Paper RF - 077 - 2019

Utilizing Macros to Create Patient Site Matching via ZIP-Code Radiuses

Kathryn Schurr M.S. - Quest Diagnostics

Abstract

Determining how to match patients to clinical trial sites is a messy problem. This problem includes many different facets not limited to: inclusion criteria, exclusion criteria, and proximity to clinical trial sites. By creating a site matching program in SAS®, users are able to take patient lists and can easily determine whether or not a patient falls within a certain proximity of the site(s) of interest. This macro takes into account the ZIP Code® location of the proposed site and the ZIP Code location of the patient. It then is able to compute whether or not that patient should be assigned to any particular site. This paper builds upon an already existing macro that calculates the ZIP Codes within a certain mile radius of a singular site location.

Introduction

Quest Diagnostics - Information Ventures recently launched a program which is changing the face of patient recruitment for Clinical Trials. By leveraging the 48 billion plus laboratory records that Quest Diagnostics obtains through normal business operations, Quest Diagnostics can help identify patients matching clinical trial inclusion and exclusion criteria based on previous lab results. Once patients have been identified who match the criteria of interest, the problem arises on how to match those patients to a potential clinical trial site. By utilizing and enhancing an existing macro which identifies ZIP Codes within a certain mile radius of one point of interest, Quest Diagnostics can fully realize the geographic potential of patient recruitment for a particular scenario.

The Original Macro

The solution to identifying patients within a ZIP Code radius began with determining which ZIP Codes lay within a certain distance of the center point of interest. The macro called "ZipsNear" taken from SASCommunity.org by user RichardK is given below. In this macro, the user inputs a base ZIP Code (base), a distance radius in miles (dist), and specifies where to put the resulting list of ZIP Codes that fit that parameter in a subsequent library (libout) and dataset (dsnout). Using mathematical operations to draw a circle around the point of interest, any ZIP Code that is partially within that circle will be included.

```
%macro ZipsNear(base=,dist=,libout=work,dsnout=)
      /des='Returns zipcodes within x miles of a given zipcode.';
  %if &base eq %str() %then
  %do;
    %put %nrstr(%%)ZipsNear parameters:;
    %put %str( %()base  = Base Zipcode,;
    %put %str(  )libout = Output library,;
    %put %str(  )dsnout = Output dataset%str(%));
    %goto Quit;
  %end;
  proc sql noprint;
    %* fetch latitude and longitude of base zip code *;
    select y, x into :lat, :lon
    from   sashelp.zipcode
    where  zip=&base ;
    %* fetch zips within specified distance from base zip code *;
    create table &libout..&dsnout. as
    select
      zip,
      put(zip,z5.) as ZipText,
      put(&base,z5.) as BaseZip,
      3956 * (2 * arsin(min(1, sqrt((sin(((y - &lat) * constant('pi')/180)/2)**2) + ((cos(&lat
* constant('pi')/180)) * (cos(y * constant('pi')/180)) * (sin(((x- &lon) *
constant('pi')/180)/2)**2)))))) as distance
    from
      sashelp.zipcode
    where
      calculated distance <= &dist
    order by zip;
  quit;
  %Quit:
 %mend ZipsNear;
```

Expanding the Original Macro

The original macro as presented above provides a list of ZIP Codes within a certain distance of only one ZIP Code. Running this macro multiple times for the sake of Clinical Trials Patient Recruitment is quite tedious. By expanding the original macro to incorporate a preloaded list of ZIP Codes of interest, the program becomes much more versatile and efficient. Along with expanding the macro to allow for looping through preloaded ZIP Codes of interest, the full program shown in this paper also matches patients to the areas of interest and creates an output file specifying how many patients matched to each site (non-exclusively) and how

many unique patients matched to all sites total. For ease of use, this entire process has been automated via various macros and global macro variables.

