Paper 274-2010

A Grad Student 'How-To' Paper on Grant Prep: Preliminary Data, Power Analysis, and Presentation

Elisa L. Priest[1,2] , Derek Blankenship[1]
[1] Institute for Health Care Research and Improvement, Baylor Health Care System
[2] University of North Texas School of Public Health, Epidemiology Department

ABSTRACT

Grant preparation is often challenging due to short timelines, changing study designs, and sample sizes limited by logistics and/or budget constraints. We recently submitted a U-19 grant consisting of five projects that required a systems biology study approach and had a substantial amount of preliminary data. Tasks included analyzing preliminary data, creating publication ready figures, and calculating power for each project within one week.

The systems biology approach required the investigation of 400+ outcome parameters for preliminary and power analyses. We used MACRO programming with PROC MIXED for longitudinal data analysis and ODS OUTPUT to produce summary tables and presentation quality figures for each outcome.

Further, we created a flexible MACRO simulation to estimate power in each project dependent on differences in sample size, number of time points, mean, and between and within subject variances. We used ODS OUTPUT to create summary tables and power curves for each project.

In conclusion, we used SAS® MACROs, STATS, and ODS OUTPUT to efficiently organize, analyze, and present preliminary results and power calculations for multiple projects quickly.

INTRODUCTION

Grant preparation is often challenging due to the short timelines, changing study designs, and sample sizes limited by logistics and/or budget constraints. We recently submitted a U-19 grant consisting of five projects that proposed a systems biology study approach to understand immunological responses to treatment over time. Our tasks included aiding in the study design, determining the appropriate statistical methods for each aim, analyzing preliminary data, creating publication ready figures, and calculating power for each project. Our timeline was *one week* total to complete all the tasks for each of the five projects.

The systems biology approach involved the investigation of 400+ immunological outcome parameters for preliminary and power analyses. Variables were obtained from antibody titers, Luminex, flow cytometry, and microarray platforms. We used MACRO programming with PROC MIXED for longitudinal data analysis, PROC PLOT and SGPLOT, and ODS OUTPUT to produce summary tables and presentation quality figures for each outcome.

Further, we created a flexible MACRO simulation to determine the power of a Wald tests statistic for the between subject variance within a longitudinal linear mixed model analysis which was dependent on sample size, number of time points, mean, and between and within subject variances. We used ODS OUTPUT to create summary tables and power curves for each project.

This paper continues the 'Grad Student How-To' series[1,2] and gives graduate students tools for grant preparation. The tools we used during our one week grant-a-thon included 1) %MACRO programming to automate SAS/STAT analyses and plots for multiple parameters, 2) ODS OUTPUT to create summary datasets and summary documents, 3) PROC GPLOT and PROC SGPLOT/SGPANEL to produce presentation-ready graphs to include in the grant, and 4) EXCEL control documents to control the variable names, labels, and lists. With these tools we were able to successfully meet our short timeline and be flexible enough to meet the changing requirements as the projects evolved. This paper is appropriate for users familiar with basic macro programming.

In conclusion, we used SAS MACROs, STATS, GRAPH, and ODS OUTPUT to efficiently organize, analyze, and present preliminary results and power calculations for 5 projects in a minimal amount of time.

OUTLINE

1. Preliminary Analysis I: Assessing the effect of treatment over time.
2. Preliminary Analysis II: Assessing the steadiness of variables over time.
3. Power Simulation: Estimating power for a between subject variance in a longitudinal analysis.

1. PRELIMINARY ANALYSIS I

One of the first tasks in the grant preparation was to do an analysis of preliminary data. A pilot study was conducted where subjects received either treatment or placebo. Biological samples were taken pre-treatment (days -7, -4, 0) and post treatment (positive days). Data was available from several types of experiments. For the purpose of this paper, we will focus on the flow cytometry data. This data included values for 58 variables across the multiple time points. First we calculated and plotted all the means (+/- one standard deviation) of the variables across time. Next, we used a linear mixed model analysis to evaluate whether there were significant differences between pre and post time points.

