

Dynamically Changing Time Zones and Daylight Savings on Time Series Data

Chao-Ying Hsieh, Southern Company Services, Inc., Atlanta, GA

ABSTRACT

SAS® programmers commonly deal with time zones and daylight saving time changes, especially when working for a large corporation with multiple subsidiaries located in different time zones. This paper presents a dynamic way to convert GMT time to local time with daylight saving time changes accounted for on time series data collected from smart meters. The technique uses SAS® functions, formats, and macros to create a program that streamlines the time conversion process.

INTRODUCTION

Southern Company has installed smart meters for its customers in states spanning across the Eastern and Central time zones. The data received from the smart meters are in Greenwich Mean Time (GMT) which differs from the customers' local time zones. The dataset needs to be converted to the local time with daylight saving time applied before the load research analysts perform the analysis. The conversion programs are hard coded to account for different time zones and daylight savings changes. This makes the conversion programs difficult to maintain and share between different time zones. Therefore, a standardized program was created that does not require code changes and can be shared across different divisions.

TIME SERIES DATA FROM SMART METERS

All of the data collected from smart meters at Southern Company's electric utility companies are stored in the Smart Energy Data Store, an Oracle database. Table 1 lists typical information collected from smart meters. Each record has a timestamp of transaction and register read of electricity usage. The measurements of register read are made at regular time interval. In this paper, I only show hourly interval data for the purpose of the presentation. All the timestamp from the smart meters is set for GMT time regardless of the location of the meters.

METER_NUMBER	READING_TIMESTAMP	READING_VALUE
xxxxx08	09MAR15:12:00:00.00 AM	61513
xxxxx08	09MAR15:01:00:00.00 AM	61516
xxxxx08	09MAR15:02:00:00.00 AM	61518
xxxxx08	09MAR15:03:00:00.00 AM	61519
xxxxx08	09MAR15:04:00:00.00 AM	61519
xxxxx08	09MAR15:05:00:00.00 AM	61520

Table 1. Typical Information from Smart Meter

LOAD RESEARCH & HOURLY DATA

According to the Association of Edison Illuminating Companies (AEIC), load research is "an activity embracing the measurement and study of the characteristics of electric loads to provide a thorough & reliable knowledge of trends, and general behavior of the load characteristics of the customers serviced by the electrical industry." Load research analysts at Southern Company have been analyzing a variety of electricity data to understand their customers' electricity consumption.

To do so, Southern Company's input data must be set in line with the customers' time. This includes not only the time zone but also daylight saving time. Daylight saving time (DST) is in effect from the second Sunday of March at 2 a.m. (local time) to the first Sunday of November at 2 a.m. (local time).

DYNAMICALLY CHANGING TIME ZONE

Function `tzzoneoff` is one of the new functions in SAS 9.4. By providing a time zone id in the argument, it will return the offset value from GMT time. Using this function, SAS programmers can add the offset value to the GMT time in order to get the local time.

The following syntax shows this basic functionality. It creates a new variable called 'LOCALTIME' which converts the variable `READING_TIMESTAMP` from GMT to EDT.

```
localtime=reading_timestamp+tzzoneoff('EDT');
```

For a list of time zone IDs and time zone names, please refer to the SAS website:

<http://support.sas.com/documentation/cdl/en/nlsref/67964/HTML/default/viewer.htm#n1tj735aocxmw7n1kfoz1qpdvb9l.htm>. To diminish the other analysts' time spent figuring out what time zone id they need to use, I used the macro variable `LOCTIME` to store a time zone id and mapped `LOCTIME` with the internal company identification number (macro variable `COMPANY`) in my program. The whole SAS program was stored as a SAS macro. The user of this program will only need to put in their company identification code to convert GMT into their local time.

```
%if &company= 2%then%do;
    %let loctime=EDT;
%end;
%else %do;
    %let loctime=CDT;
%end;

...more codes...

localtime=reading_timestamp+tzzoneoff("&loctime");
```

Table 2 has the results of conversion:

