Paper BI02-2016

Macro method to use Google Maps™ and SAS® to find the shortest driving and straight line distances between 2 addresses in the United States

Laurie Bishop, Cincinnati Children's Hospital Medical Center, Cincinnati, Ohio

ABSTRACT

Google Maps™ is a very useful tool in finding driving distances between addresses and defining a geographic representation of an address which can be used to find straight line distances between two addresses. Often distances between addresses and geocoding for locations can be useful in research. Using macro and data step SAS® code, the shortest driving distance between two addresses can be found by searching the html code resulting from a Google Maps™ driving directions search between two addresses in the United States. In addition, a search of the html code resulting from a singular address Google Maps™ search, will enable one to define the latitude and longitude for a location. This will allow use of the SAS® GEODIST function to find the straight line distance between locations. Partial addresses are taken into consideration with this method, which was developed using SAS® v9.3 for Windows®.

INTRODUCTION

The methods introduced will allow a SAS® Programmer, with basic SAS® programming skills, to use Google™ and SAS® to find all available driving distances between two US addresses specified in a data set of address pairs, determine which is the shortest, and retain it. These methods will also provide the ability to isolate the latitude and longitude of any address search in Google Maps™, and use these values to find the straight line distance between two US addresses specified in a data set of address pairs.

SHORTEST DRIVING DISTANCE

SAS® can use Google Maps™ to find route dependent distances between addresses. Google Maps™ uses specific formats for the address that its search engine will accept, dependent on the parts of the address present. If the address is partially defined, the format of the address entered into Google Maps™ must be properly defined.

FORMAT ADDRESSES FOR GOOGLE MAPS™

The SAS® data set *AddressX* (Display 1) contains starting address details defined in four separate variables, the starting address defined with Address (street number with street address), City1 (US city name), State1 (US State abbreviation), and Zip1 (US 5 digit zip code), and the ending address defined with Hospital_Address (street number with street address), HCity (US city name), HState (US State abbreviation), and HZip (US zip code). The following data step formats the address for a Google Maps™ search depending on the parts of the address that are defined, taking into consideration missing variables. After running the code below against the *AddressX* data set, the variables of the addresses will be concatenated with proper commas and plus (+) signs as one address where the starting address will be Address1 and ending address will be Address2 in data set *Address* (Display 2). The code also counts the number of observations in the *AddressX* for later use in the macros:

	Subjid	Address	City1	State1	Zip1	Hospital Address	Hcity	Hstate	Hzip
1	001	705 Riley Hospital Dr	Indianapolis	IN	46202	3333 Burnet Ave	CINCINNATI	OH	45229
2	002	375 Dixmyth Ave	CINCINNATI	OH	45220	3333 Burnet Ave	CINCINNATI	OH	45229
3	003	630 Eaton Ave	HAMILTON	OH	45013	3333 Burnet Ave	CINCINNATI	OH	45229
4	004	1 Childrens Plaza	DAYTON	OH	45404	3333 Burnet Ave	CINCINNATI	OH	45229
5	005	1950 Mt St Marys Dr	Nelsonville	OH	45764	3333 Burnet Ave	CINCINNATI	OH	45229
6	006	550 1st Ave	New York	NY	10016	3333 Burnet Ave	CINCINNATI	OH	45229
7	007	225 E Chicago Ave	Chicago	IL	60611	3333 Burnet Ave	CINCINNATI	OH	45229
8	008	2500 E Van Buren St	Phoenix	AR	85008	3333 Burnet Ave	CINCINNATI	OH	45229
9	009	4800 Sand Point Way NE	Seattle	WA	98105	3333 Burnet Ave	CINCINNATI	OH	45229
10	010		Indianapolis	IN	46202	3333 Burnet Ave	CINCINNATI	OH	45229
11	011	705 Riley Hospital Dr	Indianapolis	IN	.	3333 Burnet Ave	CINCINNATI	OH	45229
12	012	705 Riley Hospital Dr	Indianapolis		46202	3333 Burnet Ave	CINCINNATI	OH	45229
13	013		Indianapolis	IN	.	3333 Burnet Ave	CINCINNATI	OH	45229
14	014		Indianapolis		.	3333 Burnet Ave	CINCINNATI	OH	45229
15	015	705 Riley Hospital Dr		IN	46202	3333 Burnet Ave	CINCINNATI	OH	45229
16	016	705 Riley Hospital Dr	Indianapolis		.	3333 Burnet Ave	CINCINNATI	OH	45229
17	017	705 Riley Hospital Dr		IN	.	3333 Burnet Ave	CINCINNATI	OH	45229
18	018	705 Riley Hospital Dr			46202	3333 Burnet Ave	CINCINNATI	OH	45229
19	019		Indianapolis		46202	3333 Burnet Ave	CINCINNATI	OH	45229
20	020			IN	46202	3333 Burnet Ave	CINCINNATI	OH	45229
21	021	705 Riley Hospital Dr			.	3333 Burnet Ave	CINCINNATI	OH	45229
22	022		Indianapolis		.	3333 Burnet Ave	CINCINNATI	OH	45229
23	023			IN	.	3333 Burnet Ave	CINCINNATI	OH	45229
24	024				46202	3333 Burnet Ave	CINCINNATI	OH	45229