- Means
- Individual Variable Mean Plots
- PROC MIXED Models: Assessing outcomes over time

MEANS

We used an excel control document for variable names, labels, and categories. We selected all the distinct variable names into a macro variable &LIST separated by ' '. (Code 1.1). This created one long list of all the variable names so we did not have to type them.[3] Next, PROC UNIVARIATE used &LIST and ODS OUTPUT to produce values for the mean and standard deviations.[4] (Code 1.2) Then, we restructured the resulting dataset to prepare it for PROC SGPLOT. (Code 1.3, Figure 1.1)

Code 1.1: Create a macro list

```
PROC SQL noprint;
 select distinct NAME
 into :LIST
 separated by " "
 from work.cells_1;
quit;
```

Code 1.2: PROC UNIVARIATE and ODS OUTPUT

```
ODS OUTPUT BASICMEASURES=work.basic_measures;

proc univariate data=work.&input.;
class day;
var &LIST.;
run;

ODS OUTPUT CLOSE;
```

Code 1.3: Restructure data to prepare for plotting

```
data work.basic_measures_all;
set work.basic_measures (rename=( LocValue=Mean));
if LOCMEASURE in ("Mean");
High=mean+varvalue;
Low=mean-varvalue;
Keep varname day mean high low;
run;
```

Figure 1.1: Restructured data

DAY	log (WBC)	High	Low
D -7	1.674619	1.903460838	1.4457771733
D 00	1.488180	1.6854750636	1.2908858959
D 01	1.549683	1.7669600595	1.3324060514
D 03	1.578513	1.7395825943	1.4174428179

INDIVIDUAL VARIABLE MEAN PLOTS

We created a %MACRO to plot the mean and standard deviations calculated from the PROC UNIVARIATE. First, we used PROC SQL to create macro lists of variables (&VAR1 etc) and labels (&LABEL1 etc)[3]. (Code 1.4) Next, we used a looping macro to subset the data by variable and PROC SGPLOT to produce a series plot with standard deviations.[5] (Code 1.5) The plots for all variables were output to one .RTF file with one page for each plot.

Code 1.4: Create macro lists of variables and labels

```
proc sql noprint;
  select NAME as VAR into:VAR1- :VAR&SYSMAXLONG.
  from work.cells_1;
  quit;

proc sql noprint;
  select LABEL as LABEL into:LABEL1- :LABEL&SYSMAXLONG.
  from work.cells_1;
  quit;
```

Code 1.5: ODS OUTPUT and PROC SGPLOT

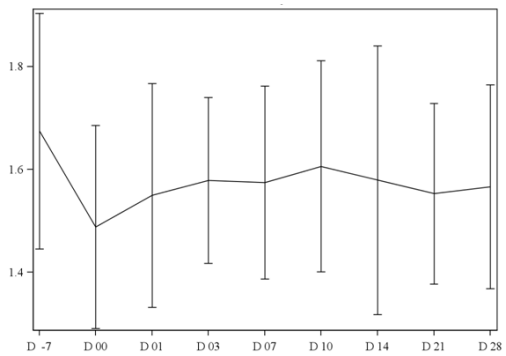
```
ods output;
ods RTF file="C:\documents and settings\desktop\GRAPHS_MEANS.RTF"  ;
%do j=1 %to &SQLOBS.;

data work.values;
set work.basic_measures_all;
if varname="&&VAR&J.";
run;

proc sgplot data=work.values noautolegend;
 xaxis type=discrete;
 series x=day y=mean;
 scatter x=day y=mean /
 markerattrs=(size=0)
 yerrorlower=low
 yerrorupper=high;
run;

%end;
ODS RTF CLOSE;
```

Figure 1.2: PROC SGPLOT



PROC MIXED MODELS: ASSESSING OUTCOMES OVER TIME

We created a third %MACRO to analyze the preliminary data using PROC MIXED and create output datasets of relevant parameters using ODS OUPUT. First, we created macro lists of variables and labels as in Code 1.4. Next, we used PROC MIXED (Code 1.6) and formatted and appended the output datasets. The code for the output dataset work.est is shown. (Code 1.7, Figure 1.3) This code was repeated for the output datasets work.sol and work.tests3. The PROC MIXED code uses a linear mixed model analysis to produce estimates and test statistics for the immunological outcomes between the stated time points, for example week -4 versus week 0. This information was used to determine if and when outcome variables change over time.

