Paper 007-2010

# From Unfriendly Text File to SAS® Dataset – Reading Character Strings

Anastasiya Osborne, Farm Service Agency (USDA), Washington, DC

## ABSTRACT

What does it take to automatically clean unwieldy text files in SAS9? It takes business necessity and intermediate–to-advanced combinations of SAS9 options, statements, character and date functions, call routines, macros, and OS commands. This paper describes how to automate an otherwise tedious and error-prone process of extracting data from text files with inconsistent structure and format. The data is then used in a real-life project.

## INTRODUCTION

This paper discusses an algorithm developed to automate repetitive, unexciting, and error-prone steps to manually extract data from loosely structured and inconsistently formatted text files. The data is then used to update and verify a database of temporary changes in crop prices as input in a loan rate making process.

The SAS program developed to automate the process uses an array of techniques to extract temporary changes in crop prices from daily text files, correspond them with the file name and creation date, execute a general macro for each file in a program directory, and run a verification program.

The need for such automation resulted from a real-world project at the Farm Service Agency, USDA. The goal of the project is to establish annual county loan rates (for the crop year) for 18 major crops, as mandated by Congress. One of the steps, traditionally done manually, was to update and verify a database of temporary changes in crop prices (called temporary differentials), reported on-line as a daily text file, cluttered with other information.

The discussed SAS program:

- automatically reads text files from a shared drive, using options NOXWAIT and CALL SYSTEM for DOS commands;
- extracts comments with temporary changes in crop prices and takes only those that correspond to the date of the file creation;
- uses two %INCLUDE statements, one to run the macro RUNEM for all the text files saved in a program directory, and another to run the date verification program (to check for missing files);
- uses character functions (SCAN, INDEX, SUBSTR, TRANSLATE; LENGTH);
- uses date and time functions and formats (TODAY, ANYDTDTEw.);
- significantly saves project time and improves data quality.

To establish county loan rates, FSA needs to create a database of Posted County Price (PCPs) and Temporary Differentials for the two previous years. Such data is reported daily online on the USDA's Farm Service Agency's website. However, the data is the opposite of user-friendly and the format of reported data points can change without notice. The text file requires a lot of manual work to make it possible to enter the data into a database for further processing.

The changes to the process, discussed in this paper, made it possible to completely automate the process and dispose of any manual steps, such as data entry and verification by hand.

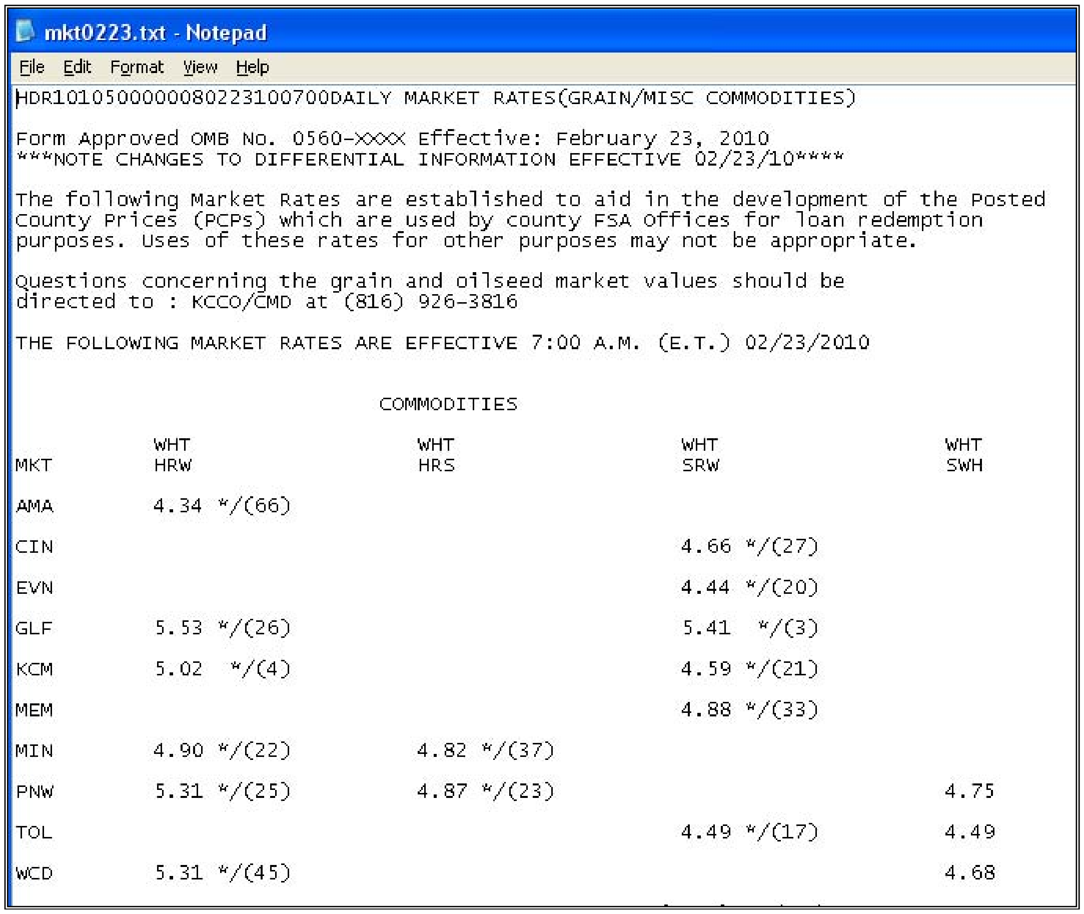
The process starts with downloading daily text files from the web. It is usually done once a month. Then the office-developed SAS program reads those files and creates several datasets with market rates sheets, temporary differentials, and verification reports.

## PROBLEMS WITH AUTOMATION

The previous official process included a lot of steps that were manual, incredibly time consuming, and vulnerable to error. Verification was also manual and thus very labor intensive. This was a hard way to obtain data for analysis. The biggest problem with the project was a step that, at the time, was considered impossible to automate – saving the text file published on the Farm Service Agency website to a shared drive, and **manually** inputting the section that reported temporary differentials (see Fig. 3). The data in the file was in such a computer-unfriendly and loose format that the only way to get this data for further processing was to type a few rows of data every day, and verify it two or three times. However, in the 14 years of this project, it became the worst and most tiresome step. The data other than temporary differentials was reported well enough for a programmer to read it automatically.

The group had an official notebook, like before the computerized age, where an analyst would write down the temporary differentials taken from the daily web file for that day, and then check the notes (twice!) whether there was a mistake. After I took over the project, many interesting SAS techniques were developed just by trying to eliminate the possibility of making mistakes and to make the process faster. After automating this step, I automatically verified the earlier temporary differentials data and found two errors. This provided me with yet more evidence that manual verification should not be relied on too much.