Display 1. AddressX SAS® data set

```
/*Adjust format of address to be Google Maps™ friendly considering missing
variables*/
data Address;
      retain subjid Address1 Address2;
      set AddressX nobs=obsnum;
      length Address1 Address2 $200.;
      format Address1 Address2 $200.;
      informat Address1 Address2 $200.;
*Format starting address;
if Address^='' and City1^='' and State1^='' and Zip1^=. then
      Address1=tranwrd(left(trim(Address)),"","+")||"+"||
      tranwrd(left(trim(City1))," ","+")||","||left(trim(State1))||"+"||
      left(strip(put(Zip1,best12.)));
else if Address^='' and State1^='' and Zip1^=. then
      Address1=tranwrd(left(trim(Address))," ","+")||"+"||
      left(trim(State1))||"+"||left(strip(put(Zip1,best12.)));
else if Address^='' and City1^='' and Zip1^=. then
      Address1=tranwrd(left(trim(Address))," ","+")||"+"||
      tranwrd(left(trim(City1))," ","+")||"+"||
      left(strip(put(Zip1,best12.)));
else if Address^='' and City1^='' and State1^='' then
      Address1=tranwrd(left(trim(Address))," ","+")||"+"||
      tranwrd(left(trim(City1))," ","+")||","||left(trim(State1));
else if City1^='' and State1^='' and Zip1^=. then
      Address1=tranwrd(left(trim(City1))," ","+")||","||left(trim(State1))||
      "+"||left(strip(put(Zip1,best12.)));
else if Address^='' and City1^='' then Address1=tranwrd(left(trim(Address)),
      " ","+")||"+"||tranwrd(left(trim(City1))," ","+");
else if Address^='' and State1^='' then
      Address1=tranwrd(left(trim(Address))," ","+")||"+"||left(trim(State1));
else if Address^='' and Zip1^=. then Address1=tranwrd(left(trim(Address)),
      " ","+")||"+"||left(strip(put(Zip1,best12.)));
else if City1^='' and Zip1^=. then Address1=tranwrd(left(trim(City1)),
      " ","+")||"+"||left(strip(put(Zip1,best12.)));
```

```
else if City1^='' and State1^='' then Address1=tranwrd(left(trim(City1)),
      " ","+")||","||left(trim(State1));
else if State1^='' and Zip1^=. then Address1=left(trim(State1))||"+"||
      left(strip(put(Zip1,best12.)));
else if Zip1^=. then Address1=left(strip(put(Zip1,best12.)));
else if Address^='' then Address1=tranwrd(left(trim(Address)),," ","+");
else if City1^='' then Address1=tranwrd(left(trim(City1)),," ","+");
else if State1^='' then Address1=left(trim(State1));
*Format ending address;
if Hospital_Address^='' and HCity^='' and HState^='' and HZip^=. then
      Address2=tranwrd(left(trim(Hospital_Address)),," ","+")||"+"||
      tranwrd(left(trim(HCity)),," ","+")||","||left(trim(HState))||"+"||
      left(strip(put(HZip,best12.)));
else if Hospital_Address^='' and HState^='' and HZip^=. then
      Address2=tranwrd(left(trim(Hospital_Address)),," ","+")||
      "+"||left(trim(HState))||"+"||left(strip(put(HZip,best12.)));
else if Hospital_Address^='' and HCity^='' and HZip^=. then
      Address2=tranwrd(left(trim(Hospital_Address)),," ","+")||
      ",+"||tranwrd(left(trim(HCity)),," ","+")||"+"||
      left(strip(put(HZip,best12.)));
else if Hospital_Address^='' and HCity^='' and HState^='' then
      Address2=tranwrd(left(trim(Hospital_Address)),," ","+")||
      ",+"||tranwrd(left(trim(HCity)),," ","+")||","||left(trim(HState));
else if HCity^='' and HState^='' and HZip^=. then
      Address2=tranwrd(left(trim(HCity)),," ","+")||","||left(trim(HState))||
      "+"||left(strip(put(HZip,best12.)));
else if Hospital_Address^='' and HCity^='' then
      Address2=tranwrd(left(trim(Hospital_Address)),," ","+")||",+"||
      tranwrd(left(trim(HCity)),," ","+");
else if Hospital_Address^='' and HState^='' then
      Address2=tranwrd(left(trim(Hospital_Address)),," ","+")||"+"||
      left(trim(HState));
else if Hospital_Address^='' and HZip^=. then
      Address2=tranwrd(left(trim(Hospital_Address)),," ","+")||"+"||
      left(strip(put(HZip,best12.)));
else if HCity^='' and HZip^=. then Address2=tranwrd(left(trim(HCity)),
      " ","+")||"+"||left(strip(put(HZip,best12.)));
else if HCity^='' and HState^='' then Address2=tranwrd(left(trim(HCity)),
      " ","+")||","||left(trim(HState));
else if HState^='' and HZip^=. then
      Address2=left(trim(HState))||"+"||left(strip(put(HZip,best12.)));
else if HZip^=. then Address2=left(strip(put(HZip,best12.)));
else if Hospital_Address^='' then
      Address2=tranwrd(left(trim(Hospital_Address)),," ","+");
else if HCity^='' then Address2=tranwrd(left(trim(HCity)),," ","+");
else if HState^='' then Address2=left(trim(HState));
*Assign macro variable obsnum=number of observations in the data set;
call symput('obsnum',obsnum);
run;
```