Code 1.6: PROC MIXED with ODS OUTPUT

```
ods proclabel "&&VAR&K. &&LABEL&K.";

ods output  Estimates=work.est  SolutionF=work.sol tests3=work.tests3 ;

proc mixed data=work.mixed1 COVTEST cl NOCLPRINT NOITPRINT ;
  class patient_id day_numeric;
  model &&var&k. =day_numeric /solution noint cl residual ;
  repeated day_numeric / subject=patient_id type=ar(1) rcorr r;
  estimate '-4 vs 0'      day_numeric   -1 1 0 0 0 0 0 0;
  estimate '-4 & 0 vs 4'  day_numeric   -0.5 -0.5 1 0 0 0 0 0;
  estimate '-4 & 0 vs 8'  day_numeric   -0.5 -0.5 0 1 0 0 0 0;
  estimate '-4 & 0 vs 12' day_numeric   -0.5 -0.5 0 0 1 0 0 0;
  estimate '-4 & 0 vs 15' day_numeric   -0.5 -0.5 0 0 0 1 0 0;
  estimate '-4 & 0 vs 16' day_numeric   -0.5 -0.5 0 0 0 0 1 0;
  estimate '-4 & 0 vs 22' day_numeric   -0.5 -0.5 0 0 0 0 0 1;
run;
quit;

ods output close;
```

Code 1.7: Format output datasets and PROC APPEND

```
data work.est;
length variable $32 variable_label $32 variable_subset $32;
set work.est;
variable="&&var&k.";
run;

PROC APPEND BASE=work.est_all_&source. DATA=work.est force;
RUN;
```

Figure 1.3: Format output datasets and PROC APPEND

Label	Estimate	Standard Error	DF	t Value	Pr > |t|
-4 vs 0	-1.0191	0.5649	33	-1.80	0.0803
-4 & 0 vs 4	-1.1914	0.4564	33	-2.61	0.0135
-4 & 0 vs 8	-1.1695	0.4281	33	-2.73	0.0100
-4 & 0 vs 12	-4.9451	0.4338	33	-11.40	<.0001
-4 & 0 vs 15	-5.5765	0.4327	33	-12.89	<.0001
-4 & 0 vs 16	-1.1476	0.4329	33	-2.65	0.0122
-4 & 0 vs 22	-6.2175	0.4616	33	-13.47	<.0001

2. PRELIMINARY ANALYSIS II

The next step in the grant preparation was to assess the outcomes' "stability" over time in the placebo group and prepare for power analysis. The measure of the between subject variance relative to the total variance (intraclass correlation coefficient) in linear mixed model analyses were used for the former objective. The power calculations were based off of the Wald z-test for the between subject. Because there was little longitudinal data published on these immunological parameters, we used the preliminary data provided by the researchers to produce estimates for the mean, between subject variance, and within subject variance observed for the parameters. Ultimately, the power calculations were based off one parameter. However, it was still necessary to write a %MACRO for all parameters so that we could base our decision on the most conservative estimates.

- Plots
- PROC MIXED Models: Estimate Variances

PLOTS

We prepared a set of plots to look at the variation in values of each parameter across time for six subjects in the study. We used the same method as presented above in Code 1.4 to create a list of the variables and their labels that we needed to analyze. Originally, we used a standard PROC GPLOT with ODS OUTPUT to produce basic graphs in one RTF file. (Code 2.1, Figure 2.1) However, with the use of PROC TRANSPOSE, the dataset was restructured and PROC SGPANEL was used to produce graphs with an updated look. (Code 2.2, Figure 2.2)

Code 2.1 PROC GPLOT

```
axis1 label=(angle=90);
symbol i=j repeat=100 w=5;

proc gplot data=work.mixed2;
   plot &&var&i.*week_numeric=patient_id/vaxis=axis1 nolegend;
run;
quit;
```