Below is an example of how the data usually looks (Fig.1). There were several problems with the data, as well as with the way it was processed previously. First, the old program would process one file only, without looping through a list of designated files, so for a 2-year period (494 work days when the files are produced) an analyst would have to run the program 494 times(!). Second, there was no verification whether any of the files were missing. Third, the dates were reported free-style, without regular format. This presented no difficulties to the human eye, but confused SAS (prior to SAS9). The fourth problem was with the nature of the temporary differentials (or price adjustments) - the price could be adjusted in one state, or several states, or all the states, and this was quite an obstacle to overcome – coming up with a mechanism for taking a not-fully-structured character string and dividing it into a proper number of SAS variables. A much lesser problem was the outdated units in the reporting system, although it still managed to give us grief at one point.



```
HDR1010500000080223100700DAILY MARKET RATES(GRAIN/MISC COMMODITIES)

Form Approved OMB No. 0560-XXXX Effective: February 23, 2010
***NOTE CHANGES TO DIFFERENTIAL INFORMATION EFFECTIVE 02/23/10****

The following Market Rates are established to aid in the development of the Posted
County Prices (PCPs) which are used by county FSA offices for loan redemption
purposes. Uses of these rates for other purposes may not be appropriate.

Questions concerning the grain and oilseed market values should be
directed to : KCCO/CMD at (816) 926-3816

THE FOLLOWING MARKET RATES ARE EFFECTIVE 7:00 A.M. (E.T.) 02/23/2010


                             COMMODITIES

           WHT                  WHT                  WHT                  WHT
MKT        HRW                  HRS                  SRW                  SWH

AMA        4.34 */(66)

CIN                                                  4.66 */(27)

EVN                                                  4.44 */(20)

GLF        5.53 */(26)                               5.41  */(3)

KCM        5.02  */(4)                               4.59 */(21)

MEM                                                  4.88 */(33)

MIN        4.90 */(22)          4.82 */(37)

PNW        5.31 */(25)          4.87 */(23)                               4.75

TOL                                                  4.49 */(17)          4.49

WCD        5.31 */(45)                                                    4.68
```

Fig 1. Average text file with posted county prices and temporary differentials, see"ftp://165.221.16.16/public/ratespub/default.htm"

## PROBLEM 1 – ONE DATE AT A TIME (NO MONTHLY OR ANNUAL LOOPS)

I took over this project in January 2007, and automated it from what was official and regular by that time. Some of the problems that came with reading the unfriendly text file were already solved by the previous ingenious programmer, and some were still there, waiting for me. The old program could read and create an output from all the numbers in the dataset, except temporary differentials. However, it would read one file at a time. The old program created a window for an analyst to type the month, date, and year separately, and the program (after checking whether the date was valid) would read the file and create posted county prices in a text file. If any portion of the date was invalid, there would be a warning, for example, "The day is incorrect!" Given that the loan rate cycle needs a 2-year period of reported posted county prices (494 work-day files), verifying it would be exceedingly labor-intensive. Also, the program could not tell whether any of the files were missing in a monthly folder, even if all the files already there had correct dates. An analyst was supposed to verify this manually.

I changed the structure of the files and saved in one folder all 494 files needed for establishing loan rates in one loan year. It worked very well to have one dataset with all the needed data, but took a long time to run. A year later I started making monthly folders, to cut down the remote computer's processing time, and this is the method used in the examples. The pre-2007 official program was modified drastically to read all the files in a folder, create a list of those files, read their names (the official file name was unconventional "mm/dd/y.txt") and create a macro to execute the files one at a time.

The files are located on the FSA public website, "ftp://165.221.16.16/public/ratespub/default.htm". For example, the website says "0101.txt" for a market sheet file from January 1. The year is not specified yet. After 2-3 weeks the file name becomes "090101.txt" if this is the data for 2009. Since the pattern is not standard, the analyst downloads the files and renames them into the comparable file names. Unfortunately, the official standard way was to just put the last digit of the year (for example, January 2009 is "01019.txt), not the last 2 or, better still, 4 digits, and it took me some time to figure out how to read those file names and reconstruct the date so that SAS would understand it.

The program starts like this, after the verbose commented-out text with the program name, date, input and output file locations, and history of the process.

```
/* First, we assign macro */
%LET as_of_today = February 3, 2010; /* to make unique files, as the name goes into
                                        the output file name */
%LET mymonth = January; /*usually the previous month, as we need to wait until the end
                     of the month to process the data*/
%LET myyear = 2010;
%LET RATESOUT = "S:\EPAS\MKTRATES\For verification\newMRADD &as_of_today..txt" ;
                       /* this was a requirement – to still produce the text file to
                       compare with the old process*/

/* What month are we processing? CHANGE HERE!*/
%LET startmonth = '01JAN2010'd ;
%LET endmonth   = '31JAN2010'd ;

/*NOTE: We now read files from monthly folders.
  NOTE: DO NOT MAKE SPACES IN THE FOLDER NAME WHEN DOWNLOADING FILES FROM THE WEB! THE
  PROGRAM WILL NOT RUN! IT SHOULD LOOK LIKE: JANUARY2010 */
%LET rawdir1 = s:\epas\mktrates\&mymonth.&myyear ;
/* Where all the SAS programs are located. */
%LET progdir = S:\EPAS\MKTRATES\SAS programs for checking data ;
```

Here is the step that was needed to make a list of files in the folder.

```
OPTIONS NOXWAIT;
RUN; /* without this option the DOS window will stay open ...*/

DATA A ; /*(It could be _null_, but I wanted to see what was going on. */
   command="dir &rawdir1\*.txt /b > &rawdir1\files.txt";
   call system(command);
RUN; quit;
/* This is how data A looks after PROC PRINT:
dir s:\epas\mktrates\January2010\*.txt /b > s:\epas\mktrates\January2010\files.txt */
```

Another way to accomplish the same task would be to use the PIPE engine.

```
filename q pipe 'dir "s:\epas\mktrates\January2010" /b' ;
```

```
DATA FILES (keep = filename);
      length filename $9 ext $3;
      infile "&rawdir1\files.txt" delimiter= '.' ;
      input filename ext;
RUN;
/* Here is the part of PROC PRINT of the dataset FILES:
                                          1    files
                                          2    mkt01040
                                          3    mkt01050
                                          …
                                         21    mkt01290 */
```

We found out the number of daily files in the folder. Now the question arises – is this the right number?

## PROBLEM 2 – HOW MANY FILES ARE ENOUGH?

The program reads the entire folder (a month worth of data), creates a dataset with market rate sheets, a dataset with temporary differentials, and also creates two verification reports – are there dates that are extraneous or missing? Are there extraneous or missing data points, even if there is the correct number of daily files? Interestingly, in January 2009 the automatic verification report showed that one daily file was missing on the FSA website. After my office reported this problem to the Farm Service Agency personnel in Kansas City, they eventually found the missing file and put it on-line. This was the first time a file had been lost in many years. It happened again in July 2009.