User Inputs

To begin, the user must specify 5 different things in the beginning of the program. These items are identified below in the SAS code by being highlighted in yellow. The first item is the location where the patient records data is stored. This becomes the QueryDataSet variable. In the example given below, a library has been previously specified and directs SAS to the DATASET PosPats. This patient dataset must have a Patient ID variable named 'Pat_ID' and a ZIP Code variable that is qualitative, of LENGTH 5, and is named 'ZIP5'.

```
/* INPUT FINAL QUERY DATASET HERE */
%let QueryDataSet          =        D.PosPats;
```

Second, the program requires the user to identify where the output file should be placed (in .csv format) when the program has completed running. This variable is called OUTFILELOCATION.

```
/* INPUT OUTPUT FILE LOCATION HERE */
%let OUTFILELOCATION  =        "/ClinicalTrials/Trial1/SiteMatch.csv";
```

The location of the site list that will be used to identify ZIP code radiuses from is next. The site list should be an EXCEL file in .xlsx format and should have the ZIP Code variable named 'Zip'. In this program the variable is called SITELISTLOCATION.

```
/* INPUT SITE LIST LOCATION HERE - XLSX FORMAT, ZipCode must be
named 'Zip' */
%let SITELISTLOCATION  =        "/ ClinicalTrials/Trial1/Sites.xlsx";
```

The fourth user input that must be specified is the number of unique ZIP Codes that the site list contains. This value should be an integer greater than 0 and less than 1,000. This becomes the variable SITENUMBER.

```
/* INPUT THE NUMBER OF SITES THAT ARE BEING MATCHED TO */
%let SITENUMBER                =        20;
```

Finally, the user must indicate what size radius they would like to utilize when performing the search. This must be a number greater than 0 and this corresponds to the variable DISTANCE.

```
/* INPUT THE DISTANCE OF RADIUS MATCH THAT IS OF INTEREST
HERE */
%let DISTANCE                  =        50;
```

Prepping the Site List

After the user has made the mandatory specifications, no other changes to the code will need to be made. The rest of the code and how it functions will now be explained. The first section of code in the program, after the user inputs, is loading the patient data and site list into SAS. The patient data must be unduplicated via the SORT procedure with the NODUPKEY option. This guarantees that the final counts of patients per site are correct. The MEANS procedure will then print out the total number of patients that will be evaluated for site matching (this is the population N). The site list is then brought into SAS via the IMPORT procedure and the ZIP Codes are properly formatted via a DATASTEP to run successfully through the program. This ensures that only 5 digit ZIP Codes are used and that the ZIP5 variable is a string variable. The Sites DATASET is then sorted with the NODUPKEY option to ensure that there are no duplicate ZIP Codes in the site list. As long as the DATASET Check1 is empty, the macro should proceed without issue.

```
DATA One;
        SET &QueryDataset;
RUN;
PROC SORT DATA = One NODUPKEY;
        BY Pat_ID;
RUN;
PROC MEANS DATA = One N;
        VAR Pat_ID;
        TITLE 'Total Patients in Query';
RUN;
FILENAME REFFILE &SITELISTLOCATION;
PROC IMPORT DATAFILE=REFFILE
        DBMS=XLSX
        OUT=WORK.Sites REPLACE;
        GETNAMES=YES;
RUN;
DATA Sites;
        SET Sites;
        ZipT = PUT(Zip,$12.);
        Zip5 = SUBSTR(ZipT,1,5);
RUN;
PROC SORT DATA = Sites NODUPKEY DUPOUT = Check1;
        BY Zip5;
RUN;
```

Globalizing the Site List

The next section of code turns the entire unduplicated site list into global macro variables that will be used to run through the ZIP Code radius macro. The first _NULL_ DATASTEP runs through ZIP Codes 1-9, the second _NULL_ DATASTEP runs through Zip Codes 10-99, and the third _NULL_ DATASTEP runs through ZIP Codes 100-999. Each DATASTEP takes a singular ZIP Code from the site list into a global macro variable called 'ZipCode*n*'. The *n* indicates the observation number of the ZIP Code from the site list. Once each observation in the site list has been placed into a global macro, the %PUT _ALL_; statement is used to print out in the log all the ZIP Codes from the site list. This can be reviewed to check that the process worked correctly and all ZIP Codes are present.

