Logs and Scripts

What is a log?

Log files keep track of everything that happens in a statistics or data management session: what you type; what the computer calculates and prints; and what goes wrong.

Why log?

You will refer to the log when you prepare your write-up of a quantitative analysis. You will need log files to remember what procedures you used and what the results were and to confirm that the procedures finished as expected.

For example, when you re-coded a 10-point scale of worker satisfaction into a binary variable, did you code responses of ``4 and below'' as dissatisfied or did you code ``below 4'' as dissatisfied--or maybe it was ``below 3''? Without a record of what you did, you may find it very hard to interpret and report your results.

You may refer to the log when you pick up a project from someone else. Let's hope that your predecessor used good logging practice.

Others can analyze your log to help you solve problems and interpret results. ``Something went wrong when I was analyzing the survey'' will elicit much less assistance from experts than will ``The response to the self-reported health question, variable HEA, is missing for all respondents over age 65.''

Others can check, replicate, and extend your work. Your logs will help people working on the same project perform analyses that build on your work.

How to log

1. In RStudio you can capture a session by creating a markdown document.

2. In R, RStudio, or Emacs/ESS you can save the entire Console session by selecting all, copying, and pasting to a text document.

3. You can run R in "batch mode," sending the output to a log file from the command line on Windows, Mac OSX, or Linux. The command

R CMD BATCH my-script.R

will run the content of the script my-script.R and send the output to my-script.Rout This is my preferred method of running scripts and logging once I have a working script.

Name and store your logs carefully. The name should help you remember what is in the log file. For example, if you are working on problem set 3, question 4, part a, you might want to call the log file econ-753-ps-3-4-a.Rout. File location matters, too: organize your projects thematically in a directory structure. This organization is especially important if you are sharing the computer with other users or are working on many projects at once. Pay attention where you put your files; it can be difficult to find them later.

You may want to write notes to yourself in the log file. These are called comments and allow you to record information without taking an action.

In Stata precede the comment with a *. For example,

* In this section, I will work on part (iii) of ps3_4_a

In R, any material following # on a line will be ignored, and it is typical to use ##. For example,

In this section, I will work on part (iii) of ps3_4_a

```
This works even if there is active code on the line. For example,

with(cps82, table(sex,occupation)) ## Tabulate occupations by sex
                                   ## for the workers in the sample

You can view logs in a text editor or in a word processor.  Modern
integrated development environments make it possible to output results
directly into a formatted document, but logs remain useful in
examining output and looking for problems.


What is a script?

A script, also called a batch file, a batch job, a program, or (in the
terminology of Stata) a do-file, is a file containing a set of
commands that the statistical application will process in sequence.

Why script and when?

Although R is a ``conversational'' application (which means that you
can give a command from the keyboard, and R shows you the result and
then awaits your next command), I recommend scripting most of your
work.  The ``conversational'' approach is ok if you are just beginning
to explore your data or if you are trying to figure out why a
procedure went wrong, but all serious data-inputting, data-cleaning,
re-coding, subsetting, and statistical procedures should be scripted.

As with logging, scripts keep track of what you have done and make it
possible to replicate your work quickly and accurately.  If you script
most of your work, you can make changes at an early point in an
analysis and modify all subsequent procedures without much difficulty.

For example, suppose that you are conducting an analysis of eldercare
in the Hill Towns: initially you decided to focus on people age 70 and
up, and so you dropped from the original dataset all people under age
70.  Two months into the analysis, you decide that you should include
people age 65 and up.  If you have scripted all of your work along the
way, the modification involves several keystrokes and letting each
script run in sequence.  If you were not scripting, you will need to
start from scratch, review the logs (you were logging, right?), and
key in every coding command that you have given over the past two
months.

Furthermore, scripts become part of an archive for you to accumulate,
use, and share as you conduct quantitative analysis.  You will frequently
perform similar procedures on different datasets, e.g., make categorical
variables from continuous variables, and if you can refer to your earlier
work you will save lots of time.

How to script

1. As with logs, you should choose sensible names and locations for
   your scripts.  You will frequently use scripts in sequence; so it
   may make sense to number the scripts sequentially.

2. RStudio and Emacs/ESS are integrated development environments which
   include script editors. R for Windows or Mac OS has a built-in
   editor for writing scripts.

3. Start every script with comments about who wrote the program, when,
   for what purpose, and how to contact the author.

################################
## ps3_4_a.R                  ##
## Michael Ash, mash@umass.edu ##
## Econ 753                   ##
## Fall 2019                  ##
################################

4. Save the script with ".R" extension, for example, ps3_4_a.R
```

An easy way to begin a script is to log a conversational session and then edit the output, taking out all of the results, etc. You can also write directly into the script and run lines of the script with a hot key.

An Integrated Development Environment (IDE) is a computer program or application that bundles writing scripts, running scripts, and examining the contents and results of data management and statistical procedures.

Conclusion

Logging and scripting are part of responsible record-keeping and good econometric analysis. The small additional effort, of writing and commenting scripts and using the logging commands, pays for itself quickly.

