Working with Stata

The Basics

April 10, 2017

Nicola Orsini

Biostatistics Team

Department of Public Health Sciences

Karolinska Institutet

Goal

This material

- helps to get familiar with Stata language
- introduces basic commands for describing, summarizing, and graphing the data
- shows how to generate and recode variables

Stata's distinctive features

- Simple language (syntax)
- Dialogs to issue commands
- Do-file editor to save time in developing analysis
- Dataset loaded in the current memory
- Easy to repeat steps (looping)

- Periodically updates of executable and commands
- Easy to create new commands
- Stata's user community provides useful additions (Statistical Software Components) and support (Mailing list)
- Stata is cross-platform compatible (Windows, Mac, Linux, and Unix)

Stata windows

Command line - to enter the commands

Results - to see the output of the commands

Variables - to see the variables in memory

Review - to see previously entered commands

Type of files and extensions

.dta Dataset

.do Do-file

.hlp Help file

.ado Stata command

.gph Graph

How to get help

- On-line

If you know the name of the command
help ***cmdname***

If you don't know the name of the command
findit ***keywords***

Resources to help you learn and use Stata
http://www.stata.com/links/resources1.html

Motivating example

Dataset

hyponatremia.dta

Reference

"Hyponatremia among Runners in the Boston Marathon",
New England Journal of Medicine, 2005, Volume 352:1550-1556.

Descriptive abstract

Hyponatremia has emerged as an important cause of race-related death and life-threatening illness among marathon runners. We studied a cohort of marathon runners to estimate the incidence of hyponatremia and to identify the principal risk factors.

Basic language syntax

The basic Stata language syntax is

command [*varlist*][**if** *exp*][**in** *range*] [,***options***]

where square brackets denote optional qualifiers.
if *exp* restricts the scope of a command to those observations for which the value of the *exp* is true

in *range* of a command to those observations for a specific observation range

options denotes special things to do (modify the default)

Load a dataset

To load a Stata dataset (extension **.dta**) into the memory

use *filename* [, **clear**]

clear option erases all data currently in memory and proceeds with loading the new data from the disk or from a web server.

// Examples

use c:\hyponatremia, clear

use http://stats4life.se/data/hyponatremia, clear

Looking at the data

list creates a list of values of specified variables and observations (as alternative to **browse**)

describe provides information on the size of the dataset and the names, labels and types of variables.

codebook summarizes a variable in a format designed for printing a codebook (missing value; unique values; descriptive statistics).

label list list names and contents of value labels (explanations attached, if any, to categorical variables).

// Examples

* basic information of the dataset

```
describe
```

* basic information about the variables

```
codebook
```

* list all variables

```
list
```

* list all variables as spreadsheet (like Excel)

```
browse
```

Data types

There is a range of storage types (help datatypes) for numerical variables

Storage type	Minimum	Maximum	bytes
Integers			
byte	-127	100	1
int	-32,767	32,740	2
long	-2,147,483,647	2,147,483,620	4
Reals			
float	-1.70141173319*10^38	1.70141173319*10^38	4
double	-8.9884656743*10^307	8.9884656743*10^307	8

Specifying the storage type is optional.

String variables are stored as

str# **where # range 1 to 244**

The length of the string variable is automatically decided by Stata, and it is related to the longest string of characters in that variable.

To refer to the content of string variables always remember to use double quotes.

Run a command in a subset of the data

List all the variables for the first 10 observations of the data

```
list in 1/10
```

List all the variables and observations if bmi is greater than or equal to 30

```
list if bmi >= 30
```

List the variables id and na for critical hyponatremia (<=120)

```
list id na if na <= 120
```

Rules for abbreviations

Stata allows abbreviations.

As a general rule, command, option, and variable names may be abbreviated to the **shortest string** of characters that **uniquely** identifies them.

For each command the shortest abbreviation can be determined by looking at the help file.

// Examples

d // is equivalent to describe

list id bmi female // is equivalent to

l i b f

list na nas135 // is equivalent to

l na*

list id na nas135 female urinat3p // is equivalent
l id-urinat3p

Save a dataset

To **save** a dataset type

save *filename* [, **replace**]

The **replace** option indicates that if *filename* already exists, Stata will overwrite it.

// Example

```
save my_hyponatremia.dta, replace
```

STATISTICAL ANALYSIS

Descriptive statistics were used to estimate the incidence of hyponatremia and to characterize the demographic information supplied by the runners.

Summary statistics

summarize calculates and displays a variety of univariate summary statistics. If no *varlist* is specified, then summary statistics are calculated for all the variables in the dataset.

summarize [*varlist*] [*if*] [*in*] [, **detail**]

where

detail produces additional statistics, including skewness, kurtosis, various percentiles and the four smallest and largest values.

```
. summarize na, detail

Serum sodium concentration
-------------------------------------------------------------
      Percentiles      Smallest
 1%           128            114
 5%           133            118
10%           135            119       Obs                 488
25%           138            125       Sum of Wgt.         488

50%           141                      Mean           140.4057
                     Largest           Std. Dev.      4.752102
75%           143            153
90%           145            155       Variance       22.58247
95%           147            156       Skewness       -.654781
99%           152            158       Kurtosis        6.78414
```

At the finish line, the runners had a mean serum sodium concentration of 140±5 mmol per liter (range, 114 to 158).

The middle 50% of the marathon runners had sodium concentration between 138 and 143 nmmo/liter.

Quantiles and percentiles are synonymous – 0.99 quantile is the 99th percentile.

The median, defined as the middle value of a set of ranked data, is the best-known percentile.

The quantile q is defined as that value of y that splits the data into the proportions q below and 1-q above.

$$F(y_q) = q \qquad\qquad y_q = F^{-1}(q)$$

where F is the cumulative distribution function.

Histogram

A histogram is a graphical method for displaying the shape of a distribution.

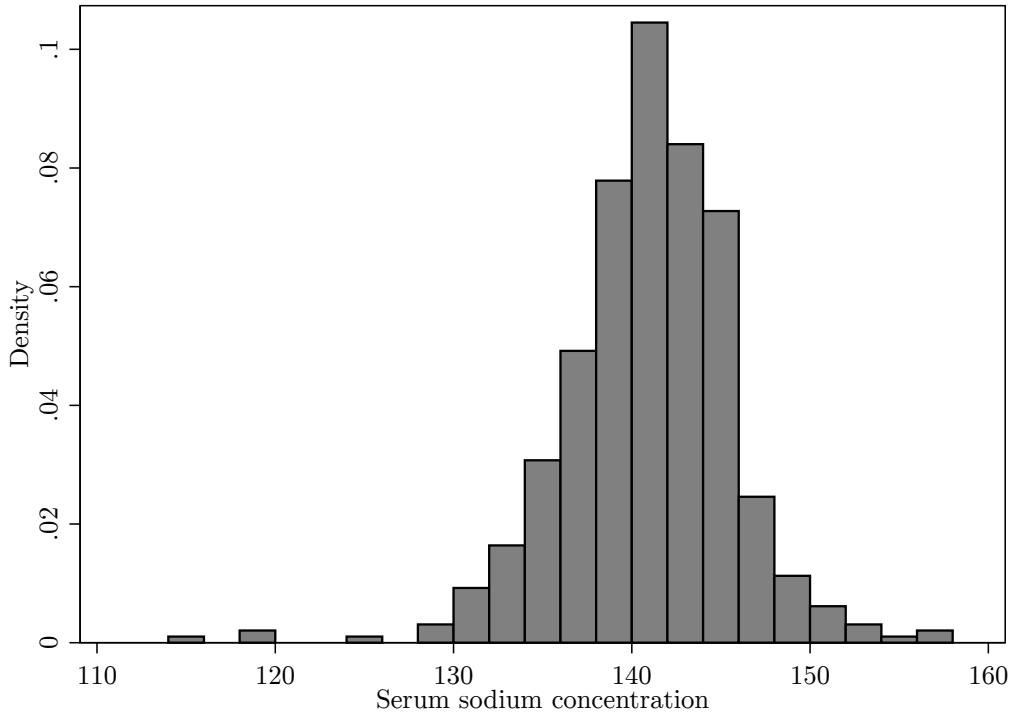
It is particularly useful when there are a large number of observations.