	Subjid	Address1	Address2
1	001	705+Riley+Hospital+Dr+Indianapolis,IN+46202	3333+Burnet+Ave+CINCINNATI,OH+45229
2	002	375+Dixmyth+Ave+CINCINNATI,OH+45220	3333+Burnet+Ave+CINCINNATI,OH+45229
3	003	630+Eaton+Ave+HAMILTON,OH+45013	3333+Burnet+Ave+CINCINNATI,OH+45229
4	004	1+Childrens+Plaza+DAYTON,OH+45404	3333+Burnet+Ave+CINCINNATI,OH+45229
5	005	1950+Mt+St+Marys+Dr+Nelsonville,OH+45764	3333+Burnet+Ave+CINCINNATI,OH+45229
6	006	550+1st+Ave+New+York,NY+10016	3333+Burnet+Ave+CINCINNATI,OH+45229
7	007	225+E+Chicago+Ave+Chicago,IL+60611	3333+Burnet+Ave+CINCINNATI,OH+45229
8	008	2500+E+Van+Buren+St+Phoenix,AR+85008	3333+Burnet+Ave+CINCINNATI,OH+45229
9	009	4800+Sand+Point+Way+NE+Seattle,WA+98105	3333+Burnet+Ave+CINCINNATI,OH+45229
10	010	Indianapolis,IN+46202	3333+Burnet+Ave+CINCINNATI,OH+45229
11	011	705+Riley+Hospital+Dr+Indianapolis,IN	3333+Burnet+Ave+CINCINNATI,OH+45229
12	012	705+Riley+Hospital+Dr+Indianapolis+46202	3333+Burnet+Ave+CINCINNATI,OH+45229
13	013	Indianapolis,IN	3333+Burnet+Ave+CINCINNATI,OH+45229
14	014	Indianapolis	3333+Burnet+Ave+CINCINNATI,OH+45229
15	015	705+Riley+Hospital+Dr+IN+46202	3333+Burnet+Ave+CINCINNATI,OH+45229
16	016	705+Riley+Hospital+Dr+Indianapolis	3333+Burnet+Ave+CINCINNATI,OH+45229
17	017	705+Riley+Hospital+Dr+IN	3333+Burnet+Ave+CINCINNATI,OH+45229
18	018	705+Riley+Hospital+Dr+46202	3333+Burnet+Ave+CINCINNATI,OH+45229
19	019	Indianapolis+46202	3333+Burnet+Ave+CINCINNATI,OH+45229
20	020	IN+46202	3333+Burnet+Ave+CINCINNATI,OH+45229
21	021	705+Riley+Hospital+Dr	3333+Burnet+Ave+CINCINNATI,OH+45229
22	022	Indianapolis	3333+Burnet+Ave+CINCINNATI,OH+45229
23	023	IN	3333+Burnet+Ave+CINCINNATI,OH+45229
24	024	46202	3333+Burnet+Ave+CINCINNATI,OH+45229

