Macros in SAS

Luke & Daiva

Date in the YYYY-MM-DD

Welcome to our Data-related workshop

Purpose:

To teach a few tips and tricks for more efficiently managing your data, tracking your computer files, understanding appropriate analytical approaches, and speeding up the process from code to tables.

Welcome to our Data-related workshop

Purpose:

To teach a few tips and tricks for more efficiently managing your data, tracking your computer files, understanding appropriate analytical approaches, and speeding up the process from code to tables.

Significance:

Topics we cover will help you get more comfortable with data, reduce the chance of overlooked errors, and give you more control over your work. They are also all important parts of a science movement gaining increasing attention – Reproducible Research.

Caveat: We aren't here to teach statistics

Need help with stats? Use these resources!

- U of T Statistical Consulting Services (click here)
- `http://www.stackoverflow.com`
- `http://stats.stackexchange.com`

Overview of other workshops?

- ODS
- Code review club

Notes and help during this workshop

Go to this website:

`https://etherpad.mozilla.org/dnsWorkshops`

What is a macro?

- Macro = macroinstruction
- Set of instructions in an abbreviated format (i.e. condensed code)
- Specifies how an input sequence should be mapped to an output sequence, according to a defined procedure
- Similar to “Find and Replace” feature

What is a macro variable?

- Method of organizing your code to cut down on typing
- %let macrovar = var1 var2 var3
- & calls a macrovariable
- Not a true macro step

```
proc print data = SASdataset;
var = &macrovar;
run;
```

4 steps to making a macro

1. Know what you want the macro to accomplish
   - Data organization
   - Statistical analysis
   - Output printing
   - Any/all of the above
2. Type the code you want to run
   - Data step
   - Proc step (proc corr, proc glm, proc contents, proc print)
3. Add macro commands and variables to Step 2
   - %macro
   - %mend (= *macro end*)
4. Save your macros in a separate file

Not using a macro

Want to test association between caffeine intake and 4 different genetic variants (CYP1A2, ADORA2A, DRD2, 5HT2RA):

```
proc glm data=genes;
    class CYP1A2 sex smoke;
    model caff = CYP1A2 BMI sex smoke;
    lsmeans/ stderr;
run;

proc glm data=genes;
    class ADORA2A sex smoke;
    model caff = ADORA2A BMI sex smoke;
    lsmeans/ stderr;
run;

proc glm data=genes;
```

Why is that undesirable?

- Risk of making typos and overlooking them
- File of your code can become very long
- Not ideal for sharing with others, especially once you leave your lab

Using a macro variable

Want to test association between caffeine intake and 4 genetic variants (CYP1A2, ADORA2A, DRD2, 5HT2RA):

```
%let gene = CYP1A2 ADORA2A DRD2 5HT2RA;
proc glm data=genes;
    class &gene sex smoke;
    model caff = &gene BMI sex smoke;
    lsmeans/ stderr;
run;
```

Using a macro

```
%macro glm (predictors, class);
proc glm data=&data ;
        class &class;
        model caff &predictors;
        lsmeans/ stderr;
        run;
%mend glm;

%glm(data= ,
class = ,
predictors = CYP1A2 ADORA2A DRD2 5HT2RA);
```

Main Exercise

- Work in pairs
- Make a repetitive code without using a macro
- Improve this code by making a macro (as a separate file)
- Note differences between the two files
- Try applying a macro to your own data

Thanks!

- Next time: Combining macros with ODS = power!