Paper CC14-2011

A Graphics Tool for the Evaluation of Longitudinal Outcomes in Clinical Care

Matthew C. Fenchel, M.S., Cincinnati Children's Hospital Medical Center, Cincinnati, OH
Laurie M. Kahill, Cincinnati Children's Hospital Medical Center, Cincinnati, OH
Katharine D. Sebastian, A.A.S., Cincinnati Children's Hospital Medical Center, Cincinnati, OH
Raouf S. Amin, M.D., Cincinnati Children's Hospital Medical Center, Cincinnati, OH
Rhonda D. VanDyke, Ph.D., Cincinnati Children's Hospital Medical Center, Cincinnati, OH

Abstract

In health-care, physicians and patients dealing with a chronic condition may be greatly helped by a graphics tool that tracks progress. Cystic fibrosis (CF) patients suffer from an overall declining lung function (FEV_1%). Understanding the progression of FEV_1% over time is of critical importance, in order to prescribe and participate in interventions. We were asked by physicians at Cincinnati Children's Hospital to develop a patient-specific, graphical tool, to give a "picture" of a lung-function over time. The graph had to meet specific -- and programmatically challenging -- requirements. The SGPLOT procedure is primarily used, along with extensive macro and macro-variable programming. While the original purpose of the program was to meet a specific need, we have attempted to write the code presented in such a way that it is "generalizable" -- i.e. that it can be used for a variety of longitudinal analyses.

Introduction

For analytical professions the phrase "a picture is worth a thousand words" could accurately be altered to "a graph is worth a thousand numbers." Graphs can be especially beneficial to the non-statistical end-users of data analyses, who may not always understand "the statistics." In the health-care field, physicians and patients dealing with outcomes from a chronic condition may be greatly helped by a graphics tool that tracks progress in an understandable fashion.

The graphics tool described in this paper is demonstrated for our motivating example. Cystic fibrosis (CF) patients suffer from an overall declining lung function, which is often measured by FEV_1 percent predicted (FEV_1%). Most CF patients are seen on a quarterly basis (sometimes more) where FEV_1% is measured. Understanding the progression of FEV_1% is of critical importance to medical personnel and patients alike, in order to prescribe and participate in needed interventions. While the overall history of lung function is important, the most recent (approximately two years) trend in a patient's FEV_1% measurements provide a "real-time" look at his current condition and possible need for additional treatment.

We were asked by physicians at Cincinnati Children's Hospital Cystic Fibrosis Center to develop a patient-specific, graphical tool that could be used in a clinical setting to accurately and efficiently give a "picture" of a lung-function over time. The graph had to meet the following requirements:

a) show all FEV_1% values -- both from measurements taken during regular visits and those take during hospitalizations due to pulmonary exacerbations, with distinct colors for each type of visit;

b) show overlapping two-year slope lines (but calculated only from non-exacerbation data) over time that were easy for patients and M.D.'s to understand (quadratic and cubic regressions were ruled out);

c) show extrapolated lines over the next 18 - 24 months, to demonstrate where lung-function might be if 1) the current slope were maintained and 2) if the current slope were improved by 5;

d) show value of each slope line in the legend;