Display 2. Address1 and Address2 variables added to AddressX data set to form Address SAS® data set

FIND SHORTEST DRIVING DISTANCE

First, create an empty data set to append all distances (*drdist*). The *ShortDrDist* macro loops through each observation of the *Address* data set containing the formatted starting and ending dates. The loop begins with a data step to assign macro variables used throughout the macro. A filename statement for Google Maps™ url is defined with the starting and ending addresses. The next data step uses an infile statement to read in the html code from the Google Maps™ page in 1000 character substrings (sitecode). Using the FIND function, sitecode is searched for the word '*miles*', which is always proceeded by the driving distance between the starting and ending addresses in miles, and only these observations are retained. All driving distances available will be defined as drivdist, whose value is assigned by isolating the numeric distance using the SCAN function with sitecode. Use SORT procedure to sort driving distance results by distance (drivdist). Use a data step to retain the smallest of all the distances. Append the observation to final driving distances data set (drdist). Always be sure to disassociate current assigned url. The resulting data set is a collection of the shortest driving distances (Display 3):

```
*Create empty data set to append all distances;
data drdist;
run;

%macro ShortDrDist(j=);
      *Create loop to access each observation;
      %do i=1 %to &j;
      *Assign macro variables needed;
      data _null_;
            set Address;
      if _n_=&i then do;
            call symput('addr1',trim(strip(Address1)));
            call symput('addr2',trim(strip(Address2)));
            call symput('Subjid',Subjid);
```

```
      end;
      run;

      *Find Road distances;
      filename gmaps url
            "http://maps.google.com/maps?daddr=&addr1.%nrstr(&saddr)=&addr2";
      data drdistX&i (drop=sitecode idrivdist rdstart);
            infile gmaps recfm=f lrecl=1000 end=eof;
            input sitecode $1000.;
            Subjid="&subjid";
            AddStart=tranwrd(tranwrd("&addr1","+"," "),",",", ");
            AddEnd=tranwrd(tranwrd("&addr2","+"," "),",",", ");
            if find(sitecode,'miles');
            idrivdist=find(sitecode,'miles');
            rdstart=idrivdist-1;
            drivdist=input(compress(scan(substr(sitecode,1,rdstart),1,
                     ' "','bc'),','),best12.);
      run;
      proc sort data=drdistX&i; by drivdist; run;
      data drdist&i;
            set drdistX&i;
            by drivdist;
      if _n_=1;
      run;
      data drdist;
            set drdist
                drdist&i;
      if drivdist^=.;
      label AddStart='Starting Address'
            AddEnd='Ending Address'
            drivdist='Shortest Driving Distance (miles)'
            Subjid='Subject Id';
      run;
      filename gmaps clear;
      %end;
%mend;

%ShortDrDist(j=&obsnum);
```

