Learning SAS

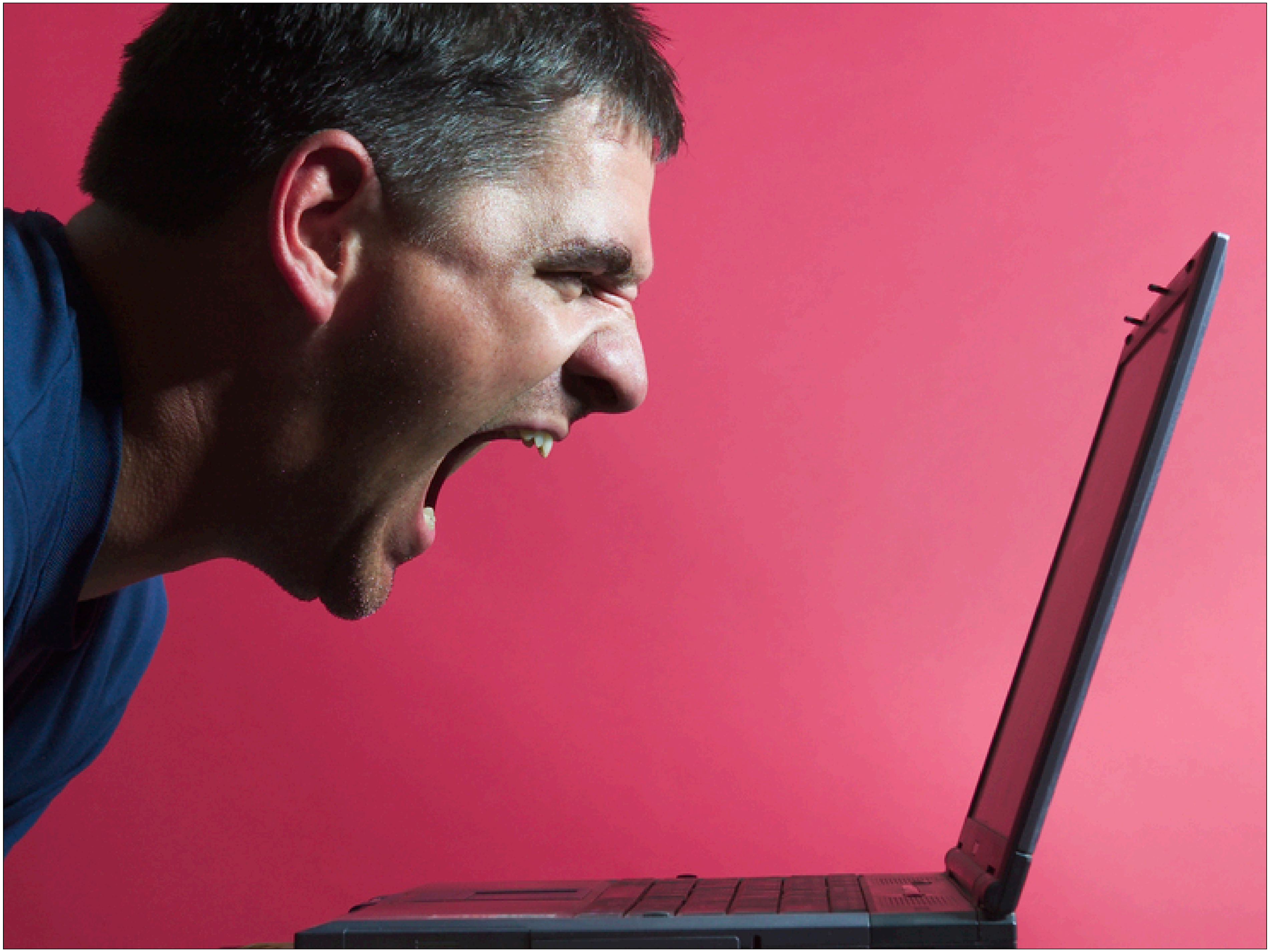
Hadley Wickham

Outline

- Intro & data manipulation basics
- Fitting models x2
- Writing macros
- No graphics (see http://support.sas.com/techsup/sample/sample_graph.html for why)