The height of each bar corresponds to its class frequency.

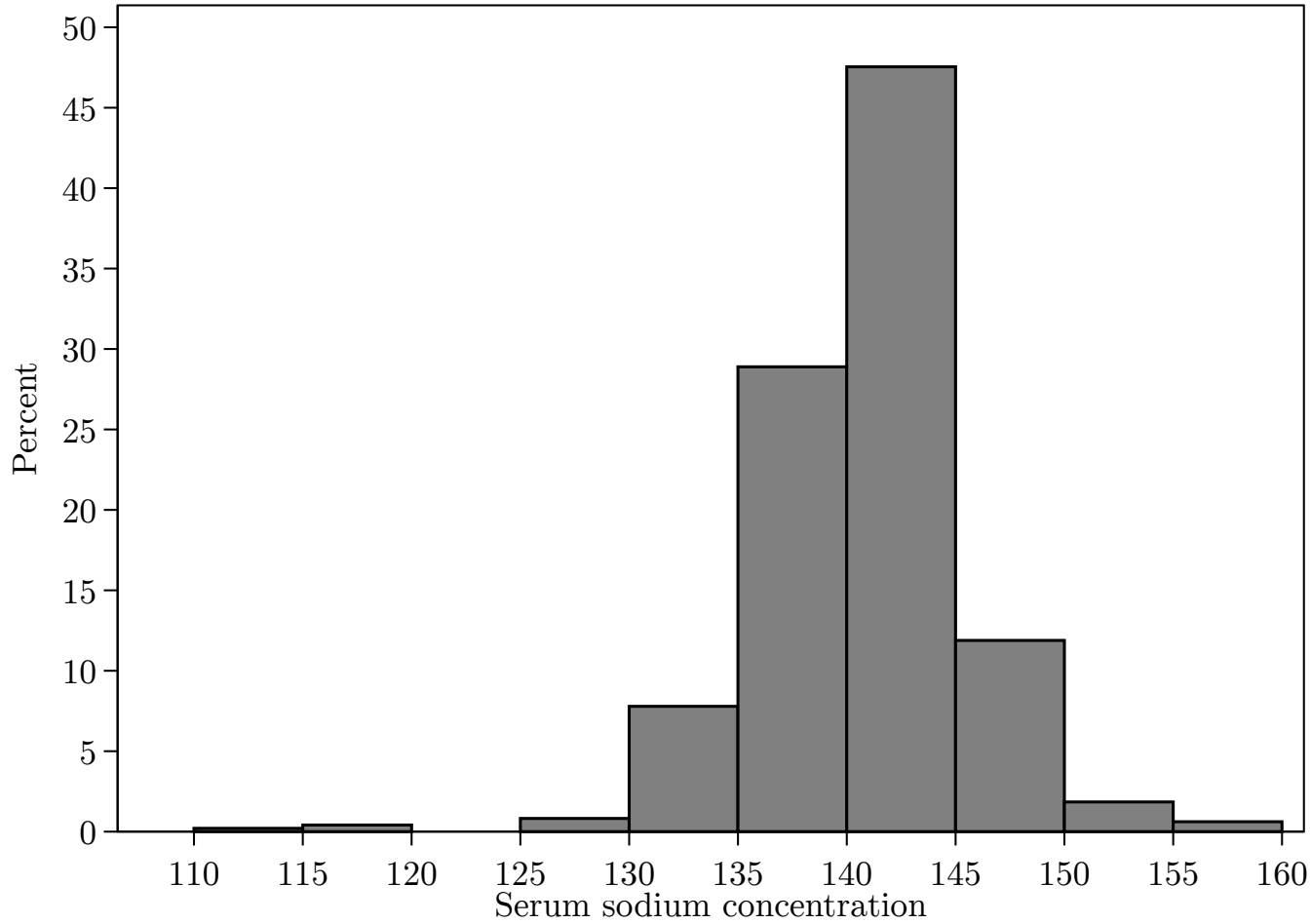
The class frequencies are represented by bars.

The command **histogram** allows you to control any aspect of the graph (y-axis and x-axis).

```
. histogram  na, frequency
```



. histogram na, percent start(110) width(5) ///
ylabel(0(5)50, angle(horiz)) xlabel(110(5)160)



Box-plot

A box plot provides an excellent visual summary of many important aspects of a distribution.

The box stretches from the lower hinge (defined as the 25th percentile) to the upper hinge (the 75th percentile) and therefore contains the middle half of the scores in the distribution.

The median is shown as a line across the box.

Box plots are useful for identifying outliers and for comparing distributions.

. graph box na

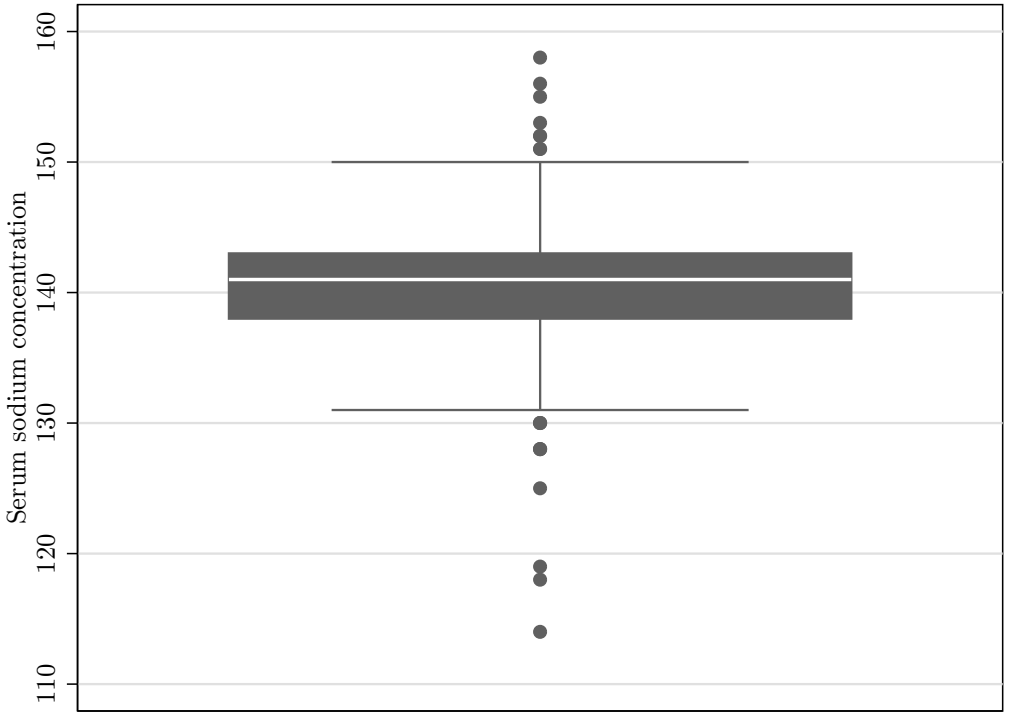


Table 1. Baseline Characteristics of the 2002 Boston Marathon Study Population.*

Characteristic	Male Runners (N=473)		Female Runners (N=293)	
	Reporting at Finish Line (N=336)	Not Reporting at Finish Line (N=137)	Reporting at Finish Line (N=175)	Not Reporting at Finish Line (N=118)
Age — yr	40.4±9.6	40.4±10.0	36.3±8.8	35.7±8.8
Nonwhite race — %	9	10	6	6
Prerace weight — kg	74.6±9.5	76.6±10.7	58.9±6.7	58.7±7.1
Body-mass index†	23.7±2.6	24.5±2.7	21.4±2.0	21.4±2.1

In the next few slides we will see how to control the presentation of descriptive statistics using the commands summarize, tabstat, and table.