Figure 2.1 Basic GPOT

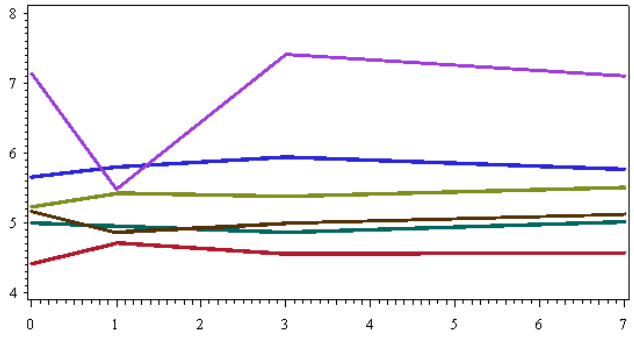


Figure 2.2 Data restructured for PROC SGPANEL

	NAME	DAY	LABEL	_1	_2	_3	_4	_5
1	B_1	D -7	B_1	3.036	6.12	11.28	6.27	5.82
2	B_1	D 00	B_1	8.17	9	15.4	4.389	9.24
3	B_1	D 01	B_1	5.32	3.774	17.08	4.284	6.624
4	B_1	D 03	B_1	4.128	2.226	10.83	3.135	5.628
5	B_1	D 07	B_1	4.802	3.519	11.52	6.014	6.5

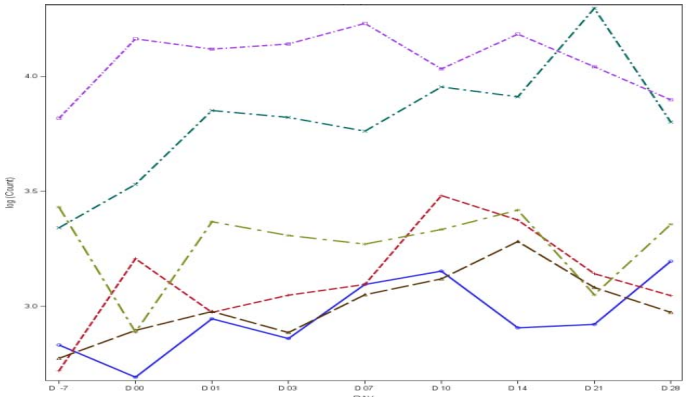
Code 2.2 PROC SGPANEL

```
proc sgpanel data=work.data_transpose_2;
  where _NAME_ in ("&&VAR&I.");
  panelby _LABEL_/novarname ;

  colaxis type=discrete values=(-7,0,1,3,7,10,14,21,28);
  rowaxis label="log (Count)";
  series x=day y=_1 /markers LINEATTRS = (THICKNESS = 3);
  series x=day y=_2 /markers LINEATTRS = (THICKNESS = 3);
  series x=day y=_3 /markers LINEATTRS = (THICKNESS = 3);
  series x=day y=_4 /markers LINEATTRS = (THICKNESS = 3);
  series x=day y=_5 /markers LINEATTRS = (THICKNESS = 3);
  series x=day y=_6 /markers LINEATTRS = (THICKNESS = 3);

run;
quit;
```

Figure 2.3 PROC SGPANEL



PROC MIXED MODELS: ESTIMATE VARIANCES

The previous graphs give us a graphical depiction of the outcome variable's change over time. In addition, we needed a quantitative measure to inform our power calculations. Next, we used PROC MIXED with ODS OUTPUT to produce summary datasets of the covariance matrix estimates of the between subject variance and within subject variance, used to calculate the intraclass correlation coefficient (ICC) for each of the immunologic parameters of interest. (Code 2.3, Figure 2.4) We then used basic data steps followed by PROC PRINT to print the relevant data of interest for each of the parameters. (Code 2.4, Figure 2.5, Figure 2.6)

Code 2.3 PROC MIXED

```
ODS OUTPUT "Covariance Parameter Estimates"=work.parameters;

proc mixed data=mixed2 COVTEST cl NOCLPRINT NOITPRINT ;
  class patient_id ;
  model &&var&i. =day_numeric /solution cl ;
  random intercept / subject=patient_id ;
  run;
  quit;

ODS OUTPUT CLOSE;
```

Figure 2.4 PROC MIXED Covariance Parameter Estimates

	Cov Parm	Subject	Estimate	Standard Error	Z Value	Pr > Z	Alpha	Lower	Upper
1	Intercept	PATIENT_ID	0.04684	0.03033	1.54	0.0612	0.05	0.01795	0.2996
2	Residual		0.009975	0.002058	4.85	<.0001	0.05	0.006913	0.01565