Today's outline

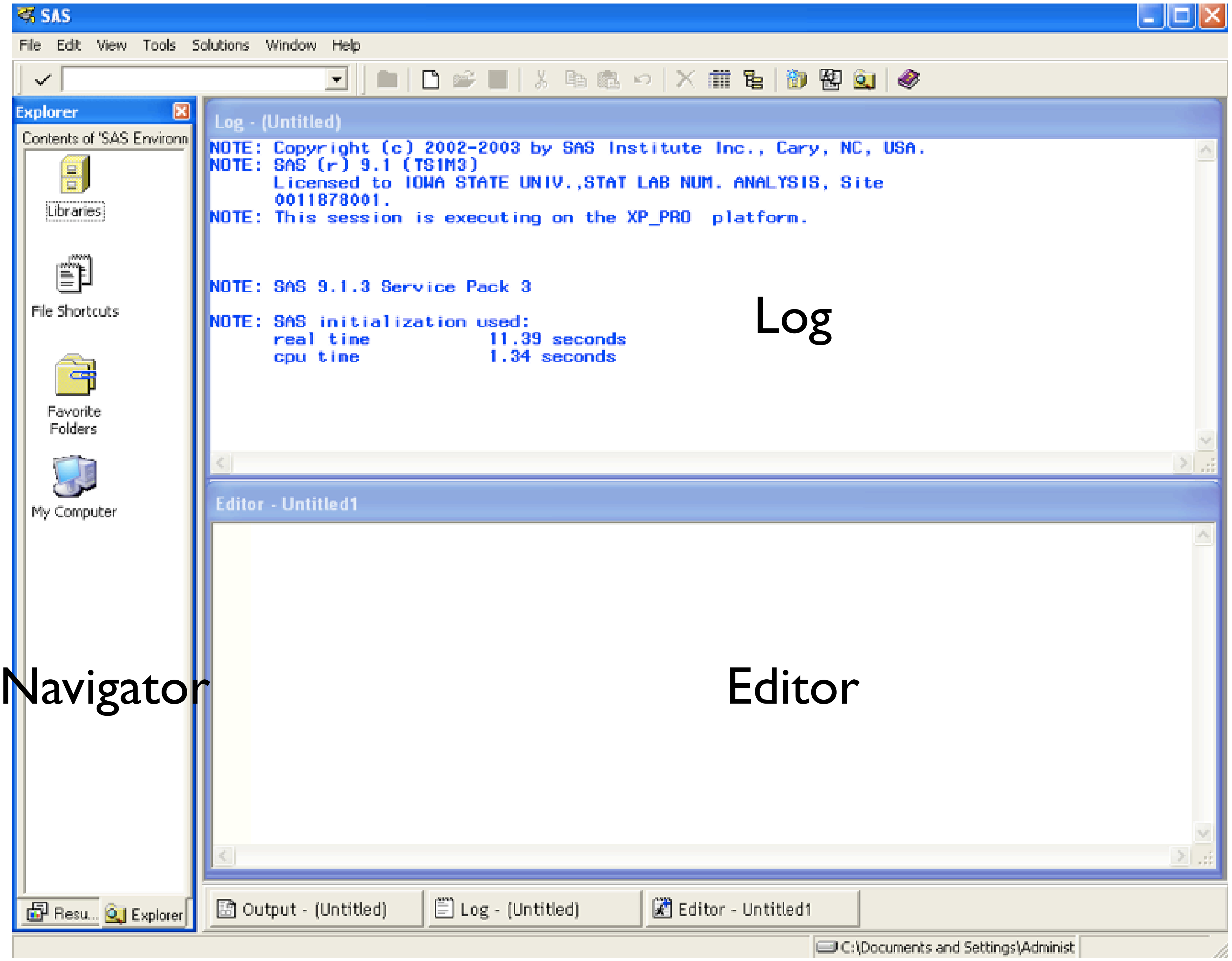
- SAS philosophy
- Intro to the SAS application
- Data import and export
- Modifying and subsetting data