	Subject Id	Starting Address	Ending Address	Shortest Driving Distance (miles)
1	001	705 Riley Hospital Dr Indianapolis, IN 46202	3333 Burnet Ave, CINCINNATI, OH 45229	115
2	002	375 Dixmyth Ave CINCINNATI, OH 45220	3333 Burnet Ave, CINCINNATI, OH 45229	1.6
3	003	630 Eaton Ave HAMILTON, OH 45013	3333 Burnet Ave, CINCINNATI, OH 45229	26.4
4	004	1 Childrens Plaza DAYTON, OH 45404	3333 Burnet Ave, CINCINNATI, OH 45229	52
5	005	1950 Mt St Marys Dr Nelsonville, OH 45764	3333 Burnet Ave, CINCINNATI, OH 45229	153
6	006	550 1st Ave New York, NY 10016	3333 Burnet Ave, CINCINNATI, OH 45229	641
7	007	225 E Chicago Ave Chicago, IL 60611	3333 Burnet Ave, CINCINNATI, OH 45229	296
8	008	2500 E Van Buren St Phoenix, AR 85008	3333 Burnet Ave, CINCINNATI, OH 45229	1788
9	009	4800 Sand Point Way NE Seattle, WA 98105	3333 Burnet Ave, CINCINNATI, OH 45229	2331
10	010	Indianapolis, IN 46202	3333 Burnet Ave, CINCINNATI, OH 45229	114
11	011	705 Riley Hospital Dr Indianapolis, IN	3333 Burnet Ave, CINCINNATI, OH 45229	115
12	012	705 Riley Hospital Dr Indianapolis 46202	3333 Burnet Ave, CINCINNATI, OH 45229	115
13	013	Indianapolis, IN	3333 Burnet Ave, CINCINNATI, OH 45229	112
14	014	Indianapolis	3333 Burnet Ave, CINCINNATI, OH 45229	112
15	015	705 Riley Hospital Dr IN 46202	3333 Burnet Ave, CINCINNATI, OH 45229	115
16	016	705 Riley Hospital Dr Indianapolis	3333 Burnet Ave, CINCINNATI, OH 45229	115
17	017	705 Riley Hospital Dr IN	3333 Burnet Ave, CINCINNATI, OH 45229	115
18	018	705 Riley Hospital Dr 46202	3333 Burnet Ave, CINCINNATI, OH 45229	115
19	019	Indianapolis 46202	3333 Burnet Ave, CINCINNATI, OH 45229	114
20	020	IN 46202	3333 Burnet Ave, CINCINNATI, OH 45229	114
21	021	705 Riley Hospital Dr	3333 Burnet Ave, CINCINNATI, OH 45229	115
22	022	Indianapolis	3333 Burnet Ave, CINCINNATI, OH 45229	112
23	023	IN	3333 Burnet Ave, CINCINNATI, OH 45229	141
24	024	46202	3333 Burnet Ave, CINCINNATI, OH 45229	114

Display 3. drdist SAS® data set

STRAIGHT LINE DISTANCE

FIND LATITUDE AND LONGITUDE OF AN ADDRESS (GEOCODE)

First, create empty data sets to append starting address (sldist1) and ending address (sldist2) latitude/ longitude pairs. The *StrLineDist* macro loops through each observation of the *Address* data set containing the formatted starting and ending dates. The loop begins with a data step to assign macro variables used throughout the macro. A filename statement for Google Maps™ url is defined with a singular address. The following data step uses an infile statement to read in the html code from the Google Maps™ page in 1000 character substrings (sitecode). Using the FIND function, sitecode is searched for the a unique character string (",[null,null,) that always proceeds the latitude and longitude corresponding to the address defined and the observation is retain. The numeric latitude (lat1, lat2) and longitude (long1, long2) are assigned by using the SCAN function with sitecode to isolate the numeric value of each. Observations with latitude/longitude pairs are appended to sldist1 for starting addresses and sldist2 for ending addresses (Display 4). Always be sure to disassociate current assigned url:

```
*Create empty data sets to append latitudes and longitudes;
data sldist1;
run;
data sldist2;
run;
%macro StrLineDist(j=);

      *Create loop to access each observation;
      %do i=1 %to &j;
            data _null_;
                  set Address;
            if _n_=&i then do;
                  call symput('addr1',trim(strip(Address1)));
                  call symput('addr2',trim(strip(Address2)));
                  call symput('Subjid',Subjid);
            end;
            run;

            /*Find Latitude and longitude in Google Maps™ page code*/
            filename gmaps url "http://maps.google.com/maps?daddr=&addr1";
```

```
            *For starting addresses;
            data sldist1&i (drop=sitecode markst);
                  infile gmaps recfm=f lrecl=1000 end=eof;
                  input sitecode $1000.;
            Subjid="&subjid";
            AddStart=tranwrd(tranwrd("&addr1","+"," "),",",", ");
            if find(sitecode,'",[null,null,') then do;
                  markst=find(sitecode,'",[null,null,');
                  lat1=input(scan(substr(sitecode,markst),4,',]'),best12.);
                  long1=input(scan(substr(sitecode,markst),5,',]'),best12.);
                  output;
            end;
            run;
            data sldist1;
                  set sldist1
                        sldist1&i;
            if lat1^=.;
            label AddStart='Starting Address'
                    lat1='Starting Address latitude'
                    long1='Starting Address longitude'
                    Subjid='Subject Id';
            run;
            proc sort data=sldist1; by Subjid; run;
            filename gmaps clear;
            *For ending addresses;
            filename gmaps url "http://maps.google.com/maps?daddr=&addr2";
            data sldist2&i (drop=sitecode markst);
                  infile gmaps recfm=f lrecl=1000 end=eof;
                  input sitecode $1000.;
            Subjid="&subjid";
            AddEnd=tranwrd(tranwrd("&addr2","+"," "),",",", ");
            if find(sitecode,'",[null,null,') then do;
                  markst=find(sitecode,'",[null,null,');
                  lat2=input(scan(substr(sitecode,markst),4,',]'),best12.);
                  long2=input(scan(substr(sitecode,markst),5,',]'),best12.);
                  output;
            end;
            run;
            data sldist2;
                  set sldist2
                        sldist2&i;
            if lat2^=.;
            label AddEnd='Ending Address'
                    lat2='Ending Address latitude'
                    long2='Ending Address longitude'
                    Subjid='Subject Id';
            run;
            proc sort data=sldist2; by Subjid; run;
            filename gmaps clear;
      %end;
%mend;

%StrLineDist(j=&obsnum);
```