Code 2.4 Restructuring PROC MIXED Covariance Parameter data sets

```
data work.parameters_between;
       set work.parameters;
       if covparm="Intercept" then do;
                     Between_Subject_Variance=Estimate;
                     Between_p_value=ProbZ;
                     Between_Lower=Lower;
                     Between_Upper=Upper;
       end;
       if covparm="Intercept" ;
       keep Between_Subject_Variance Between_P_Value Between_Lower Between_Upper;
       run;

       data work.parameters_within;
       set work.parameters;
       if covparm="Residual" then do;
                     Within_Subject_Variance=Estimate;
                     Within_p_value=ProbZ;
                     Within_Lower=Lower;
                     Within_Upper=Upper;
                     end;
       if covparm="Residual" ;
       keep Within_Subject_Variance WITHIN_P_VALUE Within_Lower Within_Upper;
       run;

       data work.parameters_all;
       merge work.parameters_between work.parameters_within;
       ICC=between_subject_variance/(within_subject_variance+between_subject_variance);
       variable="&&var&i."; variable_label="&&label&i."; data_source="&source.";
       run;
       quit;

       proc print data=work.parameters_all noobs;
       run;
```

Figure 2.5 PROC MIXED Covariance Parameter Estimates Restructured

	Between_Subject_Variance	Between_p_value	Between_Lower	Between_Upper	Within_Subject_Variance	Within_p_value	Within_Lower	Within_Upper	ICC
1	0.0468436446	0.0612277157	0.0179471918	0.2995692167	0.0099751707	6.2456884E-7	0.0069128362	0.0156506196	0.8244389525

Figure 2.6 PROC PRINT for PROC MIXED Covariance Parameter Estimates

Between_ Subject_ Variance	Between_ p_value	Between_ Lower	Between_ Upper	Within_ Subject_ Variance	Within_ p_value	Within_ Lower	Within_ Upper	ICC
0.046844	0.061228	0.017947	0.29957	.009975171	.000000625	.006912836	0.015651	0.82444

3. POWER SIMULATION

A simulation program was written for power calculations because no closed form solution was found for the between subject variance or ICC used in the context of this study. Estimates obtained from the preliminary analysis were used to inform our simulations.

We used a %MACRO to create a dataset of random numbers for seeds across our simulations. Next, we created our longitudinal population data using parameters from proposed study designs. The %MACRO allowed us to be flexible and change our simulation parameters: # of longitudinal time points, size of the population, # of simulations, mean value of the immunological parameter, between subject variation, and within subject variation. We used a macro based SAS/STAT® analysis to simplify analysis, ODS OUTPUT to summarize statistics, and an EXCEL control document to define the simulation parameters. We also used PROC PLOT for our power curves.

- Initialize seeds
- Get next seed
- Create a population
- Analyze and compile the results
- Repeat the simulations for different parameters
- Plots

INITIALIZE SEEDS

First we created a permanent dataset of random values. These random values were used as initial 'seed' values in random number functions when we are creating the population data for the simulation. We calculated how many total seed values were needed based on the size of the population and the total number of simulations we planned. We chose fifteen million total seeds. (Code 3.1, Figure 3.1)

Code 3.1: Initialize seeds

```
%MACRO INITIALIZE ();

%Global Initial;
%Let Initial=0;

data Initialseed (keep=id InitialSeed);
      do I = 1 to 15000000;
            InitialSeed =int(10000*(ranuni(0)));
            id=I;
            output;
      end;
run;

data SIM.InitialSeed;
       set work.InitialSeed;
run;

%MEND INITIALIZE;
```

Figure 3.1 Initial seed data

	InitialSeed	id
1	8775	1
2	9357	2
3	2655	3
4	230	4
5	4246	5

GET NEXT SEED

Next, we created %MACRO SEED that pulled the next available seed from the seed dataset. This %MACRO takes the macro variable &Initial and adds 1 each time the %MACRO SEED is called. At the start of each simulation, we defined which initial seed value to use. (Code 3.2) This made the simulation reproducible.

Code 3.2: Get next seed

```
%LET Initial=35;

%MACRO SEED ();

 %let Initial=%EVAL(&Initial. +1);

 data work.seed ;
  set sim.initialseed (Firstobs=&Initial. obs=&Initial.);;
 run;

 data _null_;
  set work.seed;
  call symputx ('InitialSeed',InitialSeed);
 run;

%put Seed is # &Initial. and is valued &InitialSeed. ;


%MEND SEED;
```

CREATE A POPULATION

After the seed data set was created and %MACRO SEED could provide us with seeds, we created our population values. This population dataset was based on parameters that were defined by the proposed study designs and preliminary data. These parameters were 1) the total number of simulations 2) the total number of people in each simulation population 3) the mean value of the immunological parameter 4) the between subject variance expected for the immunological parameter and 5) the within subject variance expected for the immunological parameter. These parameters were all passed into %MACRO TEST.