This is the part where automatic verification is in order. We need to compare the files that are in the folder to the files that should have been there. How do we know whether the number of the files is correct? The market rates are published once a day during work days, but not on federal holidays. I took the list of federal holidays for the last few years and for the next few years out of the U.S. Office of Personnel Management's website "http://www.opm.gov/Operating_Status_Schedules/fedhol/2010.asp", created a permanent dataset (a reference table), and checked the list of files in monthly folders with the list of federal holidays. The verification module registers which dates are missing from the master list of every calendar day in a period. Now, it's okay to have missing data for weekends, but then the program checks the list of federal holidays. If the missing date is not on the list of federal holidays and it was not Saturday or Sunday (those three cases are legitimate reasons to be missing a file), the verification program creates a report.

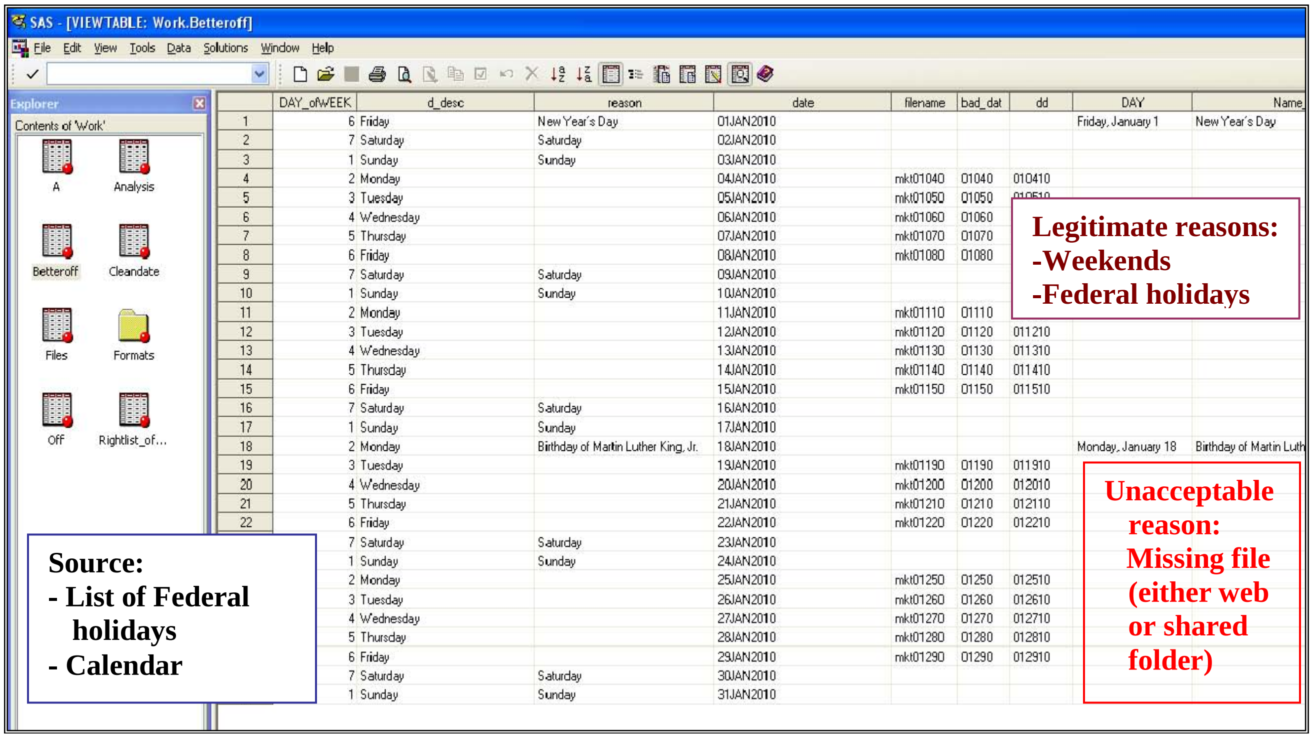
In other words: the program compiles the list of files in a month, and compares them to a list of calendar days in that month, and also to the list of federal holidays, and reports whether the reason for missing files was valid (holidays, weekend), or the files were missing for non-legitimate reasons (analyst forgot to download the file, or the Farm Service Agency forgot to put that file on the web – we are all human.)

```
DATA RIGHTLIST_OFDATES (rename = (date = date_see));
   date =  &startmonth. ; /* Change - monthly */
       do while( date <= &endmonth. );
         output;
          date = intnx( 'day', date, 1 );
       end;
       keep date ;
       format date date9. ;
RUN; /* The data set WORK.RIGHTLIST_OFDATES has 31 observations and 1 variable
(from 01JAN2010 to 31JAN2010). */

/* make a match between the perfect list of calendar days and the reported data */;
DATA CLEANDATE ;
   set files ;
   if substr(filename, 1, 1) ^= "m" then delete; /* take out one bad observation */
   bad_date = substr(filename, 4, 5) ; /* official date */
   dd = substr(bad_date, 1, 4)||"1"||substr(bad_date, 5, 1); /* The third symbol of
the year became "1" instead of "0" in January 2010 to signify a new decade. The year
YY0Y became XX1Y. This is currently hard-coded, but Howard Schreier proposed a cut-off
solution. */
   date_see = input(dd, ANYDTDTE12.) ; /* ANYDTDTEw. reads multiple date layouts. */
   format date_see date9. ;
RUN ;
```

Below is the partial print-out of the dataset CLEANDATE. We can see the conventional date now.

```
/*
                       Obs    filename    bad_date      dd       date_see

                         1    mkt01040     01040      010400    04JAN2010
                         2    mkt01050     01050      010500    05JAN2010
                         3    mkt01060     01060      010600    06JAN2010
                         …
                        20    mkt01290     01290      012900    29JAN2010 */


PROC SORT data = CLEANDATE ;
   by date_see ;
RUN ;

DATA ANALYSIS (rename = (date_see = date )) ;
   merge  rightlist_ofdates
          cleandate
          ;
   by date_see ;
RUN ;

DATA OFF ;
   merge analysis (in = a)
   mr.Fedholidays2003_2010 (in = b)
    ;
      by date ;
      if a ;
       if date = . or date < '01jun2008'd then delete ; /* This can be replaced with a
       macro variable. In earlier versions, all two previous years of data would be
       processed in June of every year. */
RUN ;

ODS listing ;
PROC CONTENTS data = ANALYSIS ;
RUN ;

PROC CONTENTS data = MR.FEDHOLIDAYS2003_2010 ;
RUN ;

/* Question: How would I know then is Saturday or Sunday? */
PROC FORMAT ;
   value DWKf
       1 = "Sunday"
       2 = "Monday"
       3 = "Tuesday"
       4 = "Wednesday"
       5 = "Thursday"
       6 = "Friday"
       7 = "Saturday"
      ;
RUN ;

DATA BETTEROFF ;
   length reason $40. ;
   set off ;

   DAY_ofWEEK = weekday(date) ; /*day of the week as numbers, 1 is Sunday */
   d_desc =  DAY_ofWEEK ;

   if filename = " " then
   do ;
```

```
        if day ^= " " then reason = name_XH ;
        else if day = " "  and DAY_ofWEEK = 1 then reason = "Sunday" ;
        else if day = " "  and DAY_ofWEEK = 7 then reason = "Saturday" ;
        else /*reason = " " */;
        end ;
    format d_desc DWKf. ; /* To see familiar representation. */
RUN ;
```