"You must realize that R is written by experts in statistics and statistical computing who, despite popular opinion, do not believe that everything in SAS and SPSS is worth copying. Some things done in such packages, which trace their roots back to the days of punched cards and magnetic tape when fitting a single linear model may take several days because your first 5 attempts failed due to syntax errors in the JCL or the SAS code, still reflect the approach of "give me every possible statistic that could be calculated from this model, whether or not it makes sense". The approach taken in R is different. The underlying assumption is that the useR is thinking about the analysis while doing it."
– Douglas Bates

SAS application

- Four components:
  - editor
  - log
  - output
  - navigator
- Refer back to the following slides



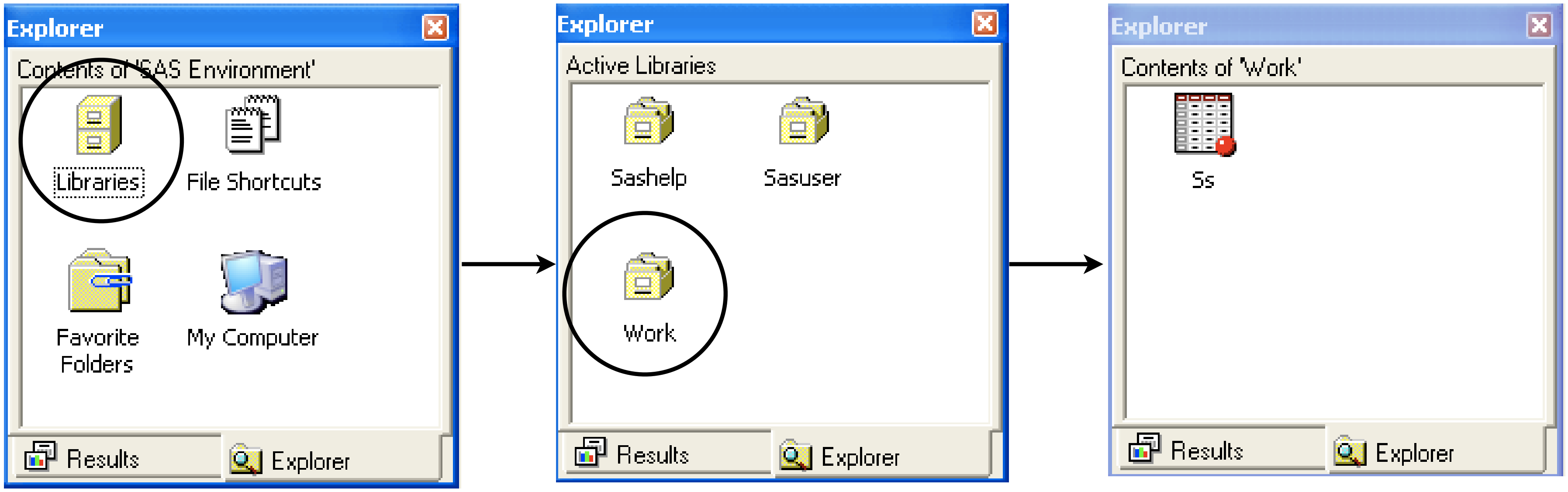
```
NOTE: Copyright (c) 2002-2003 by SAS Institute Inc., Cary, NC, USA.
NOTE: SAS (r) 9.1 (TS1M3)
      Licensed to IOWA STATE UNIV.,STAT LAB NUM. ANALYSIS, Site
      0011878001.
NOTE: This session is executing on the XP_PRO  platform.



NOTE: SAS 9.1.3 Service Pack 3

NOTE: SAS initialization used:
      real time           11.39 seconds
      cpu time            1.34 seconds
```