WARNING: The SAS code WILL error out on this section of code. ANY observation in the site list 0-9 will error out in the second and third DATASTEPS, ANY observation in the site list 10-99 will error out in the first and third DATASTEPS, and ANY observation in the site list 100-999 will error out in the first and second DATASTEPS. This is because the number of digits in those observation numbers are not correct for the _NULL_ DATASTEPS. **If these are the only errors in the program – the program ran successfully.**

```
data _null_;
set Sites;
call symput('ZipCode'|| put(_n_,1.), Zip5);
run;

data _null_;
set Sites;
call symput('ZipCode'|| put(_n_,2.), Zip5);
run;

data _null_;
set Sites;
call symput('ZipCode'|| put(_n_,3.), Zip5);
run;

%PUT _ALL_;
```

Creating Radius Datasets

Once the ZIP Codes have all been globalized, the next step is to run each of them through the macro ZipsNear that was previously explained. To do this, the macro 'macloop1' was created. This macro loops through all of the global ZipCode values and creates individual

datasets for each macro variable that contains a list of each ZIP Code within the pre-specified distance radius.

```
%macro macloop1;
%local j;
        %do j = 1 %to &&SiteNumber;
                %ZipsNear(base=&&zipcode&j,dist=&&DISTANCE,libout=work,dsnout=zip_&j);
        %end;
%mend macloop1;
%macloop1;
```

Determining Unique Patient Match

Once the radius DATASETS have been created, the DATASTEP Zip_Radius will combine all of the newly created DATASETS and sort them via the SORT procedure and the NODUPKEY option. This is so that only one instance of each ZIP Code will exist. By only allowing one instance of each ZIP code to exist, when merged with the patient dataset - the program will identify the unique number of patients that fall within the ZIP Code radius of all sites. The MEANS procedure is again used to output the unique number of patients matching to any site within the specified radius parameter previously designated (this would be the sample N).

```
DATA Zip_Radius;
        SET Zip_1 - Zip_&SITENUMBER;
RUN;
PROC SORT DATA = Zip_Radius NODUPKEY;
        BY ZipText;
RUN;
DATA One_2;
        SET One (RENAME = (Zip = Zip5);
RUN;
PROC SORT DATA = One_2;
        BY Zip;
RUN;
PROC MEANS DATA = Matched_Unique N;
        VAR Pat_Master_ID;
        TITLE 'Unique Patients Matching Criteria & Radius';
RUN;
```

Identifying Patients within the ZIP Code Radius

In order to identify the patients within the ZIP Code Radius, the FREQ procedure is used to create a dataset containing the number of patients within each ZIP code in the population DATASET. Once the dataset with patient count and ZIP Code information has been created, it

will be utilized in the Combine macro. The Combine macro takes the patient ZIP Code count DATASET and merges it with each ZIP Code Radius DATASET. The ZIP Codes and related counts are only kept if the patient ZIP Codes are in the radius listing. Once each radius DATASET has been combined with the patient information, the program combines all of the datasets via a DATASTEP to create a comprehensive list. Because the previous ZipsNear macro outputted the Base ZIP Code of each site that was evaluated or the ZIP Code from which the radius was drawn; the MEANS procedure can recreate the site list from the BaseZip variable and sum up the total number of patients matching to each site.

```
PROC FREQ DATA = One_2 NOPRINT;
        TABLES Zip / OUT = Zips_Study;
RUN;

%MACRO Combine(nobs=);
        %DO i=1 %TO &NOBS;
                PROC SORT DATA = Zip_&i;
                        BY ZipText;
                RUN;
                DATA Zip_2_&i;
                        MERGE Zip_&i  (IN = a RENAME = (ZipText=Pat_Zip)) Zips_Study
                        (RENAME = (Zip = Pat_Zip));
                        BY Pat_Zip;
                        IF A;
                        IF Pat_Zip = "" OR Count = . THEN DELETE;
                        DROP _NAME_ PERCENT;
                RUN;
        %END;
%MEND;

%Combine(nobs=&&SITENUMBER);

DATA Final_Radius;
        SET Zip_2_1-Zip_2_&SITENUMBER;
RUN;

PROC MEANS DATA = Final_Radius NOPRINT SUM;
        CLASS BaseZip;
        VAR Count;
        OUTPUT OUT = HeatMap SUM = Patients;
RUN;
```