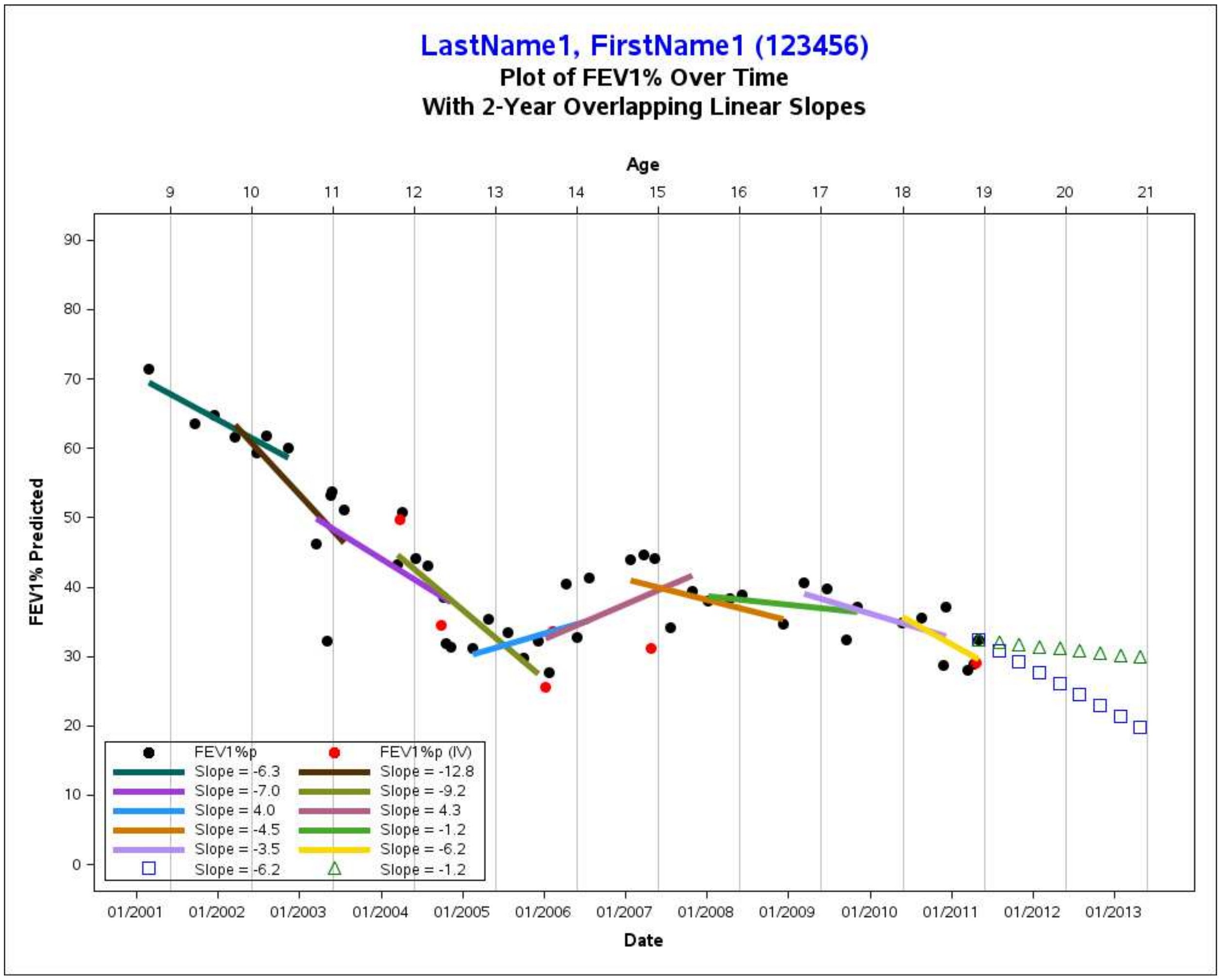
e) use one x-axis for date of visit (X1AXIS) and the second x-axis for age (X2AXIS);

f) make graph "interactive" -- i.e. when a cursor is placed over a data point or line, the value of that FEV_1% measurement is displayed along with a patient's date-of-visit or age;

This paper describes the program that was developed, which could easily be adapted to other, similar needs for individual graphs with longitudinal characteristics. This program uses the MIXED and SGPLOT procedures, along with macros and macro variables, and can be used by someone with intermediate statistical and SAS knowledge. The entire program is contained in the Appendix and various sections are discussed, in order, throughout the paper.

Sample Graph

Since the premise of this paper is that “picture is worth a thousand words”, we thought we would show a sample graph at the beginning, so the reader would know what the “end product” looks like. The black dots represent actual FEV_1% values from clinical visits. The red dots represent FEV_1% values from hospital visits due to exacerbations. Solid lines represent overlapping, two-year, linear regression slopes based on the clinical (non-exacerbation) data only. The blue open squares represent extrapolated values based on the patient’s last two-year slope. The green open triangles represent extrapolated values based on an improvement of 5 in the patient’s last two-year slope.



User Input

After a thorough introductory section, the user is prompted for some basic information. The first subsection involves locations for the data file, the PNG files and the HTML files. Because of the interactive nature of the graphs -- i.e. data values appear when the cursor is placed over a dot or line on the graph (when the graph is viewed in a HTML browser) two sets of output files are required and are generated by SAS. (This program assumes that the user may want the PNG and HTML files in two separate locations -- but this is not required for this feature to work correctly.)