SAS - [VIEWTABLE: Work.Betteroff]
File Edit View Tools Data Solutions Window Help
Explorer
Contents of 'Work'
A
Analysis
Betteroff
Cleandate
Files
Formats
Off
Rightlist_of...
Legitimate reasons:
-Weekends
-Federal holidays
Unacceptable reason: Missing file (either web or shared folder)
Source:
- List of Federal holidays
- Calendar

| | DAY_ofWEEK | d_desc | reason | date | filename | bad_dat | dd | DAY | Name_ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 6 | Friday | New Year's Day | 01JAN2010 | | | | Friday, January 1 | New Year's Day |
| 2 | 7 | Saturday | Saturday | 02JAN2010 | | | | | |
| 3 | 1 | Sunday | Sunday | 03JAN2010 | | | | | |
| 4 | 2 | Monday | | 04JAN2010 | mkt01040 | 01040 | 010410 | | |
| 5 | 3 | Tuesday | | 05JAN2010 | mkt01050 | 01050 | 010510 | | |
| 6 | 4 | Wednesday | | 06JAN2010 | mkt01060 | 01060 | [illegible] | | |
| 7 | 5 | Thursday | | 07JAN2010 | mkt01070 | 01070 | [illegible] | | |
| 8 | 6 | Friday | | 08JAN2010 | mkt01080 | 01080 | [illegible] | | |
| 9 | 7 | Saturday | Saturday | 09JAN2010 | | | | | |
| 10 | 1 | Sunday | Sunday | 10JAN2010 | | | | | |
| 11 | 2 | Monday | | 11JAN2010 | mkt01110 | 01110 | [illegible] | | |
| 12 | 3 | Tuesday | | 12JAN2010 | mkt01120 | 01120 | 011210 | | |
| 13 | 4 | Wednesday | | 13JAN2010 | mkt01130 | 01130 | 011310 | | |
| 14 | 5 | Thursday | | 14JAN2010 | mkt01140 | 01140 | 011410 | | |
| 15 | 6 | Friday | | 15JAN2010 | mkt01150 | 01150 | 011510 | | |
| 16 | 7 | Saturday | Saturday | 16JAN2010 | | | | | |
| 17 | 1 | Sunday | Sunday | 17JAN2010 | | | | | |
| 18 | 2 | Monday | Birthday of Martin Luther King, Jr. | 18JAN2010 | | | | Monday, January 18 | Birthday of Martin Luth |
| 19 | 3 | Tuesday | | 19JAN2010 | mkt01190 | 01190 | 011910 | | |
| 20 | 4 | Wednesday | | 20JAN2010 | mkt01200 | 01200 | 012010 | | |
| 21 | 5 | Thursday | | 21JAN2010 | mkt01210 | 01210 | 012110 | | |
| 22 | 6 | Friday | | 22JAN2010 | mkt01220 | 01220 | 012210 | | |
| | 7 | Saturday | Saturday | 23JAN2010 | | | | | |
| | 1 | Sunday | Sunday | 24JAN2010 | | | | | |
| | 2 | Monday | | 25JAN2010 | mkt01250 | 01250 | 012510 | | |
| | 3 | Tuesday | | 26JAN2010 | mkt01260 | 01260 | 012610 | | |
| | 4 | Wednesday | | 27JAN2010 | mkt01270 | 01270 | 012710 | | |
| | 5 | Thursday | | 28JAN2010 | mkt01280 | 01280 | 012810 | | |
| | 6 | Friday | | 29JAN2010 | mkt01290 | 01290 | 012910 | | |
| | 7 | Saturday | Saturday | 30JAN2010 | | | | | |
| | 1 | Sunday | Sunday | 31JAN2010 | | | | | |

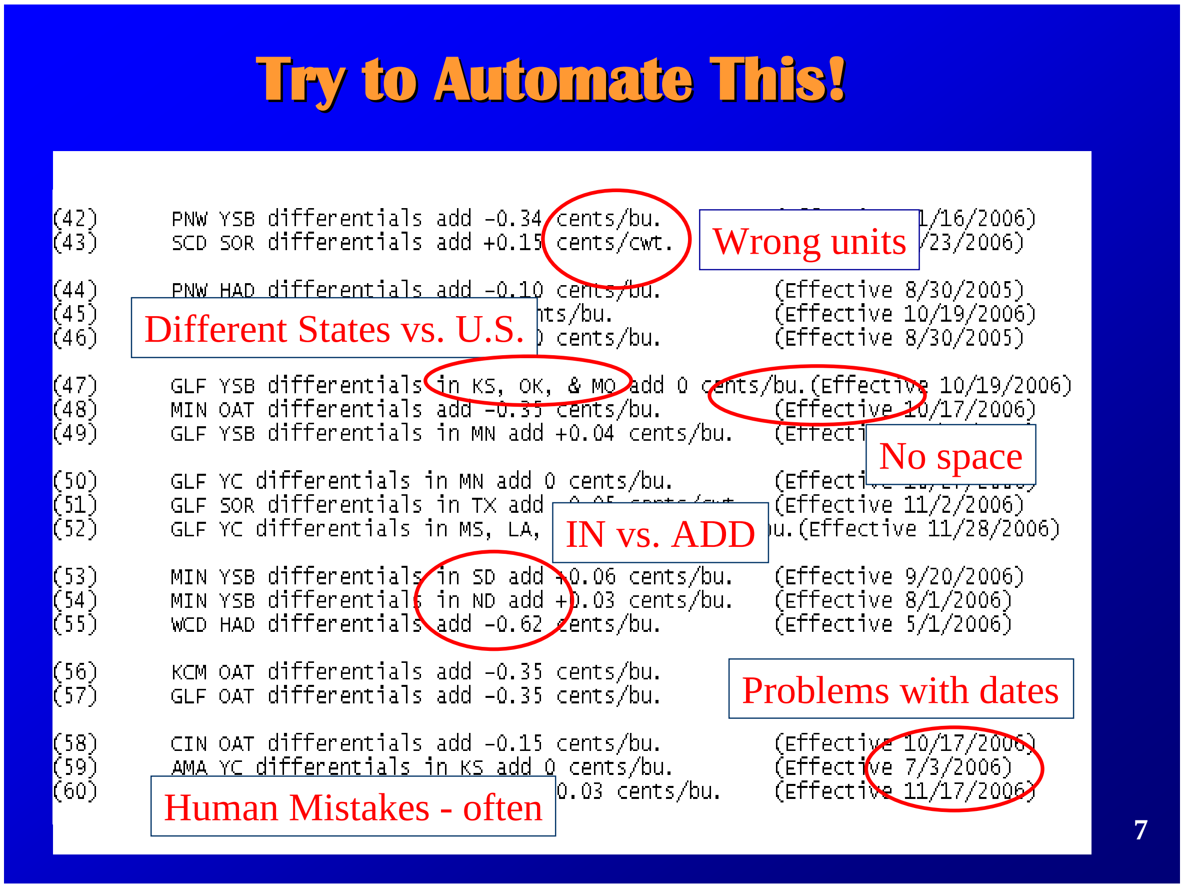
Fig 2. Explanation for each of the missing file.

```
/* Check for the reasons of non-reports, if there are any. */
DATA BAD ;
   set betteroff ;
   if filename = " " and reason = " " ;
RUN ;
/*end of checking for date validity */
```