Logs and Scripts

What is a log?

Log files keep track of everything that happens in a statistics or data management session: what you type; what the computer calculates and prints; and what goes wrong.

Why log?

You will refer to the log when you prepare your write-up of a quantitative analysis. You will need log files to remember what procedures you used and what the results were and to confirm that the procedures finished as expected.

For example, when you re-coded a 10-point scale of worker satisfaction into a binary variable, did you code responses of ``4 and below'' as dissatisfied or did you code ``below 4'' as dissatisfied--or maybe it was ``below 3''? Without a record of what you did, you may find it very hard to interpret and report your results.

You may refer to the log when you pick up a project from someone else. Let's hope that your predecessor used good logging practice.

Others can analyze your log to help you solve problems and interpret results. ``Something went wrong when I was analyzing the survey'' will elicit much less assistance from experts than will ``The response to the self-reported health question, variable HEA, is missing for all respondents over age 65.''

Others can check, replicate, and extend your work. Your logs will help people working on the same project perform analyses that build on your work.

How to log

1. In RStudio you can capture a session by creating a markdown document.

2. In R, RStudio, or Emacs/ESS you can save the entire Console session by selecting all, copying, and pasting to a text document.

3. You can run R in "batch mode," sending the output to a log file from the command line on Windows, Mac OSX, or Linux. The command

R CMD BATCH my-script.R

will run the content of the script my-script.R and send the output to my-script.Rout This is my preferred method of running scripts and logging once I have a working script.

Name and store your logs carefully. The name should help you remember what is in the log file. For example, if you are working on problem set 3, question 4, part a, you might want to call the log file econ-753-ps-3-4-a.Rout. File location matters, too: organize your projects thematically in a directory structure. This organization is especially important if you are sharing the computer with other users or are working on many projects at once. Pay attention where you put your files; it can be difficult to find them later.

You may want to write notes to yourself in the log file. These are called comments and allow you to record information without taking an action.

In Stata precede the comment with a *. For example,

* In this section, I will work on part (iii) of ps3_4_a

In R, any material following # on a line will be ignored, and it is typical to use ##. For example,

In this section, I will work on part (iii) of ps3_4_a

```
This works even if there is active code on the line. For example,

with(cps82, table(sex,occupation)) ## Tabulate occupations by sex
                                   ## for the workers in the sample

You can view logs in a text editor or in a word processor.  Modern
integrated development environments make it possible to output results
directly into a formatted document, but logs remain useful in
examining output and looking for problems.


What is a script?

A script, also called a batch file, a batch job, a program, or (in the
terminology of Stata) a do-file, is a file containing a set of
commands that the statistical application will process in sequence.

Why script and when?

Although R is a ``conversational'' application (which means that you
can give a command from the keyboard, and R shows you the result and
then awaits your next command), I recommend scripting most of your
work.  The ``conversational'' approach is ok if you are just beginning
to explore your data or if you are trying to figure out why a
procedure went wrong, but all serious data-inputting, data-cleaning,
re-coding, subsetting, and statistical procedures should be scripted.

As with logging, scripts keep track of what you have done and make it
possible to replicate your work quickly and accurately.  If you script
most of your work, you can make changes at an early point in an
analysis and modify all subsequent procedures without much difficulty.

For example, suppose that you are conducting an analysis of eldercare
in the Hill Towns: initially you decided to focus on people age 70 and
up, and so you dropped from the original dataset all people under age
70.  Two months into the analysis, you decide that you should include
people age 65 and up.  If you have scripted all of your work along the
way, the modification involves several keystrokes and letting each
script run in sequence.  If you were not scripting, you will need to
start from scratch, review the logs (you were logging, right?), and
key in every coding command that you have given over the past two
months.

Furthermore, scripts become part of an archive for you to accumulate,
use, and share as you conduct quantitative analysis.  You will frequently
perform similar procedures on different datasets, e.g., make categorical
variables from continuous variables, and if you can refer to your earlier
work you will save lots of time.

How to script

1. As with logs, you should choose sensible names and locations for
   your scripts.  You will frequently use scripts in sequence; so it
   may make sense to number the scripts sequentially.

2. RStudio and Emacs/ESS are integrated development environments which
   include script editors. R for Windows or Mac OS has a built-in
   editor for writing scripts.

3. Start every script with comments about who wrote the program, when,
   for what purpose, and how to contact the author.

################################
## ps3_4_a.R                  ##
## Michael Ash, mash@umass.edu ##
## Econ 753                   ##
## Fall 2019                  ##
################################

4. Save the script with ".R" extension, for example, ps3_4_a.R
```

An easy way to begin a script is to log a conversational session and then edit the output, taking out all of the results, etc. You can also write directly into the script and run lines of the script with a hot key.

An Integrated Development Environment (IDE) is a computer program or application that bundles writing scripts, running scripts, and examining the contents and results of data management and statistical procedures.

Conclusion

Logging and scripting are part of responsible record-keeping and good econometric analysis. The small additional effort, of writing and commenting scripts and using the logging commands, pays for itself quickly.