Navigator


Explorer
Contents of 'SAS Environment'
Libraries
File Shortcuts
Favorite Folders
My Computer
Results
Explorer
Explorer
Active Libraries
Sashelp
Sasuser
Work
Results
Explorer
Explorer
Contents of 'Work'
Ss
Results
Explorer

Log

- Shows you any error messages
- Make sure to always have the log visible!
- It’s the only way you get any feedback

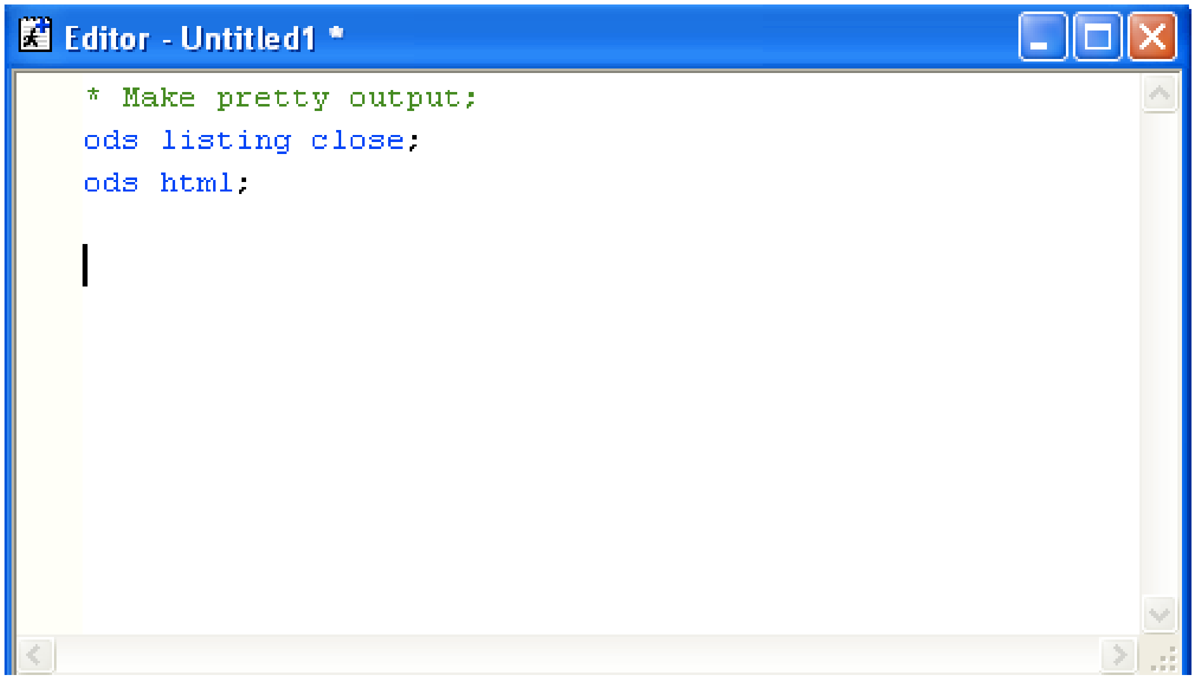
Editor

- Enter commands in here, doesn’t run automatically (like R)
- Select, and press F8 to run
- Comments begin with *
- Need a ; at the end of every statement

First code

- ods listing close;
- ods html;

- (Makes the default SAS code a little bit prettier. You will probably want to use this every time you use SAS)



```
* Make pretty output;
ods listing close;
ods html;
```

Data

- Always keep your data in a standard format (I recommend csv)
- Avoid vendor lock in
- Imagine you want to look at your data again in 10 or 20 years time
- Only use program specific data structures to cache slow operations