Now we will discuss in detail the problem with reporting dates and the function ANYDTDTEw. (already shown in action on the bottom of p.4.)

## PROBLEM 3 – VARIATIONS IN REPORTING DATES

One of the big problems with automatically reading temporary differentials was the unexplainably inconsistent formats of reported dates (see Fig. 3. on next page.) For example, August 1, 2006 could be represented as 8/1/2006, 8/1/2006, 08/01/2006, 8/01/06 and so on.  The MMDDYY informat would handle all of the variations shown, but not YYYYMMDD, as in ‘20060801’. This problem with date formats became possible to resolve easily only with SAS9, with the new date function ANYDTDTEw. that was designed to deal with such variations in reporting dates.  Applying ANYDTDTEw. to unwieldy files was the first step to successfully automating the process. The other problems remained.



## Try to Automate This!

```
(42)      PNW YSB differentials add -0.34 cents/bu.                 1/16/2006)
(43)      SCD SOR differentials add +0.15 cents/cwt.                /23/2006)

(44)      PNW HAD differentials add -0.10 cents/bu.          (Effective 8/30/2005)
(45)                                          ts/bu.         (Effective 10/19/2006)
(46)                                          ) cents/bu.    (Effective 8/30/2005)

(47)      GLF YSB differentials in KS, OK, & MO add 0 cents/bu.(Effective 10/19/2006)
(48)      MIN OAT differentials add -0.35 cents/bu.          (Effective 10/17/2006)
(49)      GLF YSB differentials in MN add +0.04 cents/bu.    (Effecti

(50)      GLF YC differentials in MN add 0 cents/bu.         (Effecti
(51)      GLF SOR differentials in TX add                    (Effective 11/2/2006)
(52)      GLF YC differentials in MS, LA,                    u.(Effective 11/28/2006)

(53)      MIN YSB differentials in SD add +0.06 cents/bu.    (Effective 9/20/2006)
(54)      MIN YSB differentials in ND add +0.03 cents/bu.    (Effective 8/1/2006)
(55)      WCD HAD differentials add -0.62 cents/bu.          (Effective 5/1/2006)

(56)      KCM OAT differentials add -0.35 cents/bu.
(57)      GLF OAT differentials add -0.35 cents/bu.

(58)      CIN OAT differentials add -0.15 cents/bu.          (Effective 10/17/2006)
(59)      AMA YC differentials in KS add 0 cents/bu.         (Effective 7/3/2006)
(60)                                     0.03 cents/bu.      (Effective 11/17/2006)
```

Fig 3. Problems with automation of reading a SAS-unfriendly text file into a database.

Let's continue with the SAS program. We need to have a code that will automatically run the macro `%RUNEM` for each of the daily files in the monthly folder. That macro will clean all the extraneous text from the daily file and come out with two SAS datasets, one for market rate sheets (official module, already in production before 2007), and one for temporary differentials (new module, that was deemed impossible to make.)

```
data m_call;
    length codeline $50. olddate1 $6.  ;
     set files;
     if filename = "files" then delete ;
      file "&PROGDIR\m_call for temp differentials.sas";
          olddate1 = substr(filename, 4, 5);
      codeline = '%runem('||'olddate = ' ||olddate1 ||")" ;
   put codeline;
run;

/* This is how the “m_call” looks.

                        Obs            codeline              olddate1    filename

                          1    %runem(olddate = 01040 )      01040      mkt01040
                          2    %runem(olddate = 01050 )      01050      mkt01050
                          3    %runem(olddate = 01060 )      01060      mkt01060
                        …
                         20    %runem(olddate = 01290 )      01290      mkt01290 */
```

Now is the time to show the code for processing of daily files.

```
filename ratesout "S:\EPAS\MKTRATES\For verification\newMRADD &as_of_today. .txt" ;

 %macro RUNEM (olddate = );
/* GOAL: read market rate sheets and extract info about changes */
/* in temporary differentials. This part was missing from J.C.  */
/* program. IDEA: extract comments and take only those          */
/* that correspond to the date of the market rate file. Create  */
/* a file with all the temporary changes and compare to the file*/
/* made by manual input.                                        */

filename in  "S:\EPAS\MKTRATES\&mymonth.&myyear.\mkt&olddate..txt" ;
/* NOTE: monthly change */
```

## PROBLEM 4 – DIFFERENT LENGTH OF TEXT (SEVERAL STATES VS. ALL U.S.)

Another problem with market sheet files was that temporary differentials could change for all of the United States, in one state only, or in several states. There was no standardization or expectation of what would be written on each row, except that the data would start after the word "**in**" and end before the word "**add**". For example, see below what variations were reported on January 04, 2007:

(50) GLF YC differentials in MN add 0 cents/bu. (Effective 10/27/2006)

(51) GLF SOR differentials in TX add -0.05 cents/cwt. (Effective 11/2/2006)

(52) GLF YC differentials in MS, LA, **& AL** add 0 cents/bu.(Effective 11/28/2006)

(74) GLF YSB differentials in LA, AL, **MS** add 0 cents/bu.(Effective 10/27/2006)

(75) TKO YSB differentials in TN add 0 cents/bu. (Effective 10/27/2006)

You can see that sometimes there is one state where the temporary differentials were reported, sometimes there are several states. Also, the last state can come after a comma or an ampersand, with or without a space. Sometimes the data in the text file would be missing, or read as "N/B". The data can be changed the next day after reporting, or almost a month later, and then the file's name changes to include "c' – for corrected.

If you look at comment 52, you will notice that there is no separation between the data units and the left parenthesis. So, the data is not lined up in columns. How would we read such a file?

