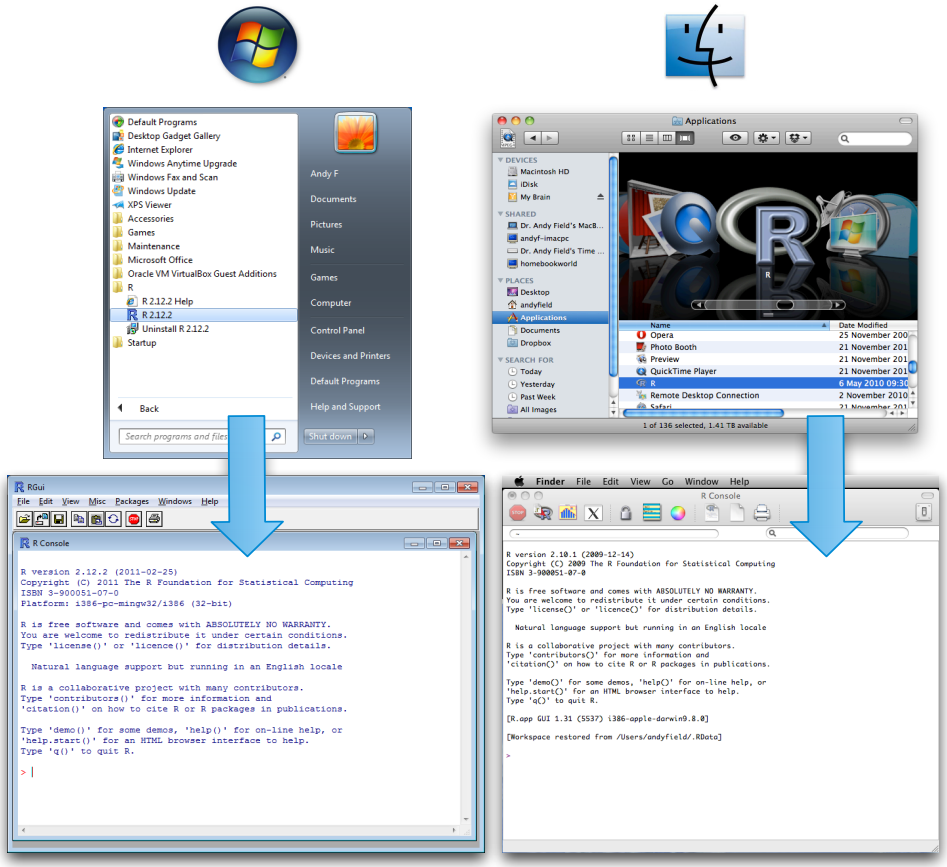
The **R** Environment

Prof. Donald Davendra

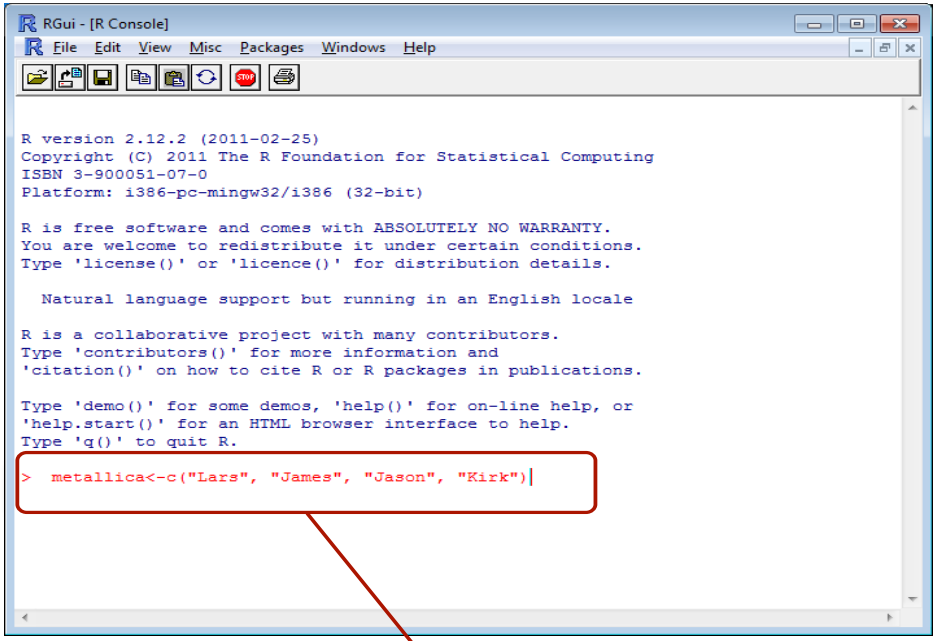
DISCOVERING STATISTICS USING R

Starting R



ANDY FIELD

The **R** Console

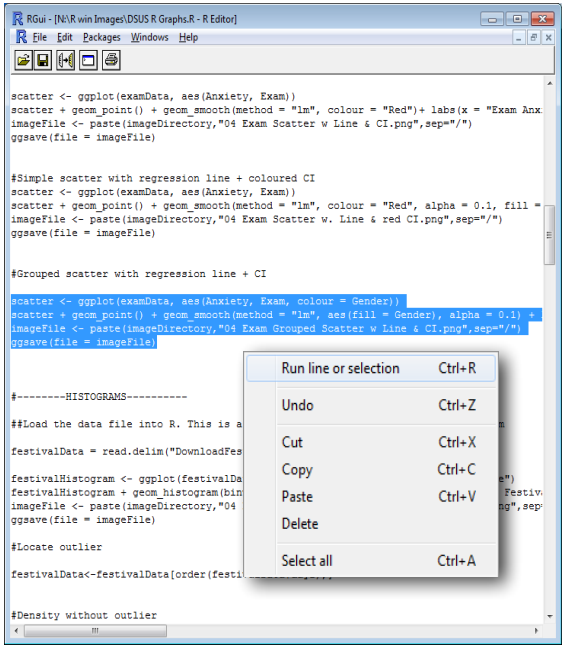


`metallica<-c("Lars", "James", "Jason", "Kirk")`

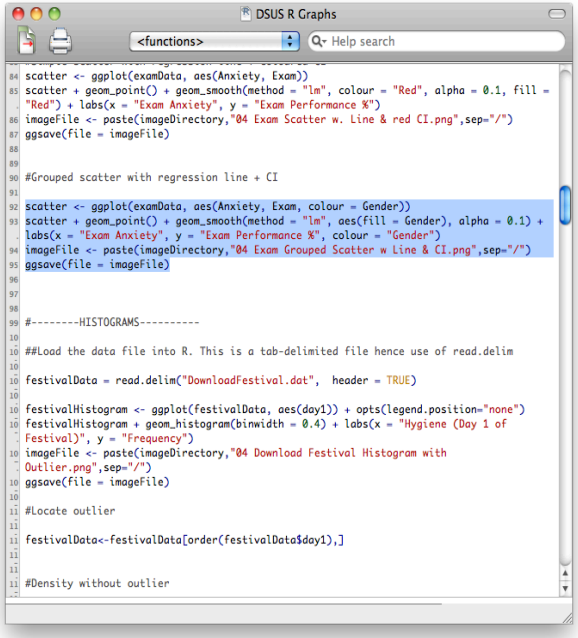
Object — Function

Using the command line in R

R Editor Window



Ctrl + R



⌘ + ↵

Executing commands from the R editor window

Set A Working Directory

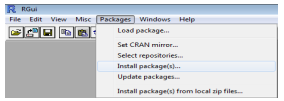
- Create a folder (in the usual way in Windows or MacOS) and place the data files you'll be using in that folder.
- When you start your session in **R** change the working directory to be the folder that you have just created.

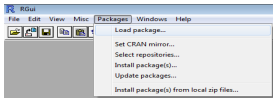
How to Set Your Working Directory

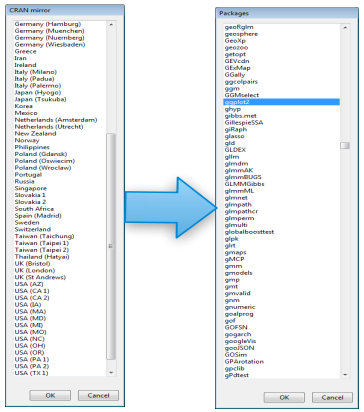
- Let's assume that you have a folder in 'My Documents' called 'Data' and within that you have created a folder called 'R Book Examples' in which you have placed some data files that you want to analyse.
- Set the working directory using *setwd()*:
  - setwd("C:/Documents/Data/R Book Examples")
- We can now access files in that folder directly. For example:
  - myData = read.delim("data.dat")

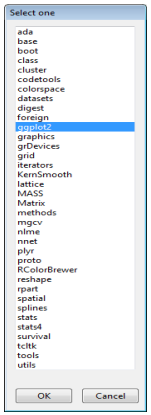
DISCOVERING STATISTICS USING R

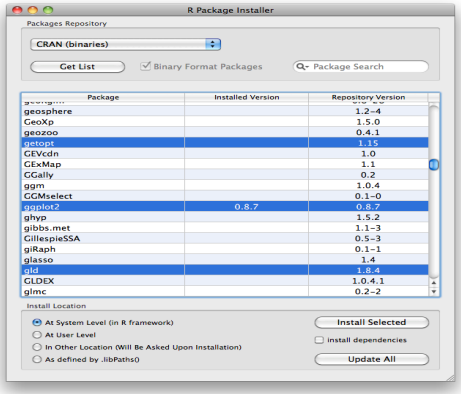
Installing Packages

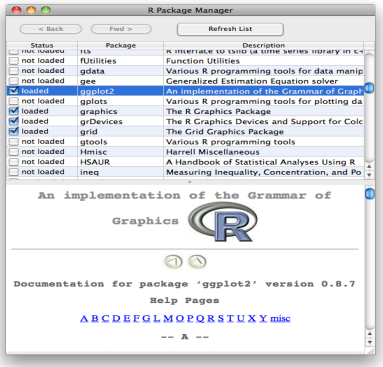












ANDY FIELD

Variable Types

- Numeric
  - Numbers (e.g. 7, 0, 120)
- String
  - Letters (e.g. 'Andy', 'Idiot')
- Coding variables/factors
  - Uses numbers to represent different groups of data (e.g. Gender)
- Date
  - Dates (e.g. 21-06-1973, 06-21-73, 21-Jun-1973)

Creating a String Variable

- We use the *c()* function and list all values in quotations so that **R** knows that it is string data.
- As such, we can create a variable called **name** as follows:
  - name<-c("Ben", "Martin", "Andy", "Paul", "Graham", "Carina", "Karina", "Doug", "Mark", "Zoe")

Creating a Date Variable

- Use the *as.Date()* function:

```
DoB<-as.Date(c("1977-07-03", "1969-05-24", "1973-06-21", "1970-07-16", "1949-10-10", "1983-11-05", "1987-10-08", "1989-09-16", "1973-05-20", "1984-11-12"))
```

- Each date has been entered as a text string (in quotations) in the appropriate format (yyyy-mm-dd).
- By enclosing these data in the *as.Date()* function, these strings are converted to date objects.

Creating a Coding Variable

- Imagine we had 5 students and 5 lecturers in a data sample and we wanted to create a coding variable called **job.**
- Enter the data:
  - job<-c(1,1,1,1,1,2,2,2,2,2)
- Or:
  - job<-c(rep(1, 5),rep(2, 5))
- Then convert **job** to a factor:
  - job<-factor(job, levels = c(1:2), labels = c("Lecturer", "Student"))

Creating a Numeric Variable

- Numeric variables are the easiest ones to create:
  - friends<-c(5,2,0,4,1,10,12,15,12, 17)
  - alcohol<-c(10,15,20,5,30,25,20,16,17,18)
  - income<-c(20000,40000,35000,22000,50000, 5000,100,3000,10000,10)
  - neurotic<-c(10,17,14,13,21,7,13,9,14,13)

Creating a Dataframe

- We can then bind the variables together into a dataframe:

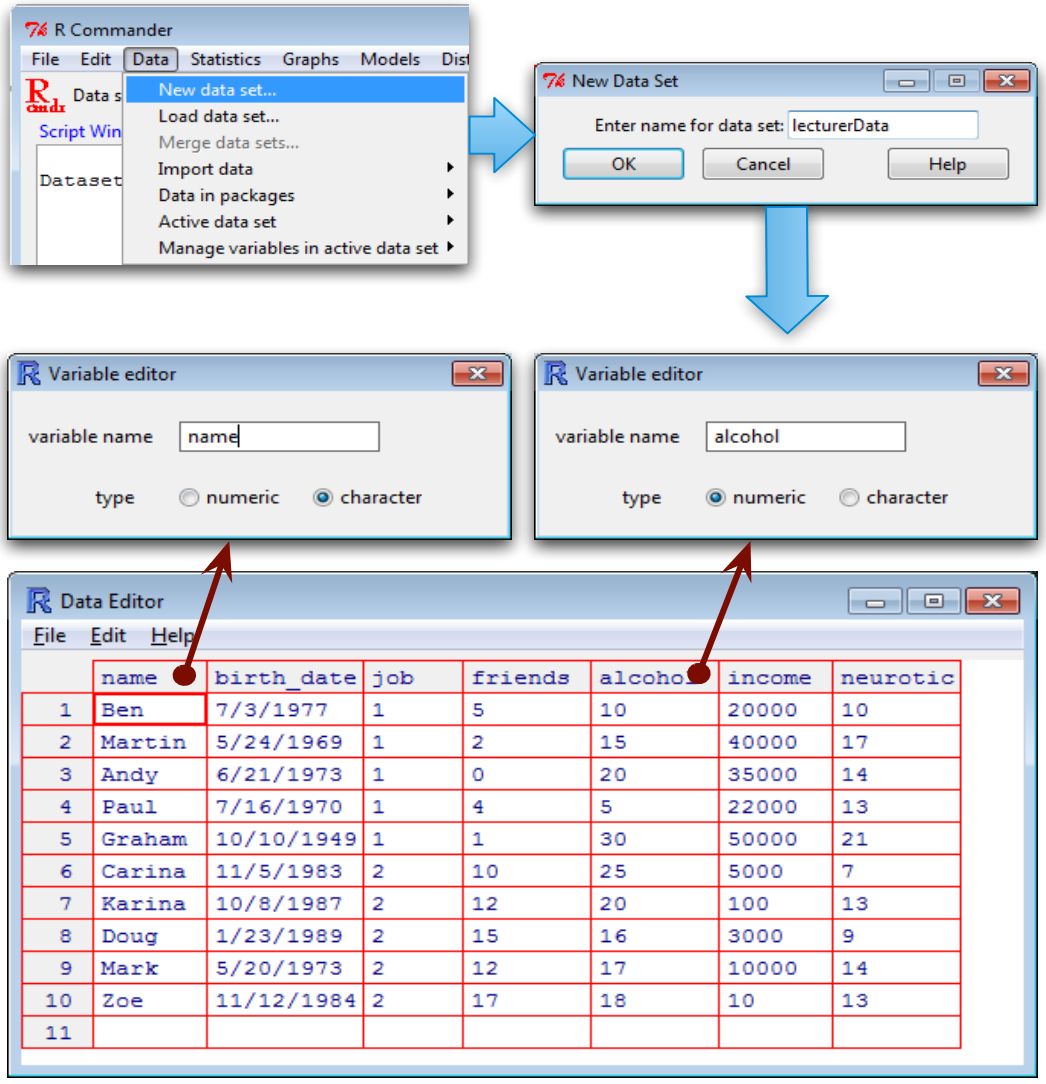
```
lecturerData<-data.frame(Name,  DoB, job, friends,
alcohol, income, neurotic)
```

lecturerData

	name	birth_date	job	friends	alcohol	income	neurotic
1	Ben	1977-07-03	Lecturer	5	10	20000	10
2	Martin	1969-05-24	Lecturer	2	15	40000	17
3	Andy	1973-06-21	Lecturer	0	20	35000	14
4	Paul	1970-07-16	Lecturer	4	5	22000	13
5	Graham	1949-10-10	Lecturer	1	30	50000	21
6	Carina	1983-11-05	Student	10	25	5000	7
7	Karina	1987-10-08	Student	12	20	100	13
8	Doug	1989-09-16	Student	15	16	3000	9
9	Mark	1973-05-20	Student	12	17	10000	14
10	Zoe	1984-11-12	Student	17	18	10	13

DISCOVERING STATISTICS USING R

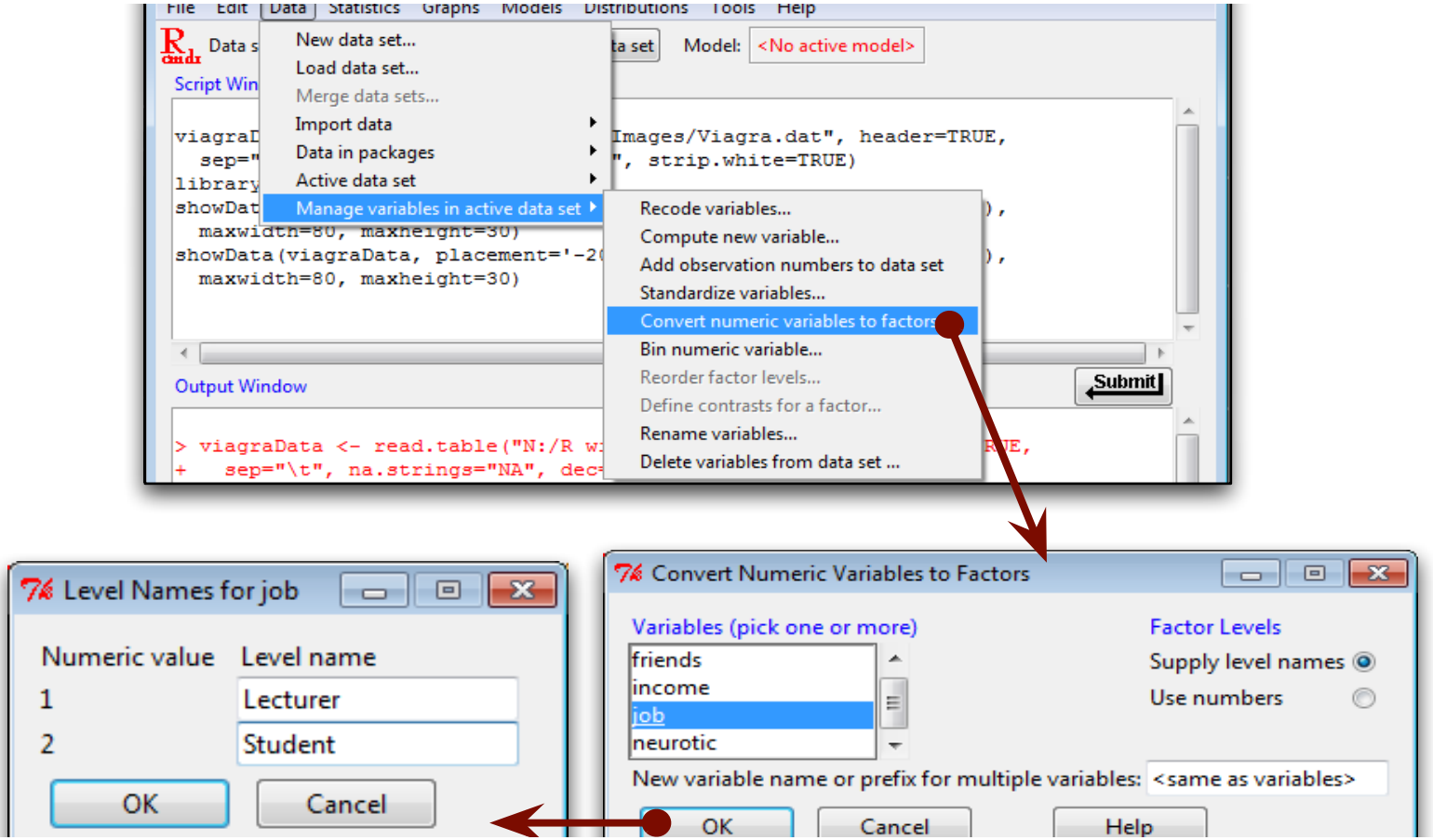
Entering data with R Commander



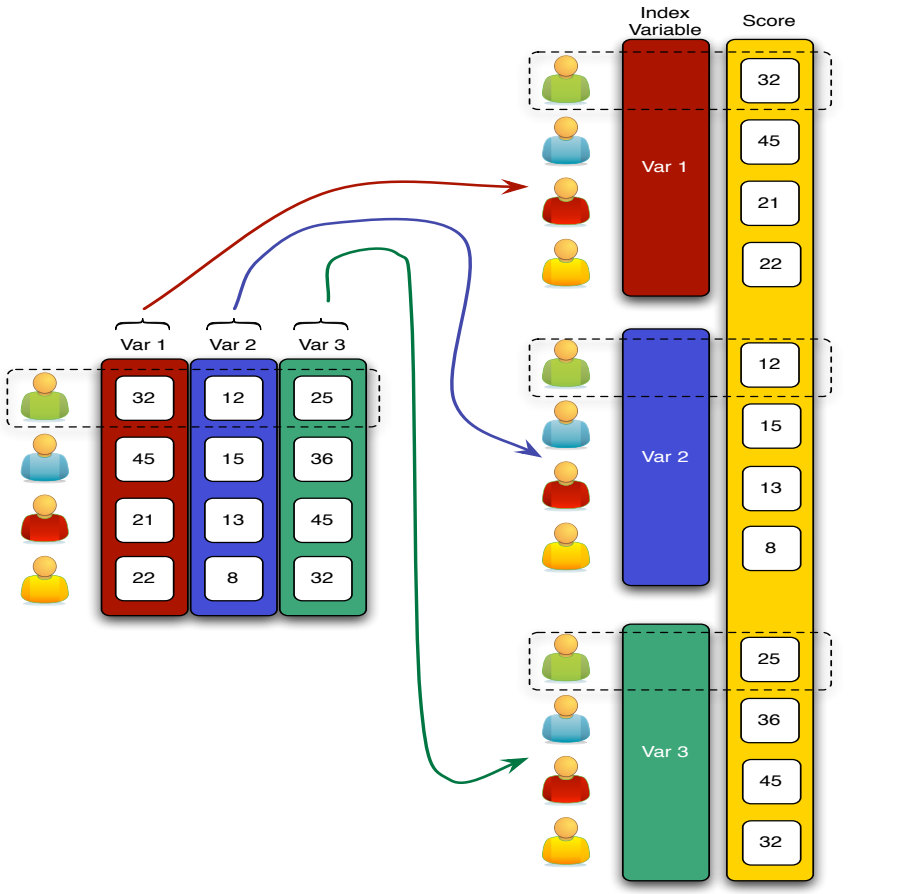
	name	birth_date	job	friends	alcohol	income	neurotic
1	Ben	7/3/1977	1	5	10	20000	10
2	Martin	5/24/1969	1	2	15	40000	17
3	Andy	6/21/1973	1	0	20	35000	14
4	Paul	7/16/1970	1	4	5	22000	13
5	Graham	10/10/1949	1	1	30	50000	21
6	Carina	11/5/1983	2	10	25	5000	7
7	Karina	10/8/1987	2	12	20	100	13
8	Doug	1/23/1989	2	15	16	3000	9
9	Mark	5/20/1973	2	12	17	10000	14
10	Zoe	11/12/1984	2	17	18	10	13
11							

DISCOVERING STATISTICS USING R

Creating coding variables with R Commander



Wide Format vs. Long Format



Reshaping Data

- To change between wide and long data formats we can use the *melt()* and *cast()* commands from the *reshape* package:
  - newDataFrame<-melt(oldDataFrame, id = c(constant variables), measured = c(variables that change across columns))
  - newData<-cast(moltenData, variables coded within a single column ~ variables coded many columns, value = "outcome variable")
- For simple data sets we can use *stack()* and *unstack()*:
  - newDataFrame<-stack(oldDataFrame, select = c(variable list))
  - newDataFrame<-unstack(oldDataFrame, scores ~ columns)