First, we defined the baseline values (TIME_0) for the immunological parameter of interest based on an expected mean value &MEAN, the between subject variation &BETWEEN_VAR, and a random seed value &INITIALSEED. (Code 3.3). After creating the baseline values, we created the additional time values. These additional values were created with an ARRAY and used the baseline values (TIME_0), total number of required values &TIMECOUNT, within subject variation &WITHIN_VAR, and a random seed value &INITIALSEED. (CODE 3.4, Figure 3.2)

Code 3.3: Create the baseline values

```
%MACRO TEST(mean=,between_var=,within_var=,simulation=,population=,timecount=);

data work.test;
*first loop defines the simulation number*;
do i= 1 to &simulation.;
       *second loop defines the number of people in the study*;
       do j=1 to &population.;
                    TIME_0=&mean.+sqrt(&between_var.)*rannor(&initialseed.);
              Simulation_ID=i;
              Person_ID=J;
              output;
       end;
end;

drop I j;
run;
```

Code 3.4: Create the remaining timepoints

```
data work.test_2;
set work.test;

array time (*) TIME_1-TIME_&timecount.;
  do k=1 to &timecount.;
       Time[k]=TIME_0 + sqrt(&within_var.)*rannor(&initialseed.);
       end;
  output;
drop k;
run;
```

Figure 3.2. Simulated data set for an immulogical parameter

	TIME_0	Simulation_ID	Person_ID	TIME_1	TIME_2	TIME_3	TIME_4	TIME_5
1	-0.25698983	1	1	0.0582786071	-0.842481951	0.2310613216	-0.364313725	-0.908524218
2	-0.317850851	1	2	1.0705308278	0.08337203	0.3882822055	-0.475368462	-1.842836024
3	-0.365121548	1	3	-1.160514835	-0.910291617	0.5614953053	-2.54690461	-1.561557721
4	-0.402542676	1	4	-0.659097204	-0.353302408	-0.048257398	0.6975749099	-0.728148524
5	-0.483387098	1	5	-0.654079172	-1.062566482	-1.160702365	-0.775943037	-0.741556724

ANALYZE AND COMPILE THE RESULTS

Next, we used PROC TRANSPOSE and basic INPUT and SCAN functions to restructure the data.(Code 3.5, Figure 3.3) Our analysis used ODS OUTPUT to capture the covariate parameters from our PROC MIXED models. (Code 3.6). We then created basic flag variables to indicate significant p-values for the Wald z-test for the between subject variance. (Code 3.7) Next, we used ODS OUTPUT with PROC FREQ to calculate the frequencies of each of the significant p-values. (Code 3.8, Figure 3.4). Finally, we used basic code to create a summary data set that included the macro variable values that we used (Figure 3.5). This code includes a variable MIXED with a value of 1. This was done because we wanted to compare the results when we repeated the PROC MIXED using a class statement. Those PROC MIXED values had a variable MIXED with a value of 2. Our final dataset is prepared using PROC APPEND and contains the all the values from both variations of the PROC MIXED models. (Code 2.9, Figure 2.6)

Code 3.5: Restructure the data for analysis

```
proc transpose data=work.test_2 out=work.test_3 prefix=value;
 by simulation_ID person_id;
run;

data work.test_3;
set work.test_3;
 TIME= input(scan(_NAME_,2,'_'),2.);
 TIME2=TIME;
run;
```

Figure 3.3. Restructured data set ready for PROC MIXED

	Simulation_ID	Person_ID	NAME OF FORMER VARIABLE	value1	TIME	TIME2
1	1	1	TIME_0	-0.25698983	0	0
2	1	1	TIME_1	0.0582786071	1	1
3	1	1	TIME_2	-0.842481951	2	2
4	1	1	TIME_3	0.2310613216	3	3
5	1	1	TIME_4	-0.364313725	4	4
6	1	1	TIME_5	-0.908524218	5	5

Code 3.6: ODS OUTPUT to capture covariate parameters from PROC MIXED

```
ODS LISTING CLOSE;
ODS OUTPUT COVPARMS=work.mixed;

proc mixed data=work.test_3 covtest;
  by simulation_ID;
  model value1=time;
  random intercept/ sub=person_id;
run;

ODS OUTPUT CLOSE;
```

Code 3.7: Create basic flags for significant p-values

```
data work.mixed_1;
set work.mixed;
if Covparm="Intercept";
if probz<0.1 then P_point1=1;
if probz<0.05 then P_point05=1;
if probz<0.01 then P_point01=1;
if probz<0.001 then P_point001=1;

if probz=. then do;
       p_point1=.;
       p_point05=.;
       p_point01=.;
       p_point001=.;
  end;
run;
```

Code 3.8: Use ODS OUTPUT and PROC FREQ

```
ODS OUTPUT onewayfreqs=work.mixed_freqs_1;
ODS LISTING;

proc freq data=work.mixed_1 ;
table p_point1 p_point05 p_point01  p_point001/missing;
run;

ODS LISTING CLOSE;
ODS OUTPUT CLOSE;
```

Figure 3.4. ODS OUTPUT from PROC FREQ

	Table	P_point1	P_point1	Frequency	Percent	Cumulative Frequency	Cumulative Percent
1	Table P_point1	.	.	889	88.90	889	88.90
2	Table P_point1	1	1	111	11.10	1000	100.00
3	Table P_point05		.	976	97.60	976	97.60
4	Table P_point05		.	24	2.40	1000	100.00
5	Table P_point01		.	1000	100.00	1000	100.00
6	Table P_point001		.	1000	100.00	1000	100.00

Figure 3.5. Restructured Summary Frequency data

	Table	Percent	mean	between_var	within_var	simulation	population	timecount	mixed	initial
1	P_point001	0.00	-0.274	0.01	0.67	1000	20	12	1	37
2	P_point01	0.00	-0.274	0.01	0.67	1000	20	12	1	37
3	P_point05	2.40	-0.274	0.01	0.67	1000	20	12	1	37
4	P_point1	11.10	-0.274	0.01	0.67	1000	20	12	1	37

Code 3.9: Use PROC APPEND to compile results

```
proc append Base=work._mixed_freqs_all data=work._mixed_freqs_all_&initial. FORCE;
run;
```

Figure 3.6. Summary data from both PROC MIXED models

	Table	Percent	mean	between_var	within_var	simulation	population	timecount	mixed	initial
1	P_point001	0.00	-0.274	0.01	0.67	1000	20	12	1	37
2	P_point01	0.00	-0.274	0.01	0.67	1000	20	12	1	37
3	P_point05	2.40	-0.274	0.01	0.67	1000	20	12	1	37
4	P_point1	11.10	-0.274	0.01	0.67	1000	20	12	1	37
5	P_point001	0.40	-0.274	0.01	0.67	1000	20	12	2	37
6	P_point01	1.20	-0.274	0.01	0.67	1000	20	12	2	37
7	P_point05	3.50	-0.274	0.01	0.67	1000	20	12	2	37
8	P_point1	7.00	-0.274	0.01	0.67	1000	20	12	2	37

REPEAT THE SIMULATIONS FOR DIFFERENT PARAMETERS

Next, we repeated the simulations for different parameters. There are a couple of different ways to do this. The basic way is to call the %MACRO TEST multiple times. A more efficient way is to create an EXCEL control document.[1] (Code 3.10, Figure 3.7)

Code 3.10: Repeat the simulations for different parameters

```
%test(mean=-0.2740, between_var=0.01,within_var=0.67,simulation=1000,population=20,timecount=12);
%test(mean=-0.2740, between_var=0.02,within_var=0.66,simulation=1000,population=20,timecount=12);
%test(mean=-0.2740, between_var=0.03,within_var=0.65,simulation=1000,population=20,timecount=12);
%test(mean=-0.2740, between_var=0.04,within_var=0.64,simulation=1000,population=20,timecount=12);
%test(mean=-0.2740, between_var=0.05,within_var=0.63,simulation=1000,population=20,timecount=12);
%test(mean=-0.2740, between_var=0.06,within_var=0.62,simulation=1000,population=20,timecount=12);
%test(mean=-0.2740, between_var=0.50,within_var=0.16,simulation=1000,population=20,timecount=12);
```

Figure 3.7: Repeat the simulations for different parameters

mean	between_var	within_var	simulation	population	timecount
-0.274	0.01	0.67	1000	20	12
-0.274	0.02	0.66	1000	20	12
-0.274	0.03	0.65	1000	20	12
-0.274	0.04	0.64	1000	20	12
-0.274	0.05	0.63	1000	20	12
-0.274	0.06	0.62	1000	20	12
-0.274	0.5	0.16	1000	20	12

PLOTS

Finally, we plotted the simulation data. We prepared basic if-then code to create a new variable for intraclass correlation (ICC). Then, we used a basic PROC GPLOT to plot the power curve. (Code 3.11, 3.12, Figure 3.8).