Finishing It Up

The final step in creating a site listing with the number of patients matching to each site is to clean up the previous output. By using a DATASTEP, the program can remove any instances from the MEANS procedure where the overall sum was provided and can remove the unwanted variables _TYPE_ and _FREQ_. Once the DATASET is clean and ready, it will be outputted to the location specified by the user in the beginning of the program and will produce the site list with the number of patients matching to a certain site within a certain radius.

```
DATA HeatMap;
        SET HeatMap;
        IF BaseZip = '' THEN DELETE;
        DROP _TYPE_ _FREQ_;
RUN;

PROC EXPORT DATA = HeatMap
        OUTFILE = &OUTFILELOCATION
        DBMS = CSV REPLACE;
RUN;
```

Conclusion

This program has proven to be instrumentally useful in efficient identification of patients who are within a certain distance of relevant clinical trial sites. This program allows users the flexibility to choose varying distances for each iteration of the program and upload modified or new clinical trial site lists easily. In the future, this program will be enhanced to allow for automatic patient site assignment based on distance to the sites of interest and the proximity to other locations in the event that a patient matched to more than one site. The full SAS code can be found in the Appendix.

CONTACT INFORMATION

Your comments and questions are much appreciated. Contact the author at:

Kathryn Schurr, M.S.
Lead Health Information Analyst
Quest Diagnostics
Hudsonville, MI 49426
Work Phone: 616.340.0045
Work Email: Kathryn.M.Schurr@QuestDiagnostics.com

REFERENCES

K, Richard. "Returning Zip Codes in a Specified Radius." *sasCommunity.org*, 14 Sept. 2007, sascommunity.org/wiki/Returning_Zip_Codes_in_a_Specified_Radius.