This is how we dealt with temporary differentials in general, including problem 4.

```
/*********************************************************************************/
                    /* TEMPORARY DIFFERENTIALS MODULE */
/*********************************************************************************/
data look  ;
   infile in length=reclen;
   input @;  /* Take everything from the text file */
   input @1 lstring $varying200. reclen; /* Take any text row, including blank ones */
   lstrip=trim(left(lstring));           /* Take all the spaces out on the left */

   if substr(lstrip, 1, 1) = "("
             and substr(lstrip, 2, 1) in ('1', '2', '3', '4', '5','6', '7', '8', '9')
   then output look ; /* This is how the strings with temporary differentials start*/
run ;

data cleanit (drop = lstring ind intermediate loc1 change1 date1 date2 date3 date4 dd
                     dd1);
       length report_date 8. comment $3. market $3. grain  $3. word  $3.
              dd $6. dd1 $6. ;
       set look ;

              comment = scan (lstrip, 1) ;
             market = scan (lstrip, 2) ;
             grain = scan (lstrip, 3) ;
```

```
                word =  scan (lstrip, 5) ;
                 ind = INDEX(lstrip,' ');  /* position in string */
                    /* Use INDEX to search for 'grouped letters' anywhere in a string or
                    to search for individual letters */

                intermediate = substr(lstrip, INDEX(lstrip, "add")) ;
                loc1 = substr(intermediate, 4, 6) ;
                change1 = trim(translate(loc1,' ','cen')) ;
                change = input(change1, comma7.2) ;

                date1 = SUBSTR(lstrip, INDEX(lstrip,'e '));
                date2 = substr(date1, 2 ) ; /*The best approximation */
                date3 = translate(date2,' ',')') ; /*Which says, "variable lstrip should
                    be the incoming value with the right parenthesis changed to a
                    space.*/
                date4 = trim(date3) ;
                date_ch = input(date4, ANYDTDTE12.) ; /* It's possible!
                                               ANYDTDTEw. can read multiple date layouts. */
                dd = substr("&olddate", 1, 4)||"0"||substr("&olddate", 5, 1);
                dd1 = input(dd, ANYDTDTE12.) ;
                report_date = input(dd1, 8. ) ;
                format  date_ch   mmddyy10. report_date mmddyy10. ;

                call symputx("normal_date", put(report_date, 8.));

        run ;

        /* Big problem – how to read different states, and rarely happening state 4. */
        data extract_info ;
           length states $20. state1 $2. state2 $3. state3rough $7. state3try $7.
                 state3 $2. state4 $2.;
           set cleanit ;

           if word = "in" then
              do ;
                 states = trim(substr(lstrip, INDEX(lstrip, "in")));
                 state1 = substr(states, 4, 2);
                 state2rough = trim(substr(states, 7, 3));
                 state2 = translate(state2rough,' ','add');

                 state3rough = substr(states, 11, 7);
                 state3try = trim(state3rough) ;
                 try1 = trim(substr(state3try, 2,  INDEX(lstrip, " "))) ;
                 try2 = trim(left(translate(try1,' ','&')));
                 try3 = scan(try2, 1) ;
                 if try3 in ('AL', 'AK', 'AS', 'AZ', 'AR', 'CA', 'CO', 'CT', 'DE',
'DC', 'FL','GA', 'GU', 'HI', 'ID', 'IL', 'IN', 'IA',  'KS', 'KY', 'LA', 'ME', 'MD', 'MH', 'MA',
'MI', 'MN', MS', 'MO', 'MT', 'NE', 'NV', 'NH', 'NJ', 'NM', 'NY','NC', 'ND', 'OH', 'OK', 'OR',
'PA', 'PR', 'RI', 'SC', 'SD', 'TN', 'TX', 'UT', 'VT', 'VA', 'WA', 'WV', 'WI', 'WY')
                then state3 = try3 ;
                else state3 = ' ';

                                        /* the hardest state 4 */

                try4 = scan(lstrip, 10) ;
                try4close = trim(left(translate(try4,' ','and')));
                if try4close in ('AL', 'AK', 'AS', 'AZ', 'AR', 'CA', 'CO', 'CT', 'DE',
         'DC', 'FL','GA', 'GU', 'HI', 'ID', 'IL', 'IN', 'IA',  'KS', 'KY', 'LA', 'ME', 'MD',
'MH', 'MA', 'MI', 'MN', MS', 'MO', 'MT', 'NE', 'NV', 'NH', 'NJ', 'NM', 'NY','NC', 'ND', 'OH',
'OK', 'OR', 'PA', 'PR', 'RI', 'SC', 'SD', 'TN', 'TX', 'UT', 'VT', 'VA', 'WA', 'WV', 'WI', 'WY')
                then state4 = try4close ;
                else state4 = ' ';
                end;
                if word = "add" then state = 'US' ;
```

```
             unit1 = substr(lstrip,  INDEX(lstrip, "cents")) ;
             unit = SUBSTR(unit1,1,INDEX(unit1,'.'));
run ;

/* Clean file */
proc sql;
      create table differentials as
      select  report_date
                    , lstrip
                    , market
                    , grain
                    , word
                    , state
                    , state1
                    , state2
                    , state3
                    , state4
                    , change
                    , date_ch


      from extract_info
             order by date_ch
             ;
quit;

/* Take only those temporary differentials that were made on a report date
of market rates. */
data  diff;
       set differentials ;
       if date_ch = report_date  ;
run ;

/* Keep all the differentials in a permanent database. */
proc datasets;
      append base = ad.temp_differentials data = diff ;
run ;
quit ;

/* The portion of the code that starts here in the SAS program is omitted because it
was an official program to deal with market rates only. I took it as a "MARKET SHEET
MODULE." */

%mend RUNEM ;

%include  "S:\EPAS\MKTRATES\SAS programs for checking data\m_call for temp
differentials.sas" ;
```

Another way to solve the problem with reading the affected states into a variable was proposed by Ian Whitlock in August 2009. It uses the **COMPBL** function that removes multiple blanks in a character string by translating each occurrence of two or more consecutive blanks into a single blank.