The second subsection asks for variable names and labels for key variables like subject, date of birth, date (of visit, treatment, etc.), the outcome variable (that will appear along the y-axis), first name and last name of the subject.

The third subsection asks specifics about the Group variable. In the graph above, data from the two different levels of the Group variable are displayed as the black or red dots.

Finally, the user can specify the second and third lines of the title. The first line of the title (containing the last name, first name and ID of the patient) is generated automatically.

After this input, there is nothing more the user needs to do. A small section of the actual data is shown below, to demonstrate the formatting and organization of the data. In brief, the data is "stacked". Values for any variables that could be used to identify the patient have been changed.

MRN	LName	FName	DOB	pftdate	fev1pct	status
123456	LastName1	FirstName1	15-Jun-95	21-Jul-03	51.2	Normal
123456	LastName1	FirstName1	15-Jun-95	17-Mar-04	43.3	Normal
123456	LastName1	FirstName1	15-Jun-95	30-Mar-04	49.8	Hosp
123456	LastName1	FirstName1	15-Jun-95	07-Apr-04	50.8	Normal
123456	LastName1	FirstName1	15-Jun-95	07-Jun-04	44.2	Normal
123456	LastName1	FirstName1	15-Jun-95	02-Aug-04	43	Hosp
123456	LastName1	FirstName1	15-Jun-95	29-Sep-04	34.5	Normal

Section A: Calculate Precise Age; Create New Variables Based on "Group"

One of the biggest challenges for this plot was calculating an age at each visit that matched *exactly* to the date. This required obtaining age that was precise to several decimal points. One of the more common methods -- (Date - DOB) / 365.25 -- gives a very close approximate value, and is certainly good enough for most statistical analyses. However, this approximation created a "shift" when attempting to plot actual data points by date, and then plotting slope lines by age. In other words, the slope lines did not line up precisely with the scatter plots. (Other versions for calculating age that the authors found created the same difficulty.)

To solve this, additional code was developed that calculated a precise age that matched with the date of visit. In short, the code solves the following issues -- mostly using the DAY, MONTH and YEAR functions, but also using SAS date values:

1) calculate the person's age -- integer value only (i.e. 7, 8, 9, etc.) -- using the DAY, MONTH and YEAR functions;
2) determine how many days are between the person's previous birthday and his next birthday (either 365 or 366);
3) determine how many days are between the visit date and the previous birthday.

In the program then, Age3 = integer age + (# days since last birthday / # days between birthdays).

Also in this section, levels from the original Group variable are separated into two additional variables, for ease of programming in PROC SGPLOT.

Section B: Assign Subject ID, Last Names and First Names to Separate Macro Variables

In this section, using the CALL SYMPUT function, each subject's ID, last name and first name are assigned to their own individual macro variables. This is mainly used for the first title of the graph. In addition, the count of the subjects is also assigned to a macro variable, for use in running the main macro, which begins in the next section.

Section C: Determine Key Values by Subject and Assign to Separate Macro Variables

The main macro -- which loops through each subject individually -- begins now. In Section C, the program obtains some key information from each subject, and assigns those values to macro variables. This information includes:

- year of first visit and year of last visit;
- date of last visit;
- last, observed value for the y-variable (in our example, $FEV_1\%$), which obviously takes place on date of last visit;
- minimum and maximum values for the outcome variable (y-variable);
- each year when visits occurred;
- the number of distinct years where the patient was seen.

Some of these values will be used later to construct the dimensions of the plot -- i.e. where each axis should begin and end. The other values will be used to calculate regression slopes for specific time periods (i.e. every two years), or to be used as the location for starting the extrapolated "lines".

Section D: Calculation of Two-Year Slopes and Extrapolated Slopes

In this section, PROC MIXED is used to obtain two-year, overlapping slopes for each subject (again, only based on the clinical, non-exacerbation data). These slope values are assigned to macro variables to be used later in the graph legend. In addition, slopes for the two extrapolated slopes -- which are based on the slope from the final two years -- are saved in their own macro variables for later use.

Section E: New Data Variables for Plotting Regression Lines

The main purpose of the code in this section is to create new variables -- with values of the y-variable -- that are in overlapping, two-year increments. For example, for y-data in the years 2000 and 2001, a new variable "y_2000" is created with values only from those two years. The next variable, "y_2001" is created for y-data for the years 2001 and 2002 only. This continues until data from the final year are assigned.