APPENDIX: SAS Code

```
%let QueryDataSet               =       D.PosPats;
/* INPUT FINAL QUERY DATASET HERE */

%let OUTFILELOCATION  =       "/schurrk/ClinicalTrials/Trial1/SiteMatch.csv";
/* INPUT OUTPUT FILE LOCATION HERE */

%let SITELISTLOCATION  =       "/schurrk/ClinicalTrials/Trial1/Sites.xlsx";
/* INPUT SITE LIST LOCATION HERE - XLSX FORMAT, ZipCode must be named 'Zip' */

%let SITENUMBER                 =       20;
/* INPUT THE NUMBER OF SITES THAT ARE BEING MATCHED TO */

%let DISTANCE                   =       50;
/* INPUT THE DISTANCE OF RADIUS MATCH THAT IS OF INTEREST HERE */


 ***NO CHANGES BELOW THIS LINE***;

 DATA One;
        SET &QueryDataset;
RUN;

PROC SORT DATA = One NODUPKEY;
        BY Pat_ID;
RUN;

PROC MEANS DATA = One N;
        VAR Pat_ID;
        TITLE 'Total Patients in Query';
RUN;

FILENAME REFFILE &SITELISTLOCATION;

PROC IMPORT DATAFILE=REFFILE
        DBMS=XLSX
        OUT=WORK.Sites REPLACE;
        GETNAMES=YES;
RUN;

DATA Sites;
        SET Sites;
        ZipT = PUT(Zip,$12.);
        Zip5 = SUBSTR(ZipT,1,5);
```

```
RUN;

PROC SORT DATA = Sites NODUPKEY DUPOUT = Check1;
          BY Zip5;
RUN;

data _null_;
set Sites;
call symput('ZipCode'|| put(_n_,1.), Zip5);
run;

data _null_;
set Sites;
call symput('ZipCode'|| put(_n_,2.), Zip5);
run;

data _null_;
set Sites;
call symput('ZipCode'|| put(_n_,3.), Zip5);
run;

%PUT _ALL_;

%macro ZipsNear(base=,dist=,libout=work,dsnout=)
      /des='Returns zipcodes within x miles of a given zipcode.';
  %if &base eq %str() %then
  %do;
    %put %nrstr(%%)ZipsNear parameters:;
    %put %str( %()base  = Base Zipcode,;
    %put %str(  )libout = Output library,;
    %put %str(  )dsnout = Output dataset%str(%));
    %goto Quit;
  %end;
  proc sql noprint;
    %* fetch latitude and longitude of base zip code *;
    select y, x into :lat, :lon
    from   sashelp.zipcode
    where  zip=&base ;
    %* fetch zips within specified distance from base zip code *;
    create table &libout..&dsnout. as
    select
      zip,
      put(zip,z5.) as ZipText,
      put(&base,z5.) as BaseZip,
```

```
      3956 * (2 * arsin(min(1, sqrt((sin(((y - &lat) * constant('pi')/180)/2)**2) + ((cos(&lat *
constant('pi')/180)) * (cos(y * constant('pi')/180)) * (sin(((x- &lon) * constant('pi')/180)/2)**2)))))) as
distance
    from
     sashelp.zipcode
    where
     calculated distance <= &dist
    order by zip;
  quit;
  %Quit:
 %mend ZipsNear;

%macro macloop1;
%local j;
        %do j = 1 %to &&SiteNumber;
                %ZipsNear(base=&&zipcode&j,dist=&&DISTANCE,libout=work,dsnout=zip_&j);
        %end;
%mend macloop1;
%macloop1;

DATA Zip_Radius;
        SET Zip_1 - Zip_&SITENUMBER;
RUN;

PROC SORT DATA = Zip_Radius NODUPKEY;
        BY ZipText;
RUN;

DATA One_2;
        SET One;
        Zip = Zip5;
RUN;

PROC SORT DATA = One_2;
        BY Zip;
RUN;

DATA Matched_Unique;
        MERGE Zip_Radius (IN = a KEEP = ZipText RENAME = (ZipText = Zip))
                One_2 (IN = b);
        BY Zip;
        IF a AND b;
RUN;
```

```
PROC MEANS DATA = Matched_Unique;
	VAR Pat_Master_ID;
	TITLE 'Unique Patients Matching Criteria & Radius';
RUN;

PROC FREQ DATA = One_2 NOPRINT;
	TABLES Zip / OUT = Zips_Study;
RUN;

%MACRO Combine(nobs=);
	%DO i=1 %TO &NOBS;
		PROC SORT DATA = Zip_&i;
			BY ZipText;
		RUN;
		DATA Zip_2_&i;
			MERGE Zip_&i  (IN = a RENAME = (ZipText=Pat_Zip)) Zips_Study
			(RENAME = (Zip = Pat_Zip));
			BY Pat_Zip;
			IF A;
			IF Pat_Zip = "" OR Count = . THEN DELETE;
			DROP _NAME_ PERCENT;
		RUN;
	%END;
%MEND;

%Combine(nobs=&&SITENUMBER);

DATA Final_Radius;
	SET Zip_2_1-Zip_2_&SITENUMBER;
RUN;

PROC MEANS DATA = Final_Radius NOPRINT SUM;
	CLASS BaseZip;
	VAR Count;
	OUTPUT OUT = HeatMap SUM = Patients;
RUN;

DATA HeatMap;
	SET HeatMap;
	IF BaseZip = '' THEN DELETE;
	DROP _TYPE_ _FREQ_;
RUN;

PROC EXPORT DATA = HeatMap
```

```
        OUTFILE = &OUTFILELOCATION
        DBMS = CSV REPLACE;
RUN;
```