R

- You should be familiar with read.csv by now
- The opposite is write.csv

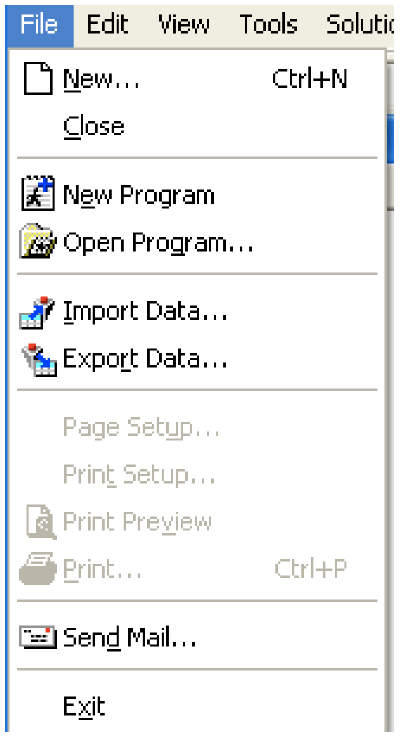
- `df <- read.csv(file.choose())`
- `write.csv(df, ”c:/path/to/file.csv”)`

SAS

-
  ```
  proc import out=df
  datafile=”c:/path/to/file.csv”
  dbms = csv;
  run;
  ```

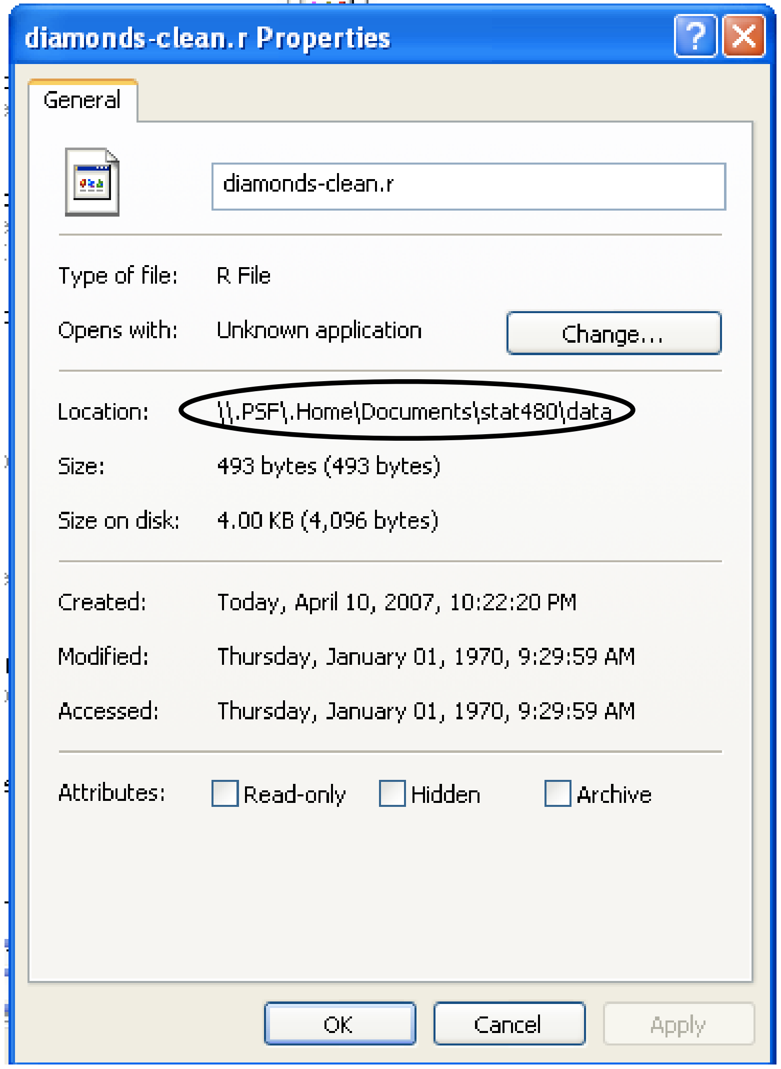
-
  ```
  proc export data=df
  outfile=”c:/path/to/file.csv”
  dbms = csv;
  run;
  ```

SAS



- Or use the menus
- A few other tips:
  - If the file is open in Excel, SAS won't open it
  - Sometimes SAS will just refuse to open a file for no good reason. If that happens try switching file formats (eg. to xls)