Question: What is the mean and standard deviation of BMI among males and females?

```
. su bmi if female == 0

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         bmi |       303     23.7401    2.666677      16.77      32.21


. su bmi if female == 1

    Variable |       Obs        Mean    Std. Dev.       Min        Max
-------------+--------------------------------------------------------
         bmi |       162     21.4313    2.044459      17.06       31.7
```

It follows a couple of alternatives to the use of –if- qualifier.

Table of summary statistics (1)

table calculate and displays tables of statistics (**help table**).

table *rowvar* [*colvar* [*supercolvar*]] [*if*] [*in*] **[, contents(*clist*)]**

where ***clist*** can be for example:

freq (for frequency default)
mean ***varname***
sd ***varname***
sum ***varname***

Up to five statistics may be specified.

```
. table female , contents(mean bmi sd bmi) format(%2.1f)

-----------------------------------
   Female |  mean(bmi)      sd(bmi)
----------+------------------------
       No |        23.7          2.7
      Yes |        21.4          2.0
-----------------------------------
```

One of the most useful option of table is format() where you can control how many digits you want to display to the right of the decimal point (help format).

We next present mean BMI by gender and hyponatremia status.

```
. table female nas135, contents(mean bmi) format(%2.1f) row col

-----------------------------
           |    Serum sodium
           |concentration <= 135
           |     mmol/liter
   Female  |    No     Yes   Total
-----------+-----------------------
       No  |  23.6    25.2    23.7
      Yes  |  21.5    21.1    21.4
           |
     Total |  23.0    22.8    22.9
-----------------------------
```

The options row and col add the requested statistics based on the total across the rows and columns.

Tables of summary statistics (2)

tabstat displays summary statistics for a series of numeric variables in a single tables (**help tabstat**).

tabstat *varlist* [, statistics(*statname*) by(*varname*)]

where ***statname*** are descriptive statistics (mean default, min, max, sd, var...).

tabstat is a useful alternative to **summarize** because it allows you to specify the list of statistics to be displayed. Several options are available.

```
. tabstat bmi, by(female) stat(mean sd) format(%2.1f)

Summary for variables: bmi
     by categories of: female (Female)

female |      mean         sd
-------+--------------------
    No |      23.7        2.7
   Yes |      21.4        2.0
-------+--------------------
 Total |      22.9        2.7
----------------------------
```

The command **tabstat** is designed to present summary statistics for a series of numeric variables in one table.

```
. tabstat bmi wtdiff runtime  , by(female) f(%4.1f)

Summary statistics: mean
  by categories of: female (Female)

female |       bmi    wtdiff   runtime
-------+------------------------------
    No |      23.7      -1.0     217.7
   Yes |      21.4      -0.1     241.2
-------+------------------------------
 Total |      22.9      -0.7     225.5
----------------------------------------
```

ABSTRACT

RESULTS

Of 766 runners enrolled, 488 runners (64 percent) provided a usable blood sample at the finish line. Thirteen percent had hyponatremia (a serum sodium concentration of 135 mmol per liter or less); 0.6 percent had critical hyponatremia (120 mmol per liter or less). On univariate analyses, hyponatremia was associated with substantial weight

Count observations

count [*if*] [*in*]

counts the number of observations that satisfy the specified condition.

```
* runners with hyponatremia (<=135 mmol/l)
. count if na <= 135
   62

* runners with severe hyponatremia (<=120 mmol/l)
. count if na <= 120
   3
```

Table of counts

```
. tabulate nas135

      Serum |
     sodium |
concentrati |
  on <= 135 |
 mmol/liter |      Freq.     Percent        Cum.
------------+-----------------------------------
         No |        426       87.30       87.30
        Yes |         62       12.70      100.00
------------+-----------------------------------
      Total |        488      100.00
```

(range, 114 to 158). Thirteen percent (62 of 488) had hyponatremia, including 22 percent of women (37 of 166) and 8 percent of men (25 of 322). Three runners (0.6 percent) had critical hyponatremia (serum sodium concentrations, 119, 118, and 114 mmol per liter).

```
. tabulate nas135 female, col
```

Serum sodium concentration <= 135 mmol/liter	Female		Total
	No	Yes	
No	297	129	426
	92.24	77.71	87.30
Yes	25	37	62
	7.76	22.29	12.70
Total	322	166	488
	100.00	100.00	100.00

Table 2. Univariate and Multivariate Predictors of Hyponatremia.*

Variable	Univariate Predictors		
	Hyponatremia (N=62)	No Hyponatremia (N=426)	P Value†
Female sex (%)	60	30	<0.001

```
. tabulate nas135 female, row
     Serum |
    sodium |
concentrat |
ion <= 135 |          Female
mmol/liter |         No        Yes |     Total
-----------+----------------------+----------
        No |        297        129 |       426
           |      69.72      30.28 |    100.00
-----------+----------------------+----------
       Yes |         25         37 |        62
           |      40.32      59.68 |    100.00
-----------+----------------------+----------
     Total |        322        166 |       488
           |      65.98      34.02 |    100.00
```

```
. tabstat bmi, by(nas135) format(%2.1f) stat(mean sd)

Summary for variables: bmi
     by categories of: nas135 (Serum sodium
concentration <= 135 mmol/liter)

nas135 |      mean          sd
-------+--------------------
    No |      23.0         2.5
   Yes |      22.8         3.7
-------+--------------------
 Total |      22.9         2.7
-----------------------------
```

Table 2. Univariate and Multivariate Predictors of Hyponatremia.*

Variable	Univariate Predictors		
	Hyponatremia (N=62)	No Hyponatremia (N=426)	P Value†
Body-mass index	22.8±3.7	23.0±2.5	0.68
Category of body-mass index			0.01
<20 (%)	25	8	—
20–25 (%)	54	73	—
>25 (%)	21	19	—

Question: Can you guess why the two *P*-values are different?

Create a new variable

The command **generate** creates a new variable. The syntax is:

generate *newvar* = ***exp*** [*if*] [*in*]

where ***exp*** could be **any reasonable combination** of

variables, numbers, **operators** (help operators) and

functions (help functions).

Operators

1. **Arithmetic:** **+** (addition); **-** (subtraction); **/** (division); ***** (multiplication); **^** (raise to a power) and prefix **!** (negation). Any arithmetic operation on missing values or an impossible arithmetic operation yields a missing value.

2. **String:** **+** (concatenation of two strings), i.e. “Name” **+** “Surname”.

