

**Week 1**  
**POL 606**  
**Introduction to RATS**

RATS has its own programming language that we will learn over the course of the semester.

Aside from the commands that are part of the program, there are also RATS procedures that are “source” files. These files have the suffix .SRC and must be read into the program you are working on.

RATS is a great program for time series, but you should stick to STATA for standard cross-sectional analysis. When it comes to pooled cross-sectional analysis, we will try to use RATS but STATA has been working on improving their program in this area.

STATA has been working on their time series aspects overall and could be used for much of what we cover in this class but for the more advanced modeling techniques RATS is best so we will use it.

One important warning about RATS is that it is a very unstable program. It has improved over the last few updates but it still is very prone to crashing. Save your programs often! Especially before you press the print button or the run button.

RATS has Input and Output – they can be in the same window but that gets very confusing. So put them into separate windows. Write your program in one window and designate that “Input.” Then tile the windows so the program is in one window and the output is in the other.

RATS has two settings, Running mode and Local Edit. The R/L button switches back and forth between the two.

When RATS is in running mode, pressing enter while in the Input window will execute the line you are on. So, when you are only interested in editing your program, put rats into Local edit mode and then turn on Running mode when you are ready to run.

You also can highlight those parts of your program you want to run and execute those lines only.

For a run through of some of the basics, use Basics.prg.

This can be found in the “estima” file within “program files.” There are many other programs and data sets there that might be helpful.

(Note that many of the examples in the RATS handbook are not part of the current RATS package.)

The first statement is the CALENDER statement that tells RATS when the data begin and how frequent the observations are.

**calendar** *year period frequency*

*year* is the year of the first entry in the data set.

*period* is the period of the first entry in the data set.

*frequency* the number of observations per year.

calendar 1975 3 12

Means monthly data beginning March 1975.

Other ways to use the calendar command for daily, weekly, and irregularly spaced data.

Next command is ALLOCATE – follows immediately after the CALENDAR command.

Tells RATS when the data end.

**allocate** 2004:04

This tells RATS the data end in April of 2004.

Next, we open the data set. RATS will look automatically for the data in the file folder that the program came from. If the program and the data are in different files, specify where the data are.

open data basics.wks

data(format=wks,org=columns) / rate m1 m2 ip ppi

**format** tells RATS whether the data is in a worksheet, excel, free format or whatever.

**org=columns** means that each column is a series.

The slash is followed by the variable names.

The slash tells RATS to reading in all the variables for the complete time period.

This is generally true. After many commands you can specify a beginning and ending date or just “/” meaning all dates.

We don't need to do this, we can read in only some variables or only some part of the time period.

\* For BASICS.RAT, you would use "data(for=rats) / rate m1 m2 ip ppi"

\* For BASICS.XLS, you would use "data(for=xls,org=columns) / rate m1 m2 ip ppi"

\* For BASICS.PRN, you would use "data(for=prn,org=columns) / rate m1 m2 ip ppi"

\* For BASICS.DAT, you would use "data(for=free,org=columns) / rate m1 m2 ip ppi"

What should each of the above data sets look like?

This varies depending on the format.

I prefer to use .txt files for the data.

I put the data into wordpad in columns and nothing else in the file.

Then the data lines are:

```
open data C:\Research\PresApp\USData.txt
data(format=free,org=columns) / pres npros1 npros2 $
iranp rally19 reagan tien bush
```

Note that when a single statement goes across multiple lines a “\$” appears at the end of the first line and next line is indented one space.

Getting your data in is one of the many things you can use the RATS wizard for, but we won't be using the wizard very much.

Once the data are in memory, we can begin with some basic commands.

```
table
table / ppi rate
table 1970:1 1980:12 ppi rate
```

We can get tables for everything, for only selected variables and for selected variables over selected periods.

```
statistics rate
```

Summary statistics.

```
print / m1 m2
print(picture='*.#') / m1 m2
```

This gives our data with specified decimal places (here it is one). ## = 2 decimal places and so on.

```
print 1960:1 1960:12 m1
```

See the series over a specific period.

## Graphing

```
graph(key=upleft,header='Real Money Supply (M1 and
M2)',subhead='Billions US$, Seasonally Adjusted') 2
# m1
# m2
```

Graphing in RATS is straightforward.

The **graph** command is followed by options and then the number of series to be graphed. This is followed by listing the series:

```
# series
```

Graphing each of your series is a good check of your data. In particular it will show any missing data or very obvious coding errors.

Simply looking at a series can tell us a lot about it – we can see trends, memory, forecasts and stationarity.

EViews is also good, and the new STATA graphing package as well, but RATS graphs are improved and pretty good. The menu can be used for them also.

For data transformations:

The most important transformation is first differencing.

Instead of using a variable in “levels” we look at the changes from one time point to the next.

\* Need first difference of the produce price index.

\* The following two instructions are equivalent:

```
set ppdiff = ppi - ppi{1}
diff ppi / ppdiff
```

\* Now, need (M1(t) - M1(t-3)):

```
set m1diff = m1 - m1{3}
```

```
print(picture='*##') / ppi ppdiff m1 m1diff
```

\* And quarter-to-quarter and annual growth rates for M2 and PPI:

\* M2(t) - M2(t-1) divided by M2(t-1):

```
set grm2 = (m2 - m2{1})/m2{1}
```

\* PPI(t) - PPI(t-1) divided by PPI(t-1):

```
set grppi = (ppi - ppi{1})/ppi{1}
```

\* IP(t) - IP(t-1) divided by PPI(t-1):

```
set grip = 100*(ip - ip{1})/ip{1}
```

\* Annualized growth rates:

```
set anngrppi = 100*( (ppi/ppi{1} )**12 -1.0)
```

```
set anngrip = 100*( (ip/ip{1} )**12 -1.0)
```

\* Need a three period weighted moving average:

```
set pratio = ppidiff/ppi
```

```
set ppisum = pratio + pratio{1} + pratio{2}
```

\* Or:

```
set ppisum = (ppidiff/ppi) + (ppidiff{1}/ppi{1}) + (ppidiff{2}/ppi{2})
```

\* Or:

```
set pratio = ppidiff/ppi
```

```
filter pratio / ppisum
```

```
# 1 2
```

```
# 1.0 1.0
```

\* Simple trend and squared-trend series, commonly used:

```
set trend = t
```

```
set trendsq = t**2
```