	Subject Id	Starting Address	Starting Address latitude	Starting Address longitude
1	001	705 Riley Hospital Dr Indianapolis, IN 46202	39.7774285	-86.1800805
2	002	375 Dixmyth Ave CINCINNATI, OH 45220	39.1400349	-84.52067269
3	003	630 Eaton Ave HAMILTON, OH 45013	39.4150602	-84.57376939
4	004	1 Childrens Plaza DAYTON, OH 45404	39.774902999	-84.16721
5	005	1950 Mt St Marys Dr Nelsonville, OH 45764	39.4632713	-82.24312449
6	006	550 1st Ave New York, NY 10016	40.742069699	-73.9743355
7	007	225 E Chicago Ave Chicago, IL 60611	41.8962515	-87.6219521
8	008	2500 E Van Buren St Phoenix, AR 85008	33.4525898	-112.0258737
9	008	2500 E Van Buren St Phoenix, AR 85008	33.4548339	-112.0291224
10	009	4800 Sand Point Way NE Seattle, WA 98105	47.6626378	-122.283265
11	010	Indianapolis, IN 46202	39.7794767	-86.17008939
12	011	705 Riley Hospital Dr Indianapolis, IN	39.7774285	-86.1800805
13	012	705 Riley Hospital Dr Indianapolis 46202	39.7774285	-86.1800805
14	013	Indianapolis, IN	39.768403	-86.158068
15	014	Indianapolis	39.768403	-86.158068
16	015	705 Riley Hospital Dr IN 46202	39.7774285	-86.1800805
17	016	705 Riley Hospital Dr Indianapolis	39.7774285	-86.1800805
18	017	705 Riley Hospital Dr IN	39.7774285	-86.1800805
19	018	705 Riley Hospital Dr 46202	39.7774285	-86.1800805
20	019	Indianapolis 46202	39.7794767	-86.17008939
21	020	IN 46202	39.7794767	-86.17008939
22	021	705 Riley Hospital Dr	39.7773936	-86.18062189
23	022	Indianapolis	39.768403	-86.158068
24	023	IN	40.2671941	-86.1349019
25	024	46202	39.7794767	-86.17008939

Display 4. sldist1 SAS® data set

FIND STRAIGHTLINE DISTANCE BETWEEN ADDRESSES

Create the sldist data set by merging sldist1 and sldist2by subjid created by the *StrLineDist* macro above. Define sldistg which is a calculation of straight line distance between the two addresses using latitude/longitude pairs for each observation in the GEODIST function and define sldisth which is the calculation of straight line distance between two addresses using latitude/longitude pairs for each observation and Haversine formula $(3949.99\cos^{-1}(sin(lat1(\pi/180)) * sin(lat2(\pi/180)) + cos(lat1(\pi/180)))$. Create dist data set by merging both drdist and sldist to include both shortest driving distance and straight line distance in one data set:

```
*Merge starting and ending data sets by subjid;
data sldist;
      merge sldist1
              sldist2;
      by Subjid;
sldistg=geodist(lat1,long1,lat2,long2,'M');
sldisth=3949.99*arcos(sin(lat1*(constant('pi')/180))*
        sin(lat2*(constant('pi')/180))+cos(lat1*(constant('pi')/180))*
        cos(lat2*(constant('pi')/180))*cos(long2*(constant('pi')/180)-
        long1*(constant('pi')/180)));
label sldistg='GEODIST function distance (miles)'
      sldisth='Haversine formula distance (miles)';
run;

*Merge both distance variables into one data set;
data dist;
      retain subjid AddStart lat1 long1 AddEnd lat2 long2 drivdist sldistg
```

```
            sldisth;
      merge sldist (drop=AddEnd AddStart)
            drdist;
      by subjid;
run;
proc sort data=dist; by subjid; run;
```