In addition, two new variables are created that contain values for the two extrapolated "lines". (By way of reminder, one extrapolated line is based on the final two-year slope, and the other extrapolated line is based on an improvement of 5 from the final two-year slope.)

All of the new variables described above will be used in plotting regressions lines against age (specifically, the "Age3" variable). While there may be more sophisticated methods for plotting these overlapping, two-year lines, this seemed like best method, considering the capabilities of PROC SGPLOT.

Section F: Set Macro Variables so that Date Axis and Age Axis Match Precisely

The second challenge with dates and ages was making sure that the start and end of each x-axis on the plots (X1AXIS is on the bottom, X2AXIS is on the top) also matched precisely -- especially when a certain amount of offset or "buffer" is desired between the first and last values of each axis and the border of the plot. This section assigns matching values of date and age to macro variables for use in setting-up the X1AXIS and X2AXIS in PROC SGPLOT.

The TEMPLATE procedure is also briefly used to assign a color to the gridlines that will be used for age This new style is based on the sansPrinter style and is invoked in the ODS statements.

Section G: Running PROC SGPLOT and ODS Statements

After all of the preparation in the previous lines of code, we finally are able to plot the graph! The first several lines involve ODS output. In specific, SAS is told where to put the HTML and PNG files, what name to use for each subject's file and which style to use. In the ODS GRAPHICS statement, an important option is invoked -- the IMAGEMAP option. This must be used in order for the scatter points and regression lines to be "active" -- i.e. that data values display when the cursor is held over a point or a line.

The first two SCATTER statements plot the actual data values from the clinical and hospital data. The user will notice that the two new Group variables are being used (described in Section A above). While it is possible to have only one SCATTER statement to plot both sets of values (using the GROUP option), PROC SGPLOT does not have a way to specify distinct colors for those distinct sets of data. Using two SCATTER statements allow the user to more easily specify (within PROC SGPLOT) attributes of the scatter symbols, without using PROC TEMPLATE. Each SCATTER statement still uses a GROUP option, but the Group2 and Group3 variables only have levels for either clinical *or* hospital data. The NOMISSINGGROUP sub-option ensures that data is not plotted when there is a missing entry in either Group2 or Group3.

The next subsection invokes a %DO - %TO loop to plot the overlapping regression lines, based on the macro variables described in previous sections. Labels for each line are also assigned, based on previously created (and described) macro variables. (In theory, this part of the code would not have needed such a %DO - %TO loop: the GROUP option could have been used instead. However, the LEGENDLABEL option has no effect when the GROUP option is used, and we wanted specific labels for each regression line.)

Please note that the first two SCATTER statements use the date variable for the x-axis, and that the REG statement uses the age variable ("Age3"). This allows the end-users to view a visit date for a specific data point, and then view the age along the regression line. This is the reason why a precise version of age was needed.

The next two scatter statements plot the two sets of extrapolated values, as described earlier. Since obviously these are projected "future" values, no actual visit dates exist for these data. Thus, they are also plotted against age -- which, again, benefits the end-users of the graph, who can see what lung function may be at a future age (if the extrapolated values were to hold steady).

The next set of statements for the axes are actually somewhat more complicated than what one might first assume. The YAXIS statement is the simplest, simply setting the values based on the maximum and minimum values of the y-variable -- which were saved previously in macro variables. To make the graph more readable, the OFFSETMIN and OFFSETMAX statements were used to create a small amount of "buffer" at each end of the axis.

In the XAXIS statement, there are a number of options used. A key point to keep in mind is the difference between the VALUES option and the MIN and MAX option. In general, the VALUES option specifies the values (or range of values) that are used as *labels* for the tick marks. If the MIN and MAX options are not used, then these range of values also are used as the minimum and maximum limits for that axis. The MIN and MAX options can be used to specify the limits of the axis -- limits that may differ from the VALUES statement. (If they do differ, then the VALUESHINT option also needs to be used, which it is in this case.)

The values for the MIN and MAX options in the XAXIS statement are dates that match up exactly with the precise ages that are used in the MIN and MAX options in the X2AXIS statement. Again, this is critical, so that the regression lines are in alignment with the scatter plots.