3. **Relational:** **>** (greater than); **<** (less than); **>=** (greater than or equal); **<=** (less than or equal); **==** (equal); **!=** (not equal)

4. **Logical:** **&** (and); **|** (or); **!** (not)

Overview of functions

Quick references are available for the following types of functions:

Type of function	See help
Mathematical functions	math functions
Probability distributions and density functions	density functions
Random-number functions	random-number functions
String functions	string functions
Programming functions	programming functions
Date and time functions	dates and times
Selecting time spans	time-series functions
Matrix functions	matrix functions

Example: Create a new variable that expresses running time in hours rather than in minutes.

gen runtimeh = runtime/60

The new variable created in the dataset is named runtimeh and it is the result of the expression runtime/60.

Each variable name in the dataset is unique.

The command generate cannot overwrite an existing variable with the same name; you need first to drop (help drop) the old version and then generate the new version.

Check the new variable

Once you create a new variable in the dataset it is always good to check that it contains what you would expect.

You can check using the list or browse commands.

You can check using the command tabulate and count for categorical variables.

You can check using the command summary for continuous variables.

Missing values

Missing values are stored as the largest positive values.

All numbers < .

You can think of a missing value as any other number when you specify an expression. For instance:

List the variables id and na among those observations with a missing value for bmi.

```
list id na if bmi == .
```

Replacing values

The command **replace** changes values of a variable that already exists.

replace *varname* = *exp* [*if*] [*in*]

For instance, replace to missing values the variable wtdiff for those runners who lost more than 5.4 kg

```
replace wtdiff = . if wtdiff < -5.4
```

How to categorize a variable

A general and valid approach is to create a new variable and run a number of replacements based on the original variable and according to the number of categories.

```
gen bmic = 1 if bmi < 20

replace bmic = 2 if bmi >= 20 & bmi < 25
replace bmic = 3 if bmi >= 25 & bmi <.

label define bmic 1 "<20" 2 "20-25" 3 ">25"

label values bmic bmic
```

```
. tab bmic nas135 , col nofreq  chi2

           |     Serum sodium
           | concentration <= 135
           |      mmol/liter
      bmic |        No         Yes |     Total
-----------+----------------------+----------
       <20 |      8.40       25.00 |     10.54
     20-25 |     72.59       53.33 |     70.11
       >25 |     19.01       21.67 |     19.35
-----------+----------------------+----------
     Total |    100.00      100.00 |    100.00

          Pearson chi2(2) =  16.6287   Pr = 0.000
```

Dichotomize a continuous variable

Create an indicator variable to identify 1 = gain weight and 0 = no change or lose weight.

```
gen     gainweight = 1 if wtdiff  > 0 & wtdiff < .
replace gainweight = 0 if wtdiff <= 0

. tab  gainweight, missing
 gainweight |      Freq.     Percent        Cum.
------------+-----------------------------------
          0 |        320       65.57       65.57
          1 |        135       27.66       93.24
          . |         33        6.76      100.00
------------+-----------------------------------
      Total |        488      100.00
```

Labels

Stata makes it easy to provide labels for the dataset, for each variable, and for each value of a categorical variable, which will help readers to better understand the data.

To label the dataset

label data "Hyponatremia among runners"

To label the variable gainweight

label variable gainweight "Post>Pre-race weight"

Two steps to set value labels associated with numbers:

1. create labels (**label define**)

label define gw 1 "Post>Pre" 0 "Post<=Pre"

2. attach the labels to the variable (**label values**)

label values gainweight gw

```
. tab gainweight

 gainweight |      Freq.     Percent        Cum.
------------+-----------------------------------
  Post<=Pre |        320       70.33       70.33
   Post>Pre |        135       29.67      100.00
------------+-----------------------------------
      Total |        455      100.00
```

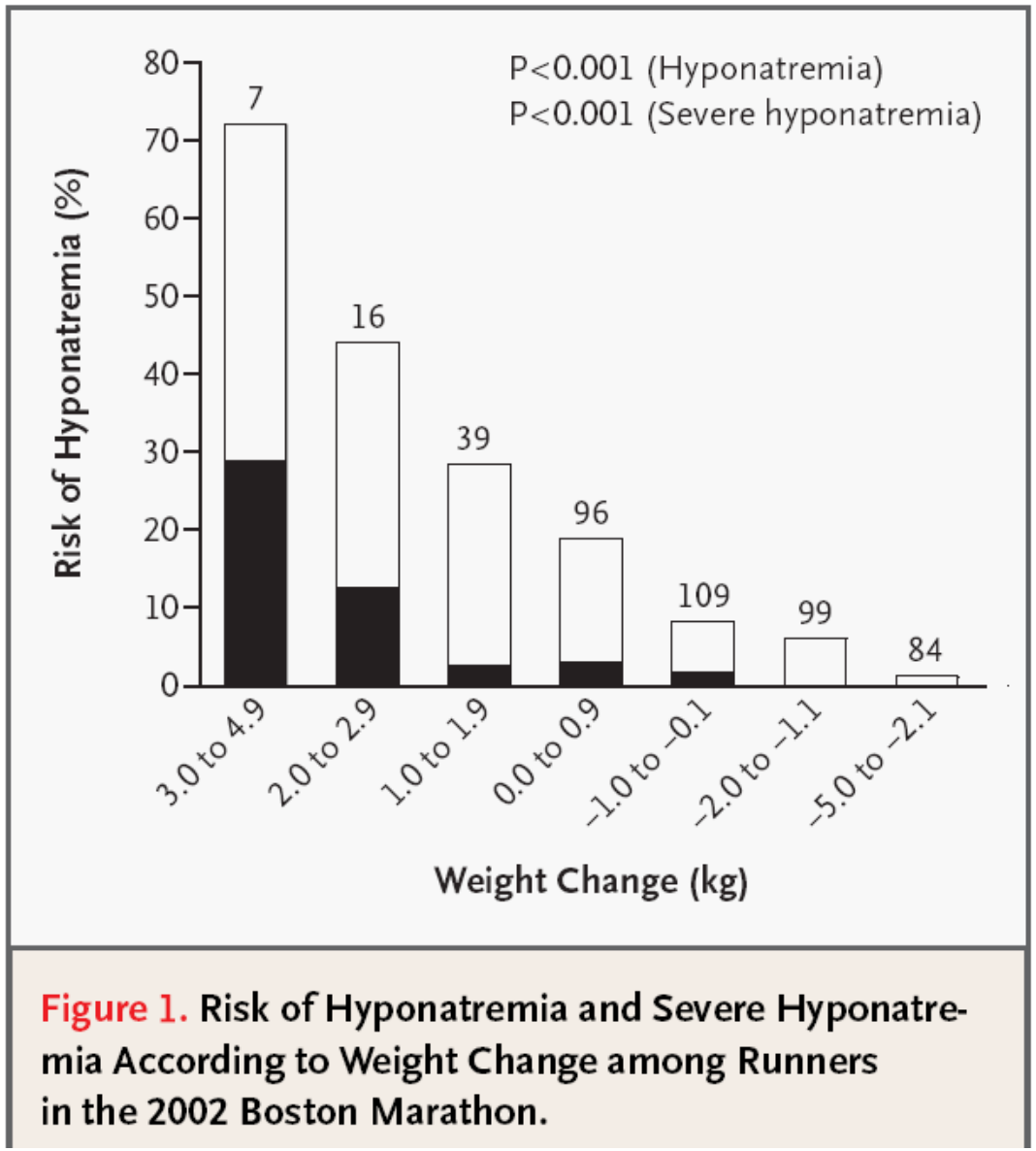


Figure 1. Risk of Hyponatremia and Severe Hyponatremia According to Weight Change among Runners in the 2002 Boston Marathon.

Graph the probability of a binary outcome

Is the risk of hyponatremia varying across categories of weight change?

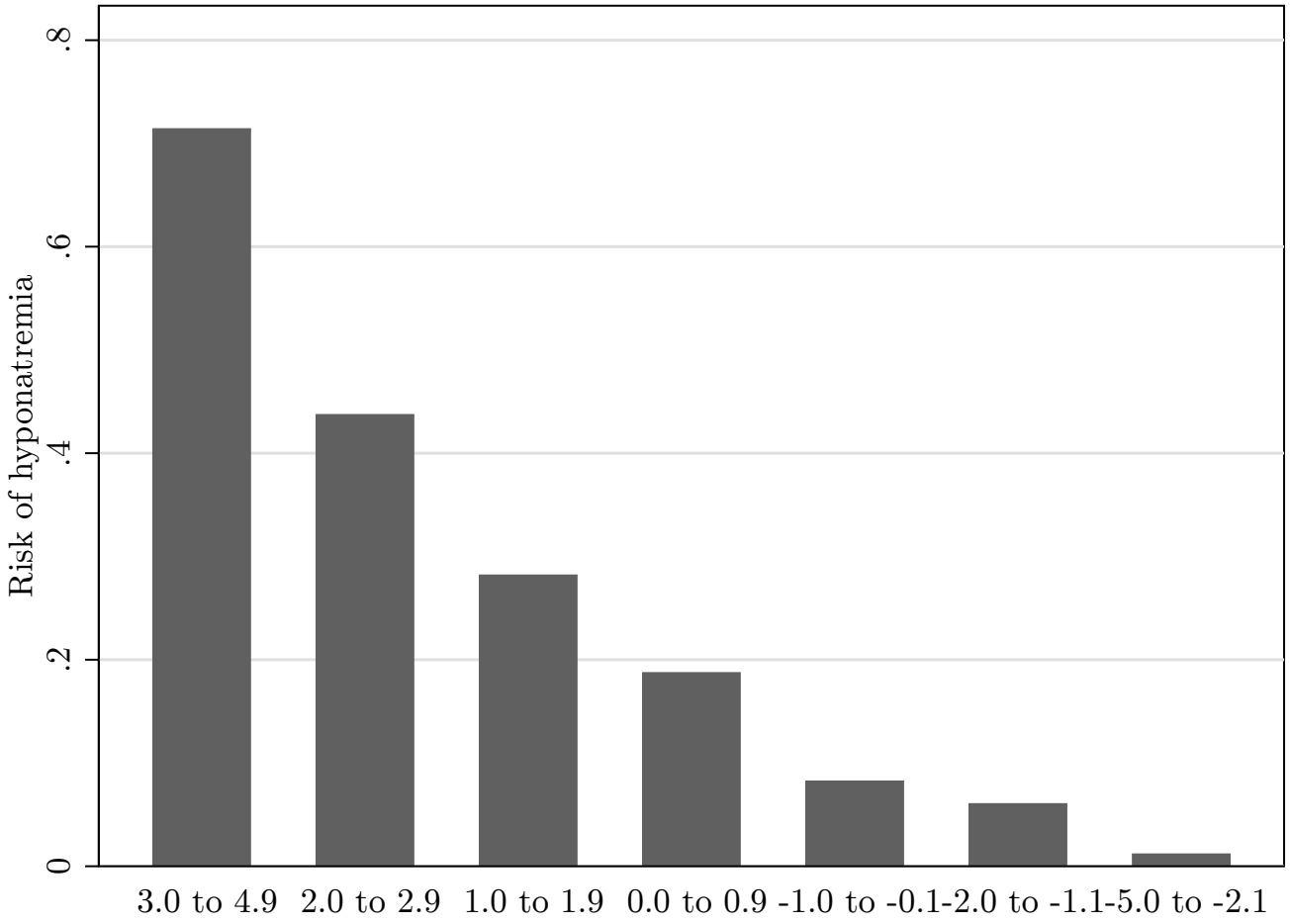
The variable weight change (post-pre race) **wtdiff**, which is continuous, has been categorized in 7 categories (**wtdiffc**).

We can start this analysis by tabulating the observed risk of hyponatremia across categories of weight change.

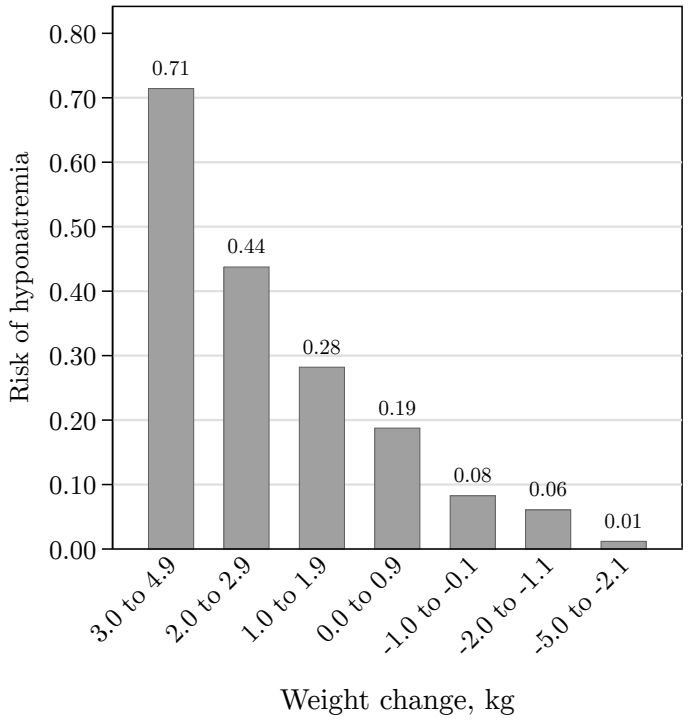
```
. table wtdiffc, c( mean nas135) f(%3.2f)

------------------------------
Categorization |
of weight      |
change         | mean(nas135)
---------------+--------------
   3.0 to 4.9  |         0.71
   2.0 to 2.9  |         0.44
   1.0 to 1.9  |         0.28
   0.0 to 0.9  |         0.19
 -1.0 to -0.1  |         0.08
 -2.0 to -1.1  |         0.06
 -5.0 to -2.1  |         0.01
------------------------------
```

The risk of hyponatremia is strongly correlated with weight gain during the race.

.graph bar (mean) nas135, over(wtdiffc) ///
ytitle(Risk of hyponatremia)



```
. graph bar nas135 , over(wtdiffc, label(angle(45))) ///
        ytitle("Risk of hyponatremia") ///
              bar(1 , fcolor(gs10)) ///
           scheme(s1mono) blabel(bar, format(%3.2f))  ///
       ylabel(0(.1).8, angle(horiz) format(%3.2fc)) aspect(1) ///
    b1title("Weight change, kg")
```



graph bar is a good starting point but to reproduce the published Figure 1 of the paper we need to use twoway command.

We create a new variable containing the risk of hyponatremia (white bars) within each category of weight change and express it as percentage (help egen)

```
egen prna = mean(nas135) ,  by(wtdiffc)

replace prna = prna*100
```

We next create a binary variable to identify subjects with severe hyponatremia (serum sodium concentration at race completion, ≤130 mmol per liter)

```
gen na130 = na <= 130 if na != .
```

As we did before, we create a variable containing the risk of severe hyponatremia (heights of black bars) within each category of weight change and express it as percentage

```
egen prna130 = mean(na130)  ,  by(wtdiffc)

replace prna130 = prna130*100
```

Now we have the variables (y and x) to produce a plot.

We are now ready to reproduce Figure 1 overlaying different types of -twoway- plots.

A customized graph is going to be a long line of code.

Twoway graph

twoway *plot* [if] [in] [, *twoway_options*]

where *plot* is defined

(*plottype* varlist , options)

and where *plottype* (**help twoway**) can be, **among many others**

plottype	**description**
scatter	scatterplot
bar	bar plot
line	line plot
area	line plot with shading
lfit	linear prediction plot

Anatomy of twoway

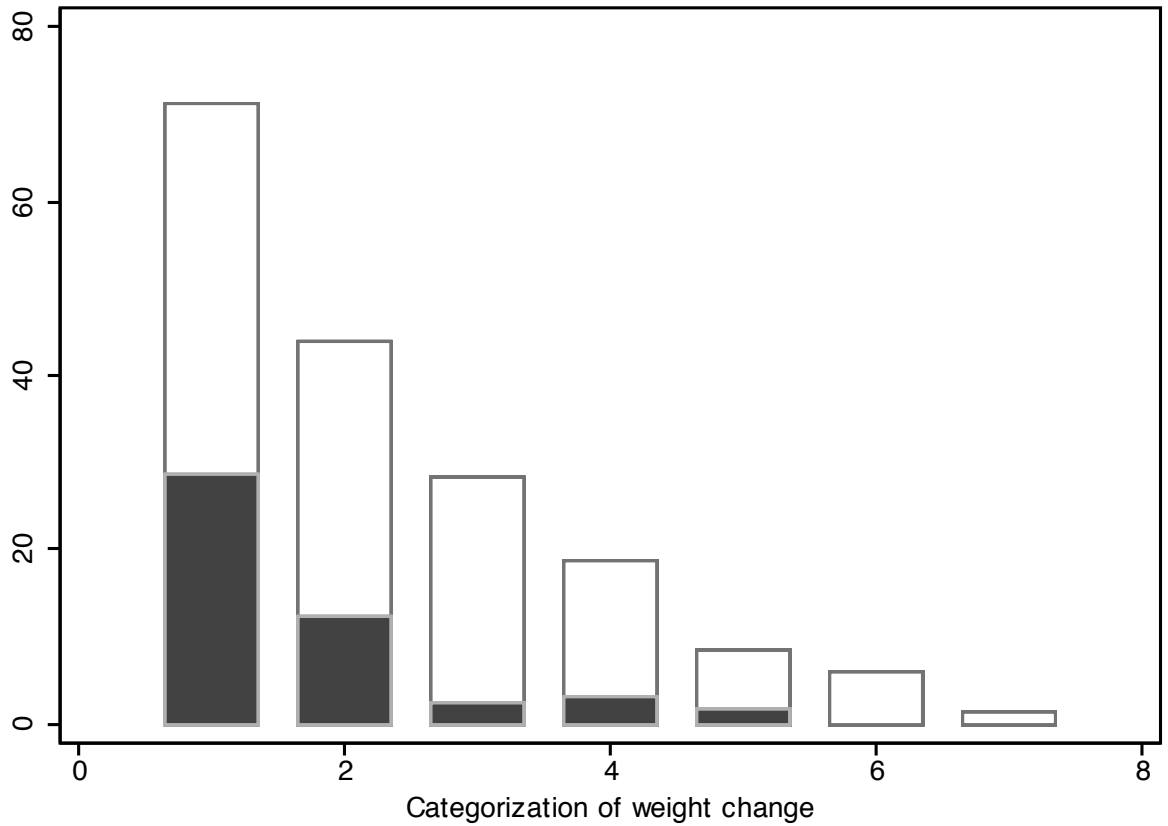
twoway ///
(*plottype1* , *options*) ///
(*plottype2* , *options*) ///
... overlay as much as you want ...
, *options for the overall graph*

Two-way common options

The most common options are:

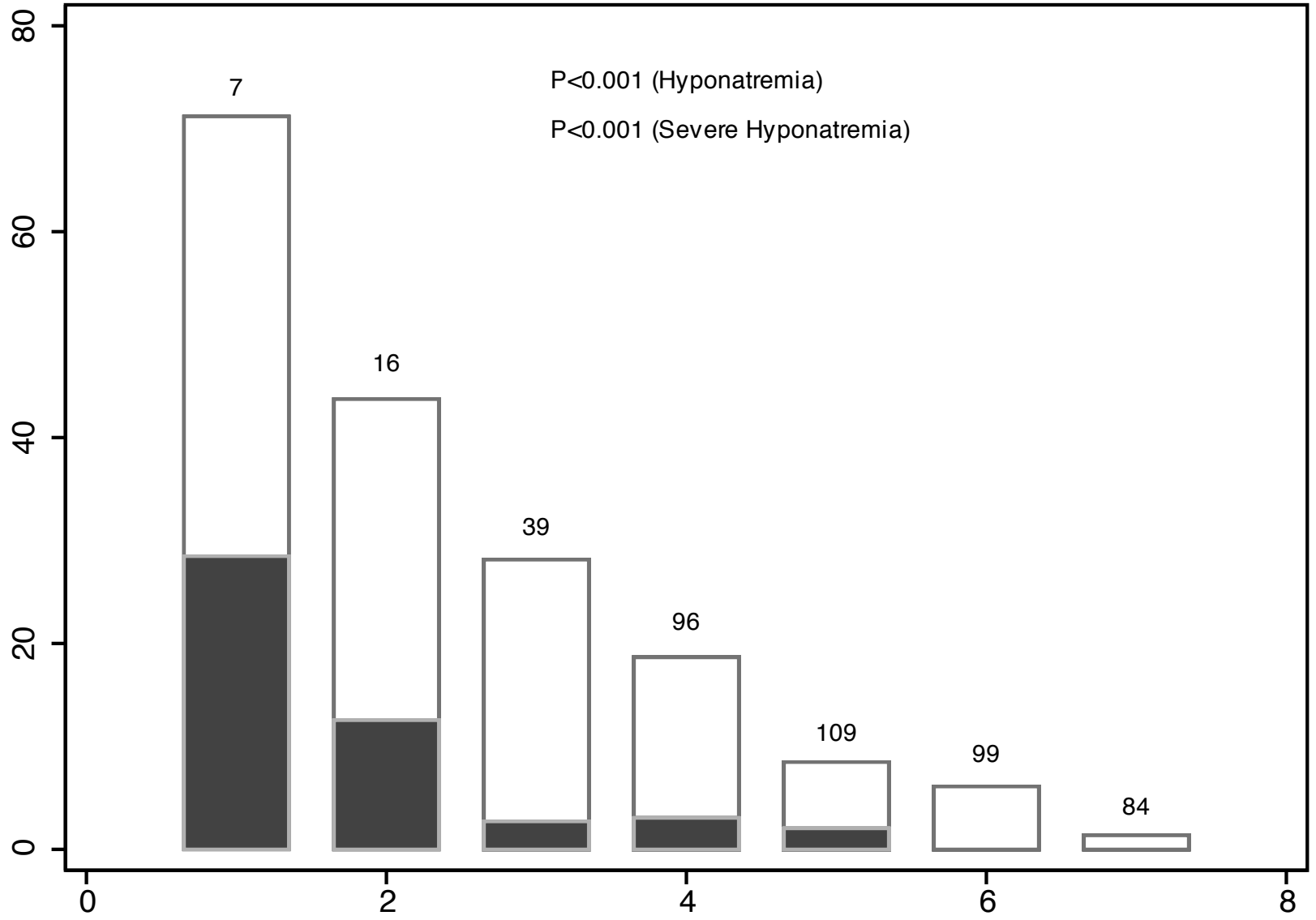
ylabel()	to specify labels on y-axis
xlabel()	to specify labels on x-axis
ytitle()	to specify a title on y-axis
xtitle()	to specify a title on x-axis
scheme()	to specify the overall look of the graph
legend()	to specify the legend of the graph

```
twoway ///
      (bar prna wtdiffc, barw(.7) fcolor(white) ) ///
      (bar prna130 wtdiffc, barw(.7) fcolor(black)) , ///
            scheme(s1mono) ///
            legend(off)
```



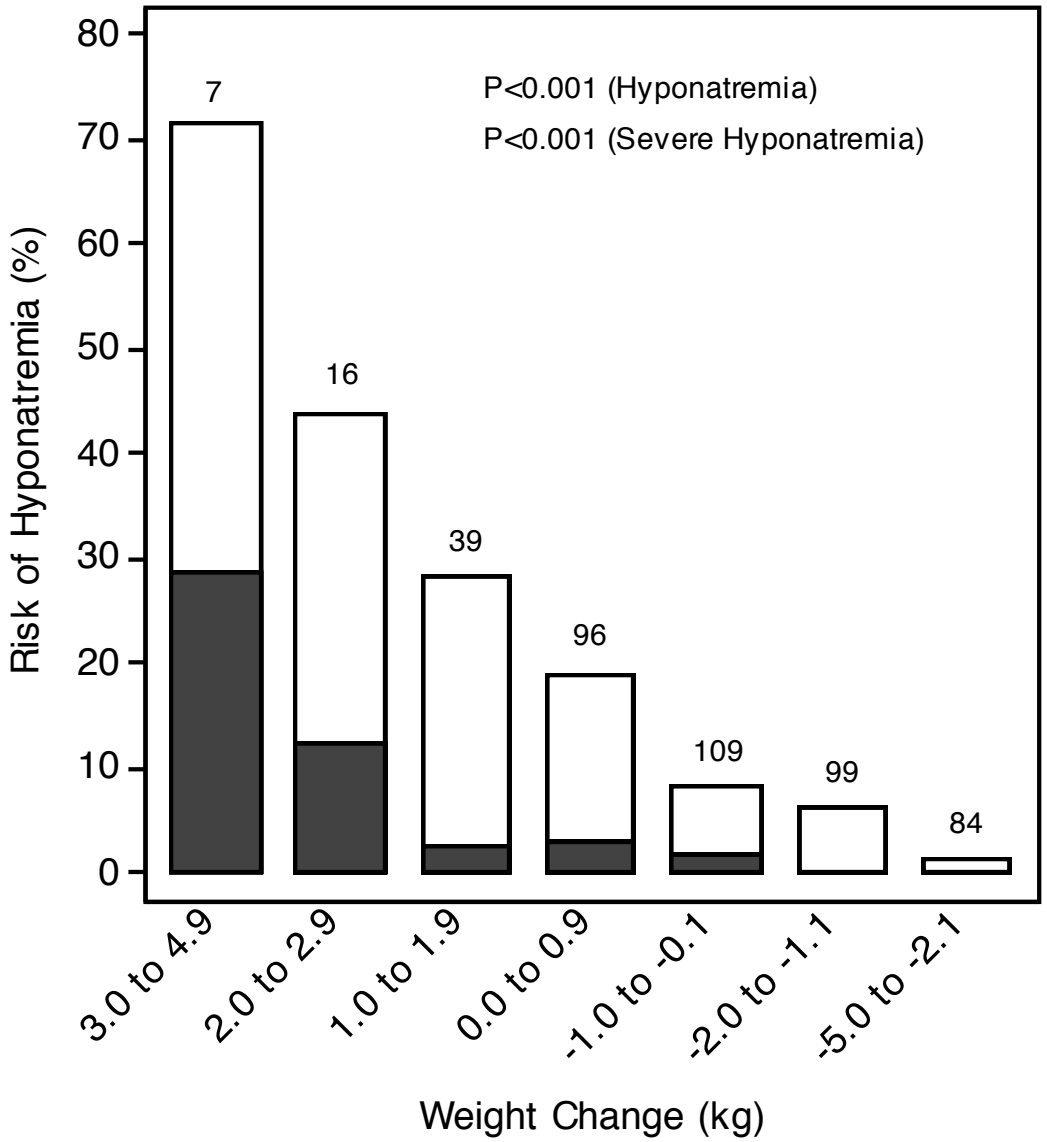
Once you get the basic picture of the graph you can add text and numbers on the plot region using the scatteri plot command. Just specify the coordinates and what you want to display.

```
twoway ///
            (bar prna wtdiffc, barw(.7) fcolor(white) ) ///
            (bar prna130 wtdiffc, barw(.7) fcolor(black)) ///
            (scatteri 72 1 (12) "7" , ms(i)  mlabc(black)) ///
            (scatteri 45 2 (12) "16" , ms(i) mlabc(black)) ///
            (scatteri 29 3 (12) "39" , ms(i) mlabc(black)) ///
            (scatteri 20 4 (12) "96" , ms(i) mlabc(black)) ///
            (scatteri 9 5  (12) "109", ms(i) mlabc(black)) ///
            (scatteri 7 6  (12) "99" , ms(i) mlabc(black)) ///
            (scatteri 2 7  (12) "84" , ms(i) mlabc(black)) ///
 (scatteri 75 3 (3) "P<0.001 (Hyponatremia)" , ms(i) mlabc(black)) /// (scatteri 70 3(3) "P<0.001
(Severe Hyponatremia)", ms(i) mlabc(black)) ///
, scheme(s1mono) legend(off)
```


P<0.001 (Hyponatremia)
P<0.001 (Severe Hyponatremia)
7
16
39
96
109
99
84
0
20
40
60
80
0
2
4
6
8

We next take care of whatever is outside the plot region, namely what is the Y and X-axis of the two-way plot.

```
twoway ///
(bar prna wtdiffc, barw(.7) fcolor(white) lcolor(black)) ///
(bar prna130 wtdiffc, barw(.7) fcolor(black) lcolor(black)) ///
(scatteri 72 1 (12) "7" , ms(i)  mlabc(black)) ///
(scatteri 45 2 (12) "16" , ms(i) mlabc(black)) ///
(scatteri 29 3 (12) "39" , ms(i) mlabc(black)) ///
(scatteri 20 4 (12) "96" , ms(i) mlabc(black)) ///
(scatteri 9 5  (12) "109", ms(i) mlabc(black)) ///
(scatteri 7 6  (12) "99" , ms(i) mlabc(black)) ///
(scatteri 2 7  (12) "84" , ms(i) mlabc(black)) ///
(scatteri 75 3 (3) "P<0.001 (Hyponatremia)" , ms(i) mlabc(black)) ///
(scatteri 70 3 (3) "P<0.001 (Severe Hyponatremia)", ms(i) mlabc(black))///
, scheme(s1mono) ///
legend(off) ///
ylabel(0(10)80, angle(horiz) ) ///
ytitle("Risk of Hyponatremia (%)") ///
xtitle("Weight Change (kg)") ///
xlabel(1(1)7,valuelabel noticks angle(45)) ///
aspect(1)
```


P<0.001 (Hyponatremia)
P<0.001 (Severe Hyponatremia)
Risk of Hyponatremia (%)
80
70
60
50
40
30
20
10
0
7
16
39
96
109
99
84
3.0 to 4.9
2.0 to 2.9
1.0 to 1.9
0.0 to 0.9
-1.0 to -0.1
-2.0 to -1.1
-5.0 to -2.1
Weight Change (kg)

Enter the data from keyboard

Variable	Univariate Predictors			Multivariate Predictors	
	Hyponatremia (N=62)	No Hyponatremia (N=426)	P Value†	Odds Ratio (95% CI)	P Value†
Category of race duration (hr:min)			<0.001		
<3:30 (%)	13	44	—	1.0§	—
3:30–4:00 (%)	35	31	—	3.6 (1.4–11.5)	0.01
>4:00 (%)	52	25	—	7.4 (2.9–23.1)	<0.001

Entering few lines of data using the keyboard can be useful when working with published data or making a graph of few data points.