Code 3.11: Format plot data and plot

```
data work.plots ;
set work._mixed_freqs_all;
 if between_var=    0.01   then ICC =    0.01   ;
 if between_var=    0.06   then ICC =    0.09   ;
 if between_var=    0.11   then ICC =    0.16   ;
 …….
label percent="Power" population="Number of Subjects" timecount="Number of Time
                                                                 Points:";
run;
```

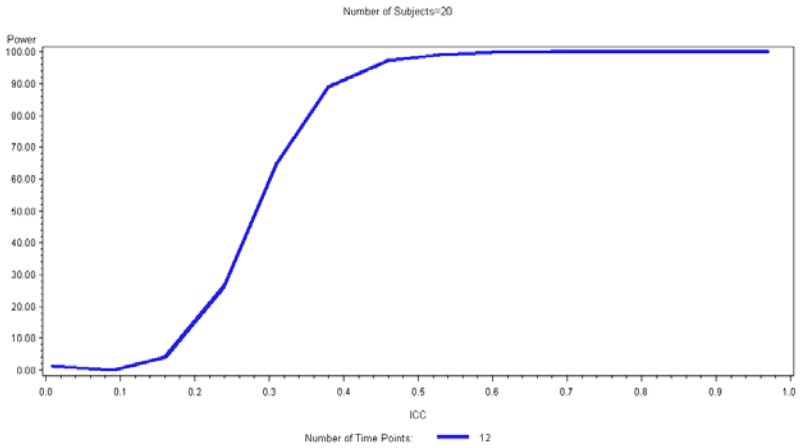
Code 3.12: Plot

```
proc sort data=work.plots;
 by population;
run;

symbol i=j width=5;
ODS LISTING;

proc gplot data=work.plots;
 by population ;
 where table ="P_point01" and mixed=2;
 plot percent*ICC=timecount;
run;
quit;
```

Figure 3.8: Power Plot by population



CONCLUSION

We recently faced a huge challenge: help with study design, determine the appropriate statistics for each aim, analyze preliminary data for 400+ parameters, create publication ready figures, and calculate power for each of 5 grants. We had one week to complete everything.

Without the following SAS tools, we could not have finished our grant preparation: 1) %MACRO programming to automate analysis and plots for multiple parameters, 2) ODS OUTPUT to create summary datasets and summary documents, 3) PROC GPLOT and PROC SGPLOT/SGPANEL to produce presentation-ready graphs to include in the grant, and 4) EXCEL control documents to control the variable names, labels, and lists. With these tools we were able to successfully meet our short timeline and be flexible enough to meet the changing requirements as the projects evolved.

This paper is the third in the 'Grad Student How-To' series[1,2] and gives graduate students tools for grant preparation.

REFERENCES

1) Priest EL. Keep it Organized- SAS tips for a research project. South Central SAS Users Group 2006 Conference, Irving, TX, October 15-18, 2006.

2) Priest EL, Adams B, Fischbach L. Easier Exploratory Analysis for Epidemiology: A Grad Student How To Paper. SAS Global Forum 2009.

3) Fehd, RJ. List processing basics: Creating and using lists of macro variables. SAS Global Forum 2007.

4) Haworth, L. Output Delivery System: The basics. Chapter 7: Output Data Sets. SAS Institute, Cary NC, 2001.

5) Carpenter, A. Carpenter's complete guide to the SAS macro language. SAS Institute. Cary, NC, 2004.

ACKNOWLEDGMENTS

Thank you to Dr. Jacques Banchereau, Dr. Damien Chaussabel, and The Baylor Institute for Immunology Research for their grant and data analysis projects which led to the development of the code in this paper.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Elisa L. Priest, MPH
elisapriest@hotmail.com