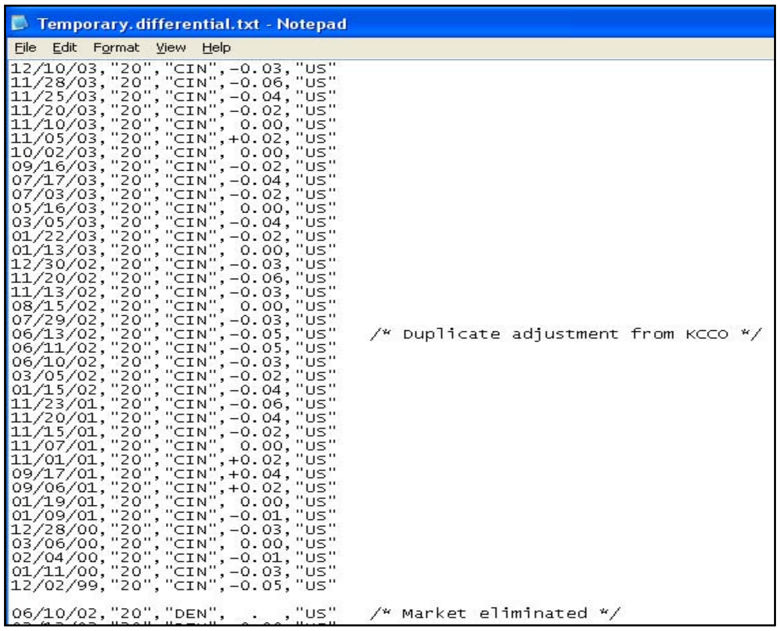
```
data Ian_idea ;
  length line $ 100 statelist $ 20 ;
  input line $char100. ;
  if line =: "(" and index("123456789", substr(line,2,1)) ;

  statelist = substr ( line, index(line, "differentials in") + 17) ;
  statelist = substr ( statelist, 1, index(statelist, "add") - 1) ;
  statelist = compbl ( translate ( statelist , "  " , ",&")) ;
  put statelist= ;
```

```
cards ;
(50)      GLF YC differentials in MN add 0 cents/bu.        (Effective 10/27/2006)
(51)      GLF SOR differentials in TX add -0.05 cents/cwt.  (Effective 11/2/2006)
(52)      GLF YC differentials in MS, LA, & AL add 0 cents/bu.(Effective 11/28/2006)
(74)      GLF YSB differentials in LA, AL, MS add 0 cents/bu.(Effective 10/27/2006)
(75)      TKO YSB differentials in TN add 0 cents/bu.      (Effective 10/27/2006)
;
run ;
/* From the log:
   statelist=MN
   statelist=TX
   statelist=MS LA AL
   statelist=LA AL MS
   statelist=TN */
```

One last word about other, less important problems, mentioned in Fig. 3. We had to deal with the wrong units too. The reporting system was so old that it kept saying “cents/bu” or “cents/cwt”, even though the data is reported in dollars. For example, in October 2007, a temporary differential was reported as “-1”, and my program read it as $0.01, because that’s what the official unit is (cent/bu). However, after checking with Kansas City, we had to change it to $1.00.  It has been such a ride to work with this reporting system! Luckily, the Farm Service Agency is reengineering the database software for the county loan program. Eventually, we expect to be able to query a database in Kansas City to get this data, not read daily text and compile a database from them.

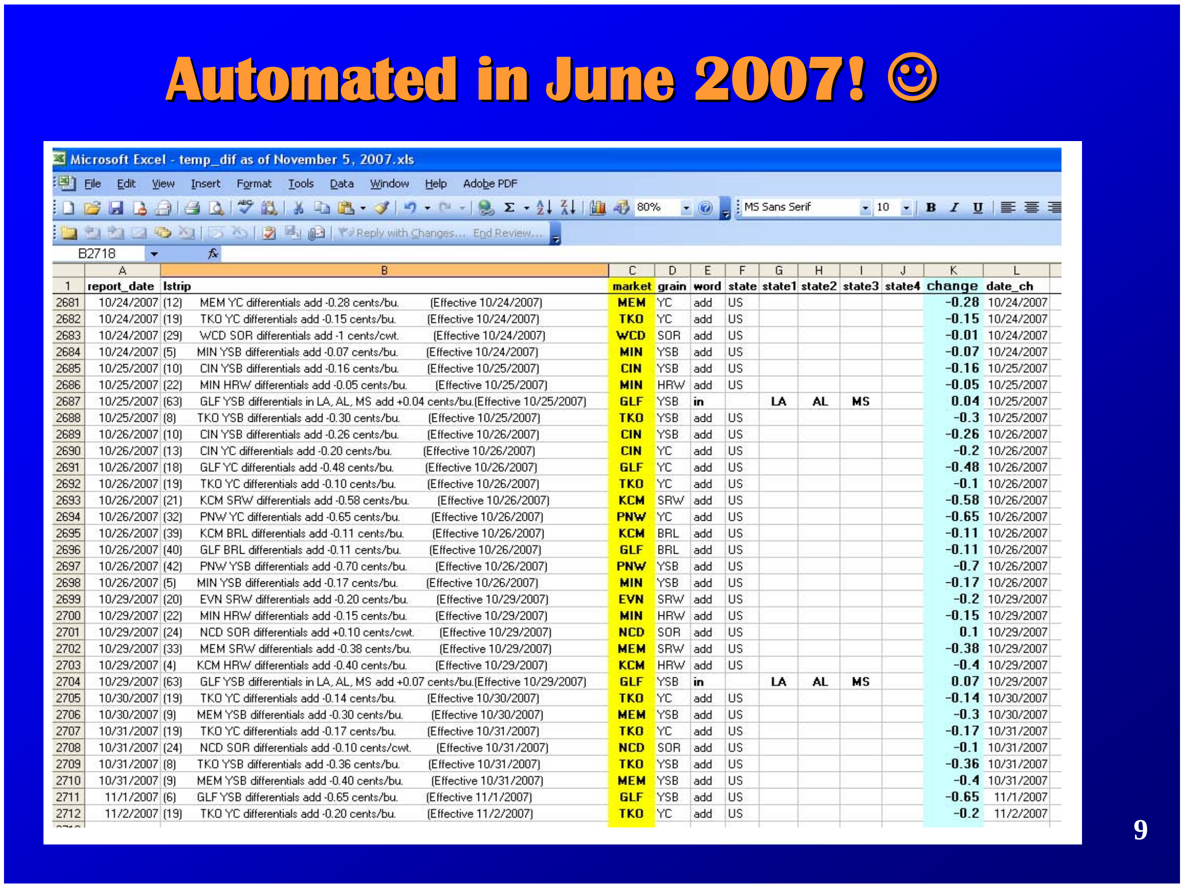
I am sure that Perl users would solve these string-searching problems in a more elegant-looking way, and I have every intention to learn how to do it, but this project modification took place in 2007, in a time crunch during production cycle, and I am proud the date-related problems were solved successfully and the seemingly impossible process automation was achieved.

Fig. 4 on the next page gives you an idea how the text file with temporary differentials looked like in the early days of the project, and Fig 5 shows the Excel version of a permanent SAS dataset.


Temporary.differential.txt - Notepad
File Edit Format View Help

```
12/10/03,"20","CIN",-0.03,"US"
11/28/03,"20","CIN",-0.06,"US"
11/25/03,"20","CIN",-0.04,"US"
11/20/03,"20","CIN",-0.02,"US"
11/10/03,"20","CIN", 0.00,"US"
11/05/03,"20","CIN",+0.02,"US"
10/02/03,"20","CIN", 0.00,"US"
09/16/03,"20","CIN",-0.02,"US"
07/17/03,"20","CIN",-0.04,"US"
07/03/03,"20","CIN",-0.02,"US"
05/16/03,"20","CIN", 0.00,"US"
03/05/03,"20","CIN",-0.04,"US"
01/22/03,"20","CIN",-0.02,"US"
01/13/03,"20","CIN", 0.00,"US"
12/30/02,"20","CIN",-0.03,"US"
11/20/02,"20","CIN",-0.06,"US"
11/13/02,"20","CIN",-0.03,"US"
08/15/02,"20","CIN", 0.00,"US"
07/29/02,"20","CIN",-0.03,"US"
06/13/02,"20","CIN",-0.05,"US"   /* Duplicate adjustment from KCCO */
06/11/02,"20","CIN",-0.05,"US"
06/10/02,"20","CIN",-0.03,"US"
03/05/02,"20","CIN",-0.02,"US"
01/15/02,"20","CIN",-0.04,"US"
11/23/01,"20","CIN",-0.06,"US"
11/20/01,"20","CIN",-0.04,"US"
11/15/01,"20","CIN",-0.02,"US"
11/07/01,"20","CIN", 0.00,"US"
11/01/01,"20","CIN",+0.02,"US"
09/17/01,"20","CIN",+0.04,"US"
09/06/01,"20","CIN",+0.02,"US"
01/19/01,"20","CIN", 0.00,"US"
01/09/01,"20","CIN",-0.01,"US"
12/28/00,"20","CIN",-0.03,"US"
03/06/00,"20","CIN", 0.00,"US"
02/04/00,"20","CIN",-0.01,"US"
01/11/00,"20","CIN",-0.03,"US"
12/02/99,"20","CIN",-0.05,"US"

06/10/02,"20","DEN",  .  ,"US"   /* Market eliminated */
```

Fig.4. Official file with temporary differentials until January 2007 – created by manual data entry.



## Automated in June 2007! ☺

Microsoft Excel - temp_dif as of November 5, 2007.xls
File Edit View Insert Format Tools Data Window Help Adobe PDF
80%
MS Sans Serif
10
Reply with Changes... End Review...
B2718

| | A | B | C | D | E | F | G | H | I | J | K | L |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | report_date | lstrip | market | grain | word | state | state1 | state2 | state3 | state4 | change | date_ch |
| 2681 | 10/24/2007 | (12) MEM YC differentials add -0.28 cents/bu. (Effective 10/24/2007) | MEM | YC | add | US | | | | | -0.28 | 10/24/2007 |
| 2682 | 10/24/2007 | (19) TKO YC differentials add -0.15 cents/bu. (Effective 10/24/2007) | TKO | YC | add | US | | | | | -0.15 | 10/24/2007 |
| 2683 | 10/24/2007 | (29) WCD SOR differentials add -1 cents/cwt. (Effective 10/24/2007) | WCD | SOR | add | US | | | | | -0.01 | 10/24/2007 |
| 2684 | 10/24/2007 | (5) MIN YSB differentials add -0.07 cents/bu. (Effective 10/24/2007) | MIN | YSB | add | US | | | | | -0.07 | 10/24/2007 |
| 2685 | 10/25/2007 | (10) CIN YSB differentials add -0.16 cents/bu. (Effective 10/25/2007) | CIN | YSB | add | US | | | | | -0.16 | 10/25/2007 |
| 2686 | 10/25/2007 | (22) MIN HRW differentials add -0.05 cents/bu. (Effective 10/25/2007) | MIN | HRW | add | US | | | | | -0.05 | 10/25/2007 |
| 2687 | 10/25/2007 | (63) GLF YSB differentials in LA, AL, MS add +0.04 cents/bu.(Effective 10/25/2007) | GLF | YSB | in | | LA | AL | MS | | 0.04 | 10/25/2007 |
| 2688 | 10/25/2007 | (8) TKO YSB differentials add -0.30 cents/bu. (Effective 10/25/2007) | TKO | YSB | add | US | | | | | -0.3 | 10/25/2007 |
| 2689 | 10/26/2007 | (10) CIN YSB differentials add -0.26 cents/bu. (Effective 10/26/2007) | CIN | YSB | add | US | | | | | -0.26 | 10/26/2007 |
| 2690 | 10/26/2007 | (13) CIN YC differentials add -0.20 cents/bu. (Effective 10/26/2007) | CIN | YC | add | US | | | | | -0.2 | 10/26/2007 |
| 2691 | 10/26/2007 | (18) GLF YC differentials add -0.48 cents/bu. (Effective 10/26/2007) | GLF | YC | add | US | | | | | -0.48 | 10/26/2007 |
| 2692 | 10/26/2007 | (19) TKO YC differentials add -0.10 cents/bu. (Effective 10/26/2007) | TKO | YC | add | US | | | | | -0.1 | 10/26/2007 |
| 2693 | 10/26/2007 | (21) KCM SRW differentials add -0.58 cents/bu. (Effective 10/26/2007) | KCM | SRW | add | US | | | | | -0.58 | 10/26/2007 |
| 2694 | 10/26/2007 | (32) PNW YC differentials add -0.65 cents/bu. (Effective 10/26/2007) | PNW | YC | add | US | | | | | -0.65 | 10/26/2007 |
| 2695 | 10/26/2007 | (39) KCM BRL differentials add -0.11 cents/bu. (Effective 10/26/2007) | KCM | BRL | add | US | | | | | -0.11 | 10/26/2007 |
| 2696 | 10/26/2007 | (40) GLF BRL differentials add -0.11 cents/bu. (Effective 10/26/2007) | GLF | BRL | add | US | | | | | -0.11 | 10/26/2007 |
| 2697 | 10/26/2007 | (42) PNW YSB differentials add -0.70 cents/bu. (Effective 10/26/2007) | PNW | YSB | add | US | | | | | -0.7 | 10/26/2007 |
| 2698 | 10/26/2007 | (5) MIN YSB differentials add -0.17 cents/bu. (Effective 10/26/2007) | MIN | YSB | add | US | | | | | -0.17 | 10/26/2007 |
| 2699 | 10/29/2007 | (20) EVN SRW differentials add -0.20 cents/bu. (Effective 10/29/2007) | EVN | SRW | add | US | | | | | -0.2 | 10/29/2007 |
| 2700 | 10/29/2007 | (22) MIN HRW differentials add -0.15 cents/bu. (Effective 10/29/2007) | MIN | HRW | add | US | | | | | -0.15 | 10/29/2007 |
| 2701 | 10/29/2007 | (24) NCD SOR differentials add +0.10 cents/cwt. (Effective 10/29/2007) | NCD | SOR | add | US | | | | | