

Displays for Statistics 5401

Lecture 1

September 7, 2005

Christopher Bingham, Instructor

612-625-1024, kb@umn.edu  
372 Ford Hall

Class Web Page

<http://www.stat.umn.edu/~kb/classes/5401>

© 2005 by Christopher Bingham

1

*Multivariate* statistics emphasizes the **simultaneous** analysis of **more than one** response variables

$$X_1, X_2, \dots, X_p, \text{ with } p > 1$$

measured or observed on a *single experimental or observational "unit"* such as a person, a tree, a plot or a classroom.

Multivariate statistics often makes inference about a whole vector

$\theta = [\theta_1, \dots, \theta_k]'$  of parameters at once, that is, simultaneously. For example several methods result in inference about a mean vector

$$\mu = [\mu_1, \mu_2, \dots, \mu_p]'$$

Univariate statistics would make separate inferences about each  $\mu_j$ .

We generally use the notation

$p \equiv$ number of variables
$n$ or $N \equiv$ number of cases

3

**Multivariate vs Univariate Statistics**

Traditional statistics deals with only *one* variable such as **height, survival time** or **crop yield**, at a time.

Such an approach is *univariate*.

Exception? **Multiple regression analysis**

where you predict  $y$  on the basis of  $k$  variables  $x_1, \dots, x_k$  using a model like

$$y = \beta_0 + \beta_1 x_1 + \dots + \beta_k x_k + e$$

No! This is part of univariate statistics because there is only *one response* variable even though there are *many predictor* variables.

Stat 5421 includes some **multivariate categorical data analysis**. Before summarizing in a contingency table, data consist of "levels" of  $p$  different categories for each of many cases. This is truly multivariate. We won't explore multivariate categorical data at all.

2

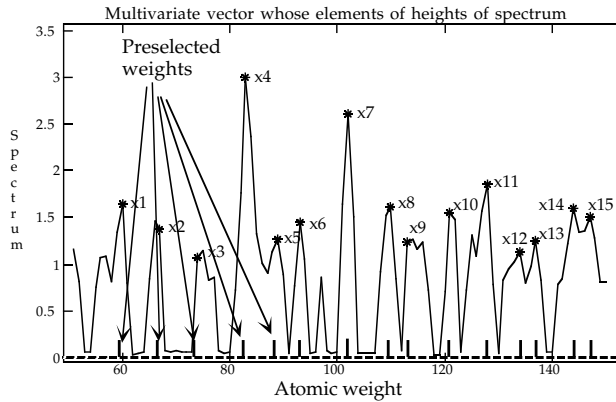
**Examples:**

- $p = 3$  measurements on a tree:
  - $X_1 =$  DBH = diameter at breast height
  - $X_2 =$  height of tree
  - $X_3 =$  age of tree
- $p = 5$  anthropometric measurements
  - $X_1 =$  body weight       $X_4 =$  femur length
  - $X_2 =$  body height       $X_5 =$  tibia length
  - $X_3 =$  skull height
- $p = 4$  scores on "battery" of tests taken by an individual
  - $X_1 =$  score on math aptitude test
  - $X_2 =$  score on abstract thinking test
  - $X_3 =$  score on verbal aptitude test
  - $X_4 =$  score on anxiety profile test
- $p = 75$  ratings on each of 75 items on a questionnaire.

4

- Heights of lines in a "spectrum":  $X_1, \dots, X_p$  = measurements of intensity (height) of spectrum at  $p$  specific frequencies or molecular weights of interest.

Example with  $p = 15$ . The location of the peaks was *chosen in advance*.



These are not real data.

Sometimes one or more *subsets* of variables *are* repeated measures, but the *whole set* of variables is not.

**Example** ( $p = 14$ ):

- $x_1 \dots x_6$  = systolic blood pressure every 4 hours (one subset of repeated measures variables)
- $x_7 \dots x_{12}$  = heart rate every 4 hours (second subset of repeated measures variables)
- $x_{13}$  = age, and  $x_{14}$  = weight (not repeated measures).

But

$x_1$  to  $x_{12}$  (or  $x_1$  to  $x_{14}$ ) are *not* repeated measures data because not all the values are comparable.

**Repeated measures**

$p = 6$  measurements of heart rate on the *same* individual every four hours for a day at 0400h, 0800h, 1200h, 1600h 2000h, 2400h.

All these examples represent multiple data items for the *same* individual or experimental/observational unit.

**Note** For measurements  $x_1, x_2, \dots, x_p$  to be considered **repeated measures**, all  $x_i$  must be *directly comparable*.

This means they are determinations of the *same* quantity at *different* times or under different conditions.

- A person's height and weight *doesn't* constitute repeated measures data.
- A person's height at ages 1, 2, 3, 5, 10 and 15 *would* be repeated measures data.
- A persons heart rate after jogging 1/4 mile, 1/2 mile, 1 mile, & 2 miles.

**An example with many variables which is not multivariate**

Suppose you have tree data like the following:

- DBH on one set of 100 trees
- height on another set of 100 trees
- age on a third set of 100 trees

These are three *univariate* data sets.

They do *not* make up a multivariate data set.

You have absolutely no information on possible relationships between variables.

Using multivariate methods on such data is like doing a paired t-test with independent samples.

**Remark:** Paired data is probably the simplest example of multivariate (bivariate) data. In fact, it's repeated measures data.

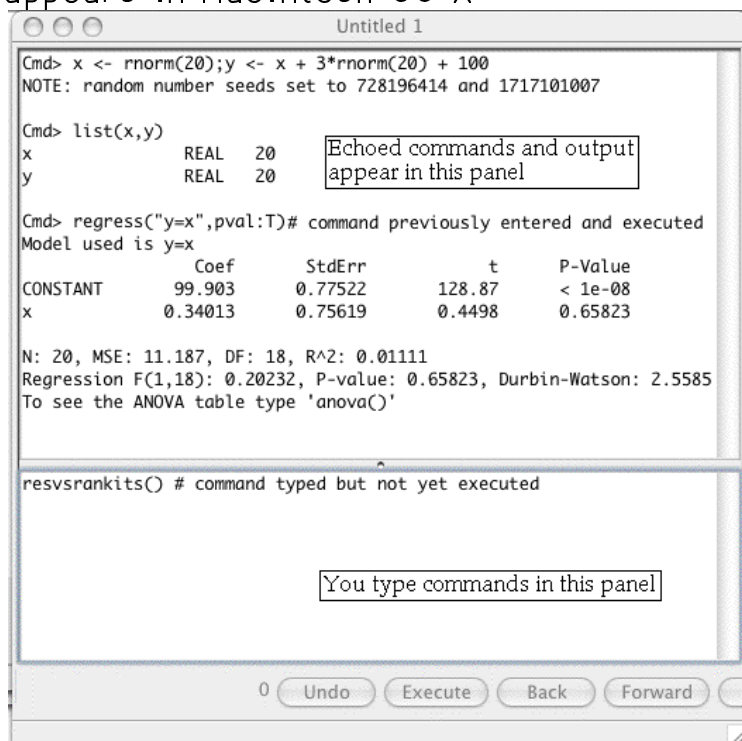
## A little bit about MacAnova

There is a completely new version, Carapace MacAnova.

In the older Classic MacAnova, you type commands in a window and output is printed in the same window after the command.

In Carapace MacAnova, you type commands in the **lower panel** of a window with two panels and a row of buttons. Then your command is echoed to the upper panel and is followed by output..

Here is the two panel window as it appears in Macintosh OS X



When you type **Return** or **Enter** after the command it is echoed above with output.

## MacAnova as *numerical* calculator

```
Cmd> 3/4 # Cmd> is the "prompt"
(1) 0.75 The answer is automatically printed

Cmd> sqrt(17) + log10(20) # you can use named functions
(1) 5.4241 4.1231 + 1.301 sqrt(17) + log10(20)
```

## MacAnova as *symbolic* calculator

```
Cmd> pi <- 3*log(640320)/sqrt(163)#natural log

Cmd> # previous lines assigns value to variable pi using

Cmd> # assignment operator <-

Cmd> pi # Comment starting with "#"
(1) 3.1416

Cmd> sqrt(2*pi) # square root of 2 times pi
(1) 2.5066
```

Anything after # is ignored so that you can add comments to any line.

"<-" is the **assignment operator**.

The value of the *right* side is assigned to the variable named on the *left* side.

```
Cmd> PI # predefined variable with value pi
(1) 3.1416

Cmd> E # predefined variable with value e
(1) 2.7183
```

Although they have the same *value*, `PI` is a different *variable* from `pi` since upper and lower case matter in names.

## Variable names

- Start with letter (a-z, A-Z)
- Continue with letter, number or `_`
- Length  $\leq$  12 characters
- `x`, residuals, Height, `y1`, `no_treatment`
- Upper and lower case matters: Height is different from height.

## No dots in names.

- `pi.hat` is illegal but `pi_hat` is OK.

```
Cmd> pi.hat <- 5/7 # illegal variable name
ERROR: do not use . in variable names near pi.

Cmd> pi_hat <- 3/7 # legal variable name
```

Names can also start with `_` (underscore) but you should avoid such names since they have a special meaning: A variable whose name begins with `"_"` is "invisible" and you may not see its value when you expect to.

Names can also start with @ followed by a letter (a-z, A-Z).

A variable with such a name is *temporary*; it will be deleted before the next command is executed.

This can be useful, like a scratch pad; you save an intermediate result in a temporary variable, keeping only the final value.

```
Cmd> @tmp <- 3*log(640432); pi <- @tmp/sqrt(163)
```

```
Cmd> @tmp
UNDEFINED
```

13

A variable can contain several values:

- A **vector** has 1 dimension
- A **matrix** is a 2 dimensional table
- An **array** has more than 2 dimensions.

```
Cmd> y <- vector(42,52,48,58, 4,5,4,3)
```

```
Cmd> # y is a vector made up of all the arguments of vector()
```

```
Cmd> y # typing y prints it
(1) 42 52 48 58 4
(6) 5 4 3
```

The numbers, 1 and 6, in ( ) identify the first numbers in the rows as being the first and sixth elements in *y*.

```
Cmd> x <- matrix(y,4) # make a matrix x with 4 rows and 2 cols
```

```
Cmd> # matrix(vec, n) makes a matrix with n rows from vec
```

```
Cmd> x # or print(x); print the value
```

```
(1,1) 42 4
(2,1) 52 5
(3,1) 48 4
(4,1) 58 3
```

The pairs of numbers in ( ) identify the first numbers in each row as being elements in rows 1 through 4 and in column 1 of *x*. For example 48 is in row 3 and column 1.

```
Cmd> x + 5 # You can do arithmetic directly with vector,matrix
```

```
(1,1) 47 9
(2,1) 57 10
(3,1) 53 9
(4,1) 63 8
```

15

## Assigning a value to a variable

```
Cmd> x <- 3.24
```

"assigns" the value 3.24 to variable *x*.

- If *x* already exists, its old value is lost
- If *x* does not previously exist, it will exist after the assignment.

## Seeing the value of a variable

Just typing a variable's name prints its value

```
Cmd> x
(1) 3.24
```

Ignore the number in ( ) for the moment.

You can also use `print()`:

```
Cmd> print(x)
x:
(1) 3.24
```

`print(x,y)` would print both *x* and *y*.

`print(x,nsig:12)` prints *x* to 12 significant digits.

See *Introduction to MacAnova* for more examples.

14

You extract information from a vector, matrix or array using **subscripts**.

```
Cmd> y[3] # single number extracts y_3 = element 3 of y
(1) 48
```

```
Cmd> y[vector(1,3,5)] # y_1, y_3 and y_5
(1) 42 48 4
```

```
Cmd> y[-3] # everything but y_3
(1) 42 52 58 4
(6) 4 3
```

```
Cmd> run(4) # numbers 1, 2, 3, 4
(1) 1 2 3 4
```

```
Cmd> y[-run(4)] # everything but y_1, y_2, y_3, y_4
(1) 4 5 4 3
```

With a *matrix* you need 2 subscripts (row, column)

```
Cmd> x[3,2] # element in row 3 of column 2 of x
```

```
Cmd> x[,2] # all of column 2
```

```
(1,1) 4
(2,1) 5
(3,1) 4
(4,1) 3
(1,1) 4
```

```
Cmd> x[-1, ] # all rows except row 1
```

```
(1,1) 52 5
(2,1) 48 4
(3,1) 58 3
```

```
Cmd> x[-run(2),2] # column 2 omitting rows 1 and 2
```

```
(1,1) 4
(2,1) 3
```

```
Cmd> x[x[,1] >= 50, ] # rows of x with column 1 >= 50
```

```
(1,1) 52 5
(2,1) 58 3
```

16