METER_NUMBER	READING_TIMESTAMP	LOCALTIME	READING_VALUE
xxxxx08	09MAR15:12:00:00.00 AM	08MAR15:08:00:00.00 PM	61513
xxxxx08	09MAR15:01:00:00.00 AM	08MAR15:09:00:00.00 PM	61516
xxxxx08	09MAR15:02:00:00.00 AM	08MAR15:10:00:00.00 PM	61518
xxxxx08	09MAR15:03:00:00.00 AM	08MAR15:11:00:00.00 PM	61519
xxxxx08	09MAR15:04:00:00.00 AM	09MAR15:12:00:00.00 AM	61519
xxxxx08	09MAR15:05:00:00.00 AM	09MAR15:01:00:00.00 AM	61520

Table 2. The Results of Conversion by Using TZZONEOFF Function

DYNAMICALLY FINDING DAYLIGHT SAVING TIME

We only need our time set to daylight saving time when daylight saving time is in effect. Rest of the time it should be set to standard time. The challenge is to find the hours for when the daylight saving time starts and ends every year. At present, daylight saving time in the United States begins at 2:00 a.m. on the second Sunday of March and ends at 2:00 a.m. on the first Sunday of November.

The following syntax shows how to use SAS functions to find when daylight saving time is in effect in the year 2015:

```
Year=2015;                                ← Set variable year as 2015
dst_beg=NWKDOM(2,1,3, YEAR);              ← DST begin date
```

```
dst_end=NWKDOM(1,1,11, YEAR);
```

← DST end date

The above code only works for a specific year. I need the program to be able to identify DST effective dates for any given years. In order to do that, I created a dataset to store the daylight saving time effective dates, then used it to create a SAS format GMTOFF.

```
data fmt(drop=year);
  attrib hlo length=$1
  start end format=datetime20.;
  fmtname="gmtoff";
  type="N";
  do year=2000 to 2100;

    dst_beg=NWKDOM(2,1,3, YEAR);
    dst_end=NWKDOM(1,1,11, YEAR);

    start=dst_beg*86400 + 3600*3;
    end=dst_end*86400 + 3600;
    label=0;

    output;

  end;

  start=.;end=.;
  hlo="O";
  label=-3600;
  output;
run;

proc format cntlin=fmt;run;
```

hlo	start	end	fmtname	type	label
	10MAR2097:03:00:00	03NOV2097:01:00:00	gmtoff	N	0
	09MAR2098:03:00:00	02NOV2098:01:00:00	gmtoff	N	0
	08MAR2099:03:00:00	01NOV2099:01:00:00	gmtoff	N	0
	14MAR2100:03:00:00	07NOV2100:01:00:00	gmtoff	N	0
O	.	.	gmtoff	N	-3600

Table 3. The Last Five Records of Data FMT

In the data step, I add a statement to adjust location with format GMTOFF. If the date time range does not fall into the DST effective time, it will roll back an hour:

```
localtime =localtime + input(put( localtime,gmtoff.), best5.);
```

Following table shows variable LOCALTIME reflects both EST and EDT time in appropriate schedule:

METER_NUMBER	READING_TIMESTAMP	LOCALTIME	READING_VALUE
xxxxx08	08MAR15:01:00:00.00 AM	07MAR15:08:00:00.00 PM	61489
xxxxx08	08MAR15:02:00:00.00 AM	07MAR15:09:00:00.00 PM	61493
xxxxx08	08MAR15:03:00:00.00 AM	07MAR15:10:00:00.00 PM	61495

METER_NUMBER	READING_TIMESTAMP	LOCALTIME	READING_VALUE
xxxxx08	08MAR15:04:00:00.00 AM	07MAR15:11:00:00.00 PM	61496
xxxxx08	08MAR15:05:00:00.00 AM	08MAR15:12:00:00.00 AM	61497
xxxxx08	08MAR15:06:00:00.00 AM	08MAR15:01:00:00.00 AM	61497
xxxxx08	08MAR15:07:00:00.00 AM	08MAR15:03:00:00.00 AM	61498
xxxxx08	08MAR15:08:00:00.00 AM	08MAR15:04:00:00.00 AM	61499
xxxxx08	08MAR15:09:00:00.00 AM	08MAR15:05:00:00.00 AM	61500
xxxxx08	08MAR15:10:00:00.00 AM	08MAR15:06:00:00.00 AM	61500

Table 4. The Final Results of Conversion after Adjustment from Daylight Saving Time in March

Figure 1 is an image from <http://www.worldtimebuddy.com/> about the comparison between GMT and Eastern Time on the second Sunday of March in 2015.