How to get file paths


diamonds-clean.r Properties

General

diamonds-clean.r

Type of file: R File

Opens with: Unknown application

Location: \\.PSF\.Home\Documents\stat480\data

Size: 493 bytes (493 bytes)

Size on disk: 4.00 KB (4,096 bytes)

Created: Today, April 10, 2007, 10:22:20 PM

Modified: Thursday, January 01, 1970, 9:29:59 AM

Accessed: Thursday, January 01, 1970, 9:29:59 AM

Attributes: ☐ Read-only ☐ Hidden ☐ Archive

- Right-click | properties
- Can select it and copy and paste
- But won't work in R: Need to replace \ with / or \\
- Instead, you can use basename(file.choose())

Your turn

- Load reshape library, and export the french fries data
- Import it into SAS
- Export it out of SAS
- Import it back into R
- Is it still the same? (look at str)

Uh oh!

- There's a problem with missing values: R uses NA, SAS uses .
- The easiest way to fix this is from R
- Both read.csv and write.csv have a `na` argument, set it to "." for SAS input and output
  - `write.csv(..., na =".")`
  - `read.csv(..., na=".")`

Working directories

- You may have noticed you had to write out a directory name an awful lot of names in that last example
- Both R and SAS have a way to avoid that, called **working directories**
- Lets you specify just the file name
- **Always** use when you're working on a project!

R

- Before:
  - `df <- read.csv("c:/path/to/file/in.csv")`
  - `write.csv(df, "c:/path/to/file/out.csv")`
- After
  - `setwd("c:/path/to/file/")`
  - `df <- read.csv("in.csv")`
  - `write.csv(df, "out.csv")`

SAS

- Before:
  - `proc import out=df datafile="c:/path/to/file/in.csv" dbms = csv; run;`
  - `proc export data=df outfile="c:/path/to/file/out.csv" dbms = csv; run;`
- After
  - `x ' cd "c:/path/to/file/" '`
  - `proc import out=df datafile="in.csv" dbms = csv; run;`
  - `proc export data=df outfile="out.csv" dbms = csv; run;`

Your turn

- Create a new directory called frenchfries, and set working directories in R and SAS to it
- Now, export from R, import into SAS, export out of SAS and import in R
- Make sure to adjust for missing values!

Modifying data

- Our first SAS programming!
- Using the data step
    - `data output_dataset_name;`
    - `set input_data_set_name;`
    - `run;`

Using help

- It’s a bit tricky! Don’t EVER use the search
- You have to navigate to what you want:
  - Data: SAS products | Base SAS | SAS Language concepts | DATA Step concepts
- We’ll talk about more places as we get to them

Data in SAS

- R processes data by columns, SAS processes by rows
- In the data step we can do all sorts of complicated things, but we’re just going to stick with simple variable transformations

```
x 'cd "C:\Documents and Settings\Administrator\Desktop"'
proc import out=ss

datafile="server-survey-sas.csv" dbms = csv; run;

data ss;
set ss;
age = 2007 - birth_yr;
lifeworked = yrs_experience / age;

group = "unknown";
if (age ^= . & age <= 20) then group = "teenage";
if (age > 20 & age <= 30) then group = "20s";
if (age > 30 & age <= 50) then group = "middle-aged";
if (age > 50) then group="elderly";

run;
```

Your turn

- Run that code for yourself
- Experiment with transformations
- Read the help on the data step:
  SAS products > base SAS > SAS language concepts > DATA Step concepts > DATA step processing

Subsetting

- Similar to R
- Three ways:
  - Create new data set
  - Use subset in procedure x2

```
data teenagers;
set ss;
where group = ”teenage”;
run;

proc print data=teenagers;
run;
```

```
proc print data=ss;
where group eg "teenage";
run;

proc print data=ss(where=(group = "teenage"));
run;
```

Differences from R

- ^= instead of !=
- x in (1 4 5) instead of x %in% c(1,4,5)
- Missing values equal negative infinity

Your turn

- Practice your subsetting!