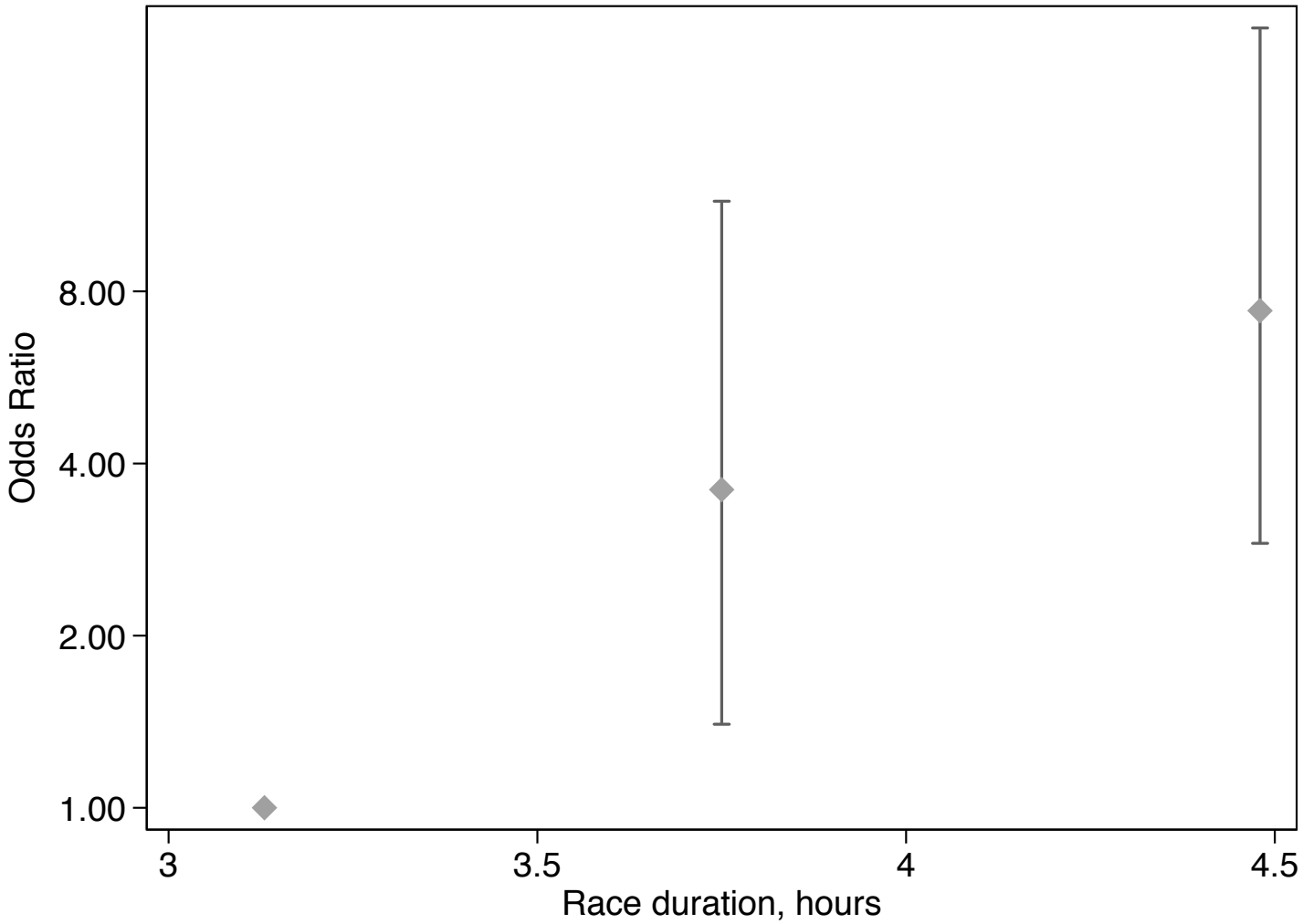
* Input the published data

```
clear
input or      lb      ub          dose
      1       1       1           3.13
      3.6     1.4     11.5        3.75
      7.4     2.9     23.1        4.48
end
```

The variable dose is the median running time in hours within each exposure interval.

* Plot the published data

```
twoway (rcap lb ub dose) ///
 (scatter or dose, sort) , ///
scheme(s1mono) ///
ytitle("Odds Ratio") ///
xtitle("Race duration, hours") ///
legend(off) yscale(log) ///
ylabel(.5 1 2 4 8, angle(h) format(%3.2fc))
```


Odds Ratio
8.00
4.00
2.00
1.00
3
3.5
4
4.5
Race duration, hours

Select variables and observations

drop eliminates variables that are explicitly listed

keep keeps variables that are explicitly listed (opposite of **drop**)

drop *varlist*
drop if *exp*
drop in *range* [if *exp*]

keep *varlist*
keep if *exp*
keep in *range* [if *exp*]

// Examples

* eliminate the variables wtdiffc and urinat3p
drop wtdiffc urinat3p

* eliminate the first 10 observations
drop in 1/10

* keep 3 variables in memory
keep id nas135 wtdiff

* keep only women
keep if female == 1

Sort observations

sort *varlist*

arranges the observations of the current data into ascending order based on the values of the variables in *varlist*

```
// Example: sort rows by running time

sort runtime
```

Prefix by

by *varlist*: *stata_cmd*

bysort *varlist*: *stata_cmd*

repeats the command for each group of observations for which the values of the variables in ***varlist*** are the same.

The prefix **bysort** gets in one line the **sort** and **by** commands.

// Examples

* sort dataset by wtdiffc

sort wtdiffc

* for each level of weight change summarize na

by wtdiffc: summarize na

* for each level of weight change summarize na

bysort wtdiffc: summarize na

Hand calculator

display *exp*

displays strings and values of scalar expressions. It is used in do-files and programs to produce formatted output.

It can be simply used as a substitute for a hand calculator.

```
// Example

display "The square root of 4 is " sqrt(4)

display sqrt(4)+ 2*log(1)

display exp(0.5)
```

Format

format [*varlist*] *%fmt*

allows you to specify the display format for variables. The internal precision of the variables is unaffected (**help format**)

Among many others

%#.#f **fixed numeric format**

This is really useful to control the number of decimal points.

Import and export of dataset

insheet reads ASCII (text) data created by a spreadsheet (.txt, .csv, .raw).

```
insheet using ///
http://nicolaorsini.altervista.org/data/hyponatremia.txt, clear
```

outsheet writes data into a file in tab or comma-separated ASCII format (.txt, .csv, .xls).

```
outsheet using hyponatremia.xls, replace
```

Summary of the commands

Description	Command
Open and save the dataset Stata format	use
Look at the dataset	describe
	list
	browse
Summary statistics	summary
Table of counts	tabulate
	count
Table of summary statistics	table
	tabstat
Graph distributions and statistics	graph histogram
	graph box
Two-way scatter plot	twoway
	scatter
	line

Data management	generate
	replace
	recode
	insheet
	outsheet