Figure 1. Comparison between GMT and Eastern Time on the Second Sunday of March in 2015

LOCALTIME will end DST on the first Sunday of November:

METER_NUMBER	READING_TIMESTAMP	LOCALTIME	READING_VALUE
xxxxx08	01NOV15:01:00:00.00 AM	31OCT15:09:00:00.00 PM	67751
xxxxx08	01NOV15:02:00:00.00 AM	31OCT15:10:00:00.00 PM	67752
xxxxx08	01NOV15:03:00:00.00 AM	31OCT15:11:00:00.00 PM	67752
xxxxx08	01NOV15:04:00:00.00 AM	01NOV15:12:00:00.00 AM	67753
xxxxx08	01NOV15:05:00:00.00 AM	01NOV15:01:00:00.00 AM	67753
xxxxx08	01NOV15:06:00:00.00 AM	01NOV15:01:00:00.00 AM	67753
xxxxx08	01NOV15:07:00:00.00 AM	01NOV15:02:00:00.00 AM	67754
xxxxx08	01NOV15:08:00:00.00 AM	01NOV15:03:00:00.00 AM	67754
xxxxx08	01NOV15:09:00:00.00 AM	01NOV15:04:00:00.00 AM	67755
xxxxx08	01NOV15:10:00:00.00 AM	01NOV15:05:00:00.00 AM	67755

Table 5. The Final Results of Conversion after Adjustment from Daylight Saving Time in November

Figure 2 is an image from <http://www.worldtimebuddy.com/> about the comparison between GMT and Eastern Time on the first Sunday of November in 2015.



Figure 2. Comparison between GMT and Eastern Time on the First Sunday of November in 2015

APPLICATION ON OTHER PLACES

There is an advantage to store daylight saving time as a dataset. Besides creating a format to simplify my coding, it can be used dynamically in other areas. For example, I need to identify when daylight saving time starts and ends to adjust meter readings for my analysis. The following syntax shows the program created macro variable MAR_DST and NOV_DST from data FMT which was used to create SAS format:

```
proc sql noprint;
select start format=best16. into: mar_dst
separated by ','
from fmt;
select end format=best16. into: nov_dst
separated by ','
from fmt;
quit;
```

There are two readings in hour one of the first Sunday in November after conversion. I need to calculate the mean of these two values:

```
proc sql;
create table novavg as
select meter_number, localtime, mean(kw) as kw
from nov
where localtime in (&nov_dst)
group by meter_number, localtime
order by meter_number, localtime;
quit;
```

← use macro variable NOV_DST

CONCLUSION

The final program was stored as a compiled macro catalog. It is shared across different divisions. To convert from GMT time to local time, the analysts just need to put in the dataset name and company ID number. It improves the accuracy and efficiency significantly.

REFERENCES

Rick Wicklin, "Spring forward, fall back: Using SAS to compute the onset of daylight saving time" The DO Loop, Statistical programming in SAS with an emphasis on SAS/IML programs. Available at

<http://blogs.sas.com/content/iml/2012/03/08/using-sas-to-compute-the-onset-of-daylight-saving-time.html>

"Why Electric Utilities Need Load Research". Available at

<http://aeic.org/wp-content/uploads/2013/07/LoadResearchPrimer.pdf>

SAS® 9.4 Language Reference: Concepts, Fourth Edition

<https://support.sas.com/documentation/cdl/en/lrcon/68089/HTML/default/viewer.htm#titlepage.htm>

ACKNOWLEDGMENTS

The following people contributed extensively to the development of this paper: Bob Bolen and Celia Wang at Southern Company and Sarah Tsai. Their support is greatly appreciated.

CONTACT INFORMATION

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

Chao-Ying Hsieh
Southern Company Services, Inc.
214 Ralph McGill BLVD
Atlanta, GA 30308
chsieh@southernco.com

SAS and all other SAS Institute Inc. product or service names are registered trademarks or trademarks of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are trademarks of their respective companies.