Back to the XAXIS statement. For ease of use, it was desired to have tick-marks at the beginning of each calendar year and in increments of one year along the lower x-axis. Thus, in the VALUES option, the range begins with the value of the first calendar year in which a visit took place -- not with the first visit date itself. In that way, the values appearing along the lower x-axis will be in increments of, for example, “01/2000”, “01/2001”, etc. Otherwise, had we started the range with the first visit date, the year increments may have started with “05/2000” or “08/2000” -- and progressed accordingly.

Continuing with this same VALUES option: the reader will notice that -- because we are using stored SAS date values in the macro variables that define the range -- PROC SGPLOT assumes that tick-marks should appear for each *day*. Therefore, we use the BY sub-option and set it = 366 (maximum days in one year). We also use the INTERVAL option and set it = YEAR. (Note: the “BY=366” is definitely necessary for the correct labeling, but the INTERVAL statement may be redundant, depending on the nature of the date data.)

In the X2AXIS statement, the reader will notice that different values are used in the range of the VALUES option than in the MIN and MAX options. The range for the VALUES option is smaller. The reason for this is because we did not want labels at the very edges of the axes.

The final statement -- the KEYLEGEND statement -- allows us to specify the location of the legend, as well as the number of columns in which the entries will be displayed (two in our example).

Conclusion

This program was designed to produce graphs that are used on a regular basis in a clinical setting. While giving an immediate “overall” picture of a longitudinal outcome, physicians and patients can also use it to obtain more specific information at any point along the graph.

Through the use of user-friendly, multiple inputs at the start of the program, a new user could easily use this code to produce graphs for any number of longitudinal outcomes. Some of these may include BMI, blood pressure, results from fitness tests, or sales. Of course, the user may need to alter the code (in addition to the inputs at the beginning) to meet a specific need. It is hoped that this paper provides enough of an overview, so that a programmer will know what to change in order to meet the needs of his analysis.

Contact Information

Comments and questions are welcome. The author may be contacted at:

Matthew Fenchel
Division of Biostatistics and Epidemiology
Cincinnati Children’s Hospital Medical Center
MLC 5041, 3333 Burnet Avenue
Cincinnati, OH 45229-3039