	Subject Id	Starting Address latitude	Starting Address longitude	Ending Address latitude	Ending Address longitude	SAS GEODIST function	Haversine formula distance
1	001	39.7774285	-86.1800805	39.140907899	-84.5015463	99.947895341	99.536249259
2	002	39.1400349	-84.52067269	39.140907899	-84.5015463	1.0294637485	1.0244607343
3	003	39.4150602	-84.57376939	39.140907899	-84.5015463	19.308910803	19.289133233
4	004	39.774902999	-84.16721	39.140907899	-84.5015463	47.261063743	47.191810306
5	005	39.4632713	-82.24312449	39.140907899	-84.5015463	123.09439338	122.50950943
6	006	40.742069699	-73.9743355	39.140907899	-84.5015463	569.62112447	566.90359627
7	007	41.8962515	-87.6219521	39.140907899	-84.5015463	251.29965595	250.61316499
8	008	33.4525898	-112.0258737	39.140907899	-84.5015463	1579.171073	1572.1551899
9	008	33.4548339	-112.0291224	39.140907899	-84.5015463	1579.286024	1572.2691218
10	009	47.6626378	-122.283265	39.140907899	-84.5015463	1969.0227412	1959.5763237
11	010	39.7794767	-86.17008939	39.140907899	-84.5015463	99.529461003	99.12067847
12	011	39.7774285	-86.1800805	39.140907899	-84.5015463	99.947895341	99.536249259
13	012	39.7774285	-86.1800805	39.140907899	-84.5015463	99.947895341	99.536249259
14	013	39.768403	-86.158068	39.140907899	-84.5015463	98.621805048	98.215515032
15	014	39.768403	-86.158068	39.140907899	-84.5015463	98.621805048	98.215515032
16	015	39.7774285	-86.1800805	39.140907899	-84.5015463	99.947895341	99.536249259
17	016	39.7774285	-86.1800805	39.140907899	-84.5015463	99.947895341	99.536249259
18	017	39.7774285	-86.1800805	39.140907899	-84.5015463	99.947895341	99.536249259
19	018	39.7774285	-86.1800805	39.140907899	-84.5015463	99.947895341	99.536249259
20	019	39.7794767	-86.17008939	39.140907899	-84.5015463	99.529461003	99.12067847
21	020	39.7794767	-86.17008939	39.140907899	-84.5015463	99.529461003	99.12067847
22	021	39.7773936	-86.18062189	39.140907899	-84.5015463	99.972869382	99.561073508
23	022	39.768403	-86.158068	39.140907899	-84.5015463	98.621805048	98.215515032
24	023	40.2671941	-86.1349019	39.140907899	-84.5015463	116.69917479	116.33235296
25	024	39.7794767	-86.17008939	39.140907899	-84.5015463	99.529461003	99.12067847

Display 5. Straight Line Distances SAS® data set

CONCLUSION

The macros presented allows the user to find the shortest driving distance between two US addresses, but more importantly it allows a user to isolate the latitude and longitude of a particular address using SAS® and Google Maps™. A couple of things to note: Running this code against 24 observations takes longer than usual. Also, if a facility is large enough with multiple buildings/businesses at one address, it is possible that more than one latitude/longitude pair may be found (see subjid 008 and 021 above. The address for subjid 008 refers to the AZ state Hospital and State Prison and for subjid 021, the address is shared by Riley Hospital for Children and a doctor's office. In both of these cases the locations are described by one address but each facility has its own latitude and longitude.)

REFERENCES

Bekkerman, Anton, Ph.D. 2010. "Going the Distance: Google Maps™ Capabilities in a Friendly SAS Environment." http://wuss.org/Proceedings13/100_Paper.pdf

Zdeb, Mike. 2010. "Driving Distances and Drive Times using SAS® and Google Maps™." *Coders' Corner*. Seattle, WA: SAS® Gobal Forum, http://support.sas.com/resources/papers/proceedings10/050-2010.pdf

CONTACT INFORMATION

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

Laurie Bishop
Cincinnati Children's Hospital Medical Center

(513) 803-9001
Laurie.bishop@cchmc.org