E-mail: Matthew.Fenchel@cchmc.org

Appendix: Complete Program

```
/***************************************************************************************************

PROJECT:       A GRAPHICS TOOL FOR THE EVALUATION OF LONGITUDINAL OUTCOMES IN CLINICAL CARE
REVISION DATE: 08-10-2011
STATISTICIAN:  Matthew Fenchel, M.S.

INTRODUCTION: "A graph is worth a thousand numbers" is probably most applicable when one is
attempting to get the "big picture" about a longitudinal outcome.

PURPOSE: The purpose of this program is to have a user-friendly method of producing scatter-plots
that offer the following characteristics:

        1) plot all values, with distinctive colors for different groups (if applicable);
        2) show overlapping, two-year slope lines;
        3) show extrapolated lines over the next 18 - 24 months, to demonstrate what the
           predicted values would be in the outcome variable if a) the current slope were
           maintained, and b) if the current slope were improved by 5;
        4) show the value of each slope-line in the legend;
        5) use two different variables for the x axes;
        6) make the graph "interactive" -- i.e. when the cursor is place over a data point or
           line, the value of the outcome variable is displayed, along with the value of the
           predictor variable.

CONCEPT: We were asked by physicians at Cincinnati Children’s Hospital Cystic Fibrosis Center to
develop a patient-specific, graphical tool that could be used in a clinical setting to accurately
and efficiently give a “picture” of lung-function over time. While the long-term history of lung
function is important to know, the most recent (approximately two years) trend in a patient’s
FEV1% measurements provide a “real-time” look at his current condition.

DESIGN:  While originally designed for a very specific application (as described above), this
program is an attempt to make the original code "generalizable" and relatively easy to apply to
other longitudinal applications.

USER INPUT:  There are a number of details (next section) that the user will need to input.

OUTPUT:  In order to produce "interactive" graphs with SAS, two graph files -- one PNG file and
one HTML file -- are created for each unique ID.  The original purpose of this program was to use
the HTML graphs in weekly chart conferences with medical personnel, in order to assess each
person's progress. In order to take advantage of the interactive capabilities of a plot, the HTML
file must be used.

NOTES:  1) PROC IMPORT is used to obtain the data. Obviously this can be changed within the
           program if the user desires to use another way of obtaining the data from a SAS or
           non-SAS data set.

        2) This program maintains some specific capabilities (variables for first and last
           names, group variable, etc.) that another user may not need nor want.  In that case,
           changes will need to be made in the main code.

***************************************************************************************************/

        options nonumber nodate formdlim = "*" minoperator;

/***************************************************************************************************
                              STEPS TO BE COMPLETED BY USER
***************************************************************************************************/

/*  Input file location information.  */

%let datapath = R:\feni7q\CF Chart Tool\MWSUG 2011\MWSUG.xls;  /* Path and name of data set. */

%let pngpath = R:\feni7q\CF Chart Tool\MWSUG 2011\PNG Files;   /* Path for PNG files. */

%let htmlpath = R:\feni7q\CF Chart Tool\MWSUG 2011\HTML Files;      /* Path for HTML files. */


/*  Input general variable information.  */

%let ID = MRN; /* Name of subject variable.  A separate plot will be produced for each subject.*/

%let DOB = dob; /* Name of birth-date variable. Used for an exact age calculation. */
```

```
%let Date = pftdate; /* Name of date variable. This will be used for the x-axis in the plots. */
%let lDate = Date;   /* Label for the date variable.  This will appear along the x-axis. */

%let lage = Age;     /* Label for the x2-axis.  This is what will appear along the top x-axis. */

%let yvar = fev1pct;  /* Name of variable that will go along the y-axis in the plots. */
%let lyvar = FEV1% Predicted; /* Label for the y-axis variable. This will be along the y-axis. */

%let FirstN = FName;  /* Name of variable containing first name. */
%let LastN = LName;   /* Name of variable containing last name. */


/*  Input Group variable information.  Only two levels of Group are allowed.  Separated colors
for each Level will appear in the scatter-plot.  Regression lines will only be calculated for one
level of Group.  */

%let Group = pulmexac;               /* Name of group variable. */

%let GLevel1 = regular; /* Data from this level will be used to calculate regression lines. */
%let GLevel2 = Yes;     /* Data from this level will NOT be used to calculate regressions. */

%let lGL1 = FEV1%p;      /* Label in plot for first level of Group. */
%let lGL2 = FEV1%p (IV); /* Label in plot for second level of Group. */


/*  Input information for 2nd and 3rd titles. 1st title will have Last Name, First Name (ID).  */

%let title_2 = Plot of FEV1% Over Time;                         /* Title 2 for the plots. */
%let title_3 = With 2-Year Overlapping Linear Slopes;           /* Title 3 for the plots. */


/***************************************************************************************
                      NOTHING MORE FOR USER TO DO.  PROGRAM STARTS FROM HERE.
***************************************************************************************/

/*  A. Calculate precise age.  Create new variables based on levels of Group variable.  **/

proc import datafile = "&datapath" out=cf replace; run;

proc sort data=cf; by &id &Date; run;

data agecode; set cf;

        %macro age;

        format &Date mmddyy8.;
        dayb = day(&dob);            mthb = month(&dob);                  yrb = year(&dob);
        day2 = day(&Date);           mth2 = month(&Date);  yr2 = year(&Date);    yr1 = yr2 - 1;
        yr3 = yr2 + 1;

        if (mth2 < mthb) or (mth2 = mthb and day2 < dayb) then BD = "NDone";
        if (mth2 > mthb) or (mth2 = mthb and day2 ge dayb) then BD = "Done";

        if BD = "NDone" then do;
                intage = yr2 - yrb - 1;
                dayyr = mdy(mthb, dayb, yr2) - mdy(mthb, dayb, yr1);
                dayel = mdy(mth2, day2, yr2) - mdy(mthb, dayb, yr1);
                dage = dayel / dayyr;
                Age3 = intage + dage;
                end;

        if BD = "Done" then do;
                intage = yr2 - yrb;
                dayyr = mdy(mthb, dayb, yr3) - mdy(mthb, dayb, yr2);
                dayel = mdy(mth2, day2, yr2) - mdy(mthb, dayb, yr2);
                dage = dayel / dayyr;
                Age3 = intage + dage;
                end;

                Age = (&Date - &dob) / 365.25;
                drop dayb -- dage;
                %mend;
                %age;

                keep &id &Date Age3 Age; run;
```

```
data cf2;
        merge cf agecode;
        by &id &Date;
        YR = Year(&Date);      format &Date mmddyy8.;

        informat Group2 $12.; format Group2 $12.;
        Group2 = &Group;
        if &Group = "&glevel2" then Group2 = "&lgl2";
        if &Group = "&glevel1" then Group2 = "";
        if &Group = "&glevel1" then Group3 = "&lgl1";
        if &Group = "&glevel2" then Group3 = "";

        Age2 = Age3;
        if &Group = "&glevel2" then Age2 = .;
        run;


/**** B.  Assign specific ID, FirstN and LastN to their own macro variables for later use.  ****/

proc sort data=cf2; by &id yr &Date; run;

proc sort data=cf2 nodupkey out=short; by &id; run;

data short2; set short;
        by &id;
        Num = left(put(_N_, 2.));
        ID2 = left(put(&id, 7.));
        Title = trim(left(&LastN))||", "||trim(left(&FirstN))||" ("||trim(left(ID2))||")";
        call symput("ID"||Num, &id);
        call symput("FName"||Num, trim(left(&FirstN)));
        call symput("LName"||Num, trim(left(&LastN)));
        call symput("Title"||Num, trim(left(Title)));
        if Last.&id = 1 then do; call symput("IDCount",_N_); end;
        run;

/*  C.  Determine minimum and maximum values for the y-variable and date, in order to set ranges
for the y and x axes. From this point, the code will run for each subject (ID) individually. */

%macro track;

        %do i = 1 %to &IDCount;
        title1; title2; title3;

%macro plots/mindelimiter=',';

        data cf3; set cf2;
                where &id = &&ID&i;
                by &id &Date;

                if First.&id = 1 then do;
                        year = year(&Date);
                        YearStart = mdy(1, 1, year);
                        call symput ("YearStart", YearStart);
                        end;

                if Last.&Date = 1 then do;
                        call symput("Last_y", &yvar);
                        call symput("LastDate", &Date);
                        end;
                run;

        proc means data=cf3 noprint; var &yvar; output out=fev max = max min = min; run;

        data fev2; set fev;
                max2 = round(max + 15, 10.);
                min2 = round(min - 20, 10.);
                call symput("Max_y", max2);
                call symput ("Min_y", min2);
                run;

        proc sort data=cf3 nodupkey out=short3; by YR; run;

        data short4; set short3;
                by YR;
```

```
            Num = left(put(_N_, 2.));
            call symput("Year"||Num, YR);
            if Last.YR = 1 then do; call symput("YRCount", _N_); end;
            if Last.YR = 1 then do; call symput("LastYear", YR); end;
            run;

        data short4; set short4;
            call symput("FirstYear", &Year1);
            run;

            %if &FirstYear = %eval(&LastYear) %then %let LastYear = %eval(&FirstYear + 1);

/***  D.  Get two-year overlapping slopes for each subject individually.  Assign slope values to
macro variables. Values for extrapolated slopes will also be calculated.   */

        %do k = &FirstYear %to (&LastYear - 1);
            proc mixed data=cf3;
                where YR in (&k, %eval(&k+1)) and &Group ne "&GLevel2";
                model &yvar = Age3 / s;
                ods output solutionf = fe;
                run;

            data fe2; set fe;
                where Effect = "Age3";

                call symput("s&k", round(Estimate, 0.01));

                Est = left(put(Estimate, 5.1));
                Slope2 = "Slope = "||Est;
                call symput("Slope&k",Slope2);

                    %if &k = (&LastYear - 1) %then %do;
                        call symput("Extrap", round(Estimate, 0.0001));
                        call symput("Extrap5", round((Estimate+5), 0.0001));

                        %let SlopeEx = &&Slope&k;
                        Est5 = left(put((Estimate+5), 5.1));
                        Slope3 = "Slope = "||Est5;
                        call symput("SlopeEx5", Slope3);
                        %end;
                run;
            %end;

/**  E.  Create new variables for plotting all regression lines (including extrapolations).  **/

        data cf4; set cf3;
            %do k = &FirstYear %to (&LastYear - 1);
                if YR in (&k, %eval(&k+1)) and &Group ne "glevel2" then do;
                y_&k = &yvar;  label y_&K = "&lGL1";
                end;
            %end;
            run;

        data cf5; set cf4 end=lastone;
            output;
            if lastone then do;
            do i = 0 to 8;
            _y_Ex = &Last_y + (i*0.25*&Extrap);           label _y_Ex = "&lGL1 Extrap";
            _y_Ex5 = &Last_y + (i*0.25*&Extrap5);         label _y_Ex5 = "&lGL1 Extrap5";
            &Date = &LastDate + (i*91);
                %age;
                &yvar = .;
                %do j = %eval(&LastYear-1) %to %eval(&LastYear-1); y_&j = .; %end;
                output;
                end; end; run;

        data cf5; set cf5;
            if i = 8 then do;
                Max_y = &Max_y;        Min_y = &Min_y;
            if _y_Ex5 > (Max_y-10) then do;
                max2 = round(_y_Ex5 + 15, 10.); call symput("Max_y", max2); end;
            if _y_Ex < (Min_y+10) then do;
                min2 = round(_y_Ex - 15, 10.); call symput("Min_y", min2); end;
                end; run;
```

```
/* F.  Set macro variables so that the Date axis (X1Axis) and Age axis (X2Axis) match exactly. */

       proc means data=cf5 noprint;
              var &Date age3;
              output out=maxnum max(&Date age3)=maxdate maxage min(&Date age3)=mindate minage;
                     run;

       data maxnum; set maxnum;
              call symput("MaxDate", maxdate+183);         call symput("MinDate", mindate-183);
              call symput("MaxDate2", maxdate);            call symput("MinDate2", mindate);
              call symput("MaxAge", (maxage+0.5));         call symput("MinAge", (minage-0.5));
              call symput("MaxAge2", round(maxage,1)); call symput("MinAge2", round(minage,1));
                     run;

       proc template;
              define style styles.sansprinter2;
              parent=styles.sansprinter;
              class graphgridlines  / contrastcolor=ltgray;
              end;
              run;

/**  G.  Code for actual, final plot for each subject individually.  ***/

       options orientation = landscape;

       ods listing close;
       filename pngout "&pngpath";
       filename htmlout "&htmlpath";
       ods html path=htmlout body = "&&LName&i.., &&FName&i...html"
                gpath = pngout (url = "&pngpath.\")
                style = sansprinter2;

       ods graphics / height = 8in width = 10in imagemap=on;
       title1 height = 1.7 color = blue "&&Title&i";
       title2 height = 1.4 "&title_2";
       title3 height = 1.4"&title_3";
       title4 " ";
       proc sgplot data=cf5;

       scatter y=&yvar x=&Date / markerattrs = (color=black symbol=circlefilled size=8)
                                 group=Group3 nomissinggroup;
       scatter y=&yvar x=&Date / markerattrs = (color=red symbol=circlefilled size=8)
                                 group=Group2 nomissinggroup;

       %do k = &FirstYear %to (&LastYear - 1);
              reg y=y_&k x=age3 / nomarkers lineattrs = (pattern=1 thickness = 4) legendlabel =
                                 "&&Slope&k" x2axis;   %end;

       scatter y = _y_Ex x = age3 / markerattrs = (color = blue symbol=square size=10)
                                 legendlabel = "&SlopeEx" x2axis;
       scatter y = _y_Ex5 x = age3 / markerattrs = (color = green symbol=triangle size=10)
                                 legendlabel = "&SlopeEx5" x2axis;

       label &yvar = "&lyvar" &Date = "&ldate" Age3 = "&lAge";
       yaxis values = (&Min_y to &Max_y by 10) offsetmin = 0.04 offsetmax = 0.04;
       xaxis values = (&YearStart to &maxdate2 by 366) interval = year tickvalueformat = mmyys7.
              min = &mindate max = &maxdate valueshint;
       x2axis grid values = (&minAge2 to &maxAge2) min = &minage max = &maxage valueshint;
       keylegend / location = inside position = bottomleft across = 2;
       run;

       ods html close;
       ods listing;

       %mend plots;   %plots;
       %end;
       %mend track; %track;

       /*  END OF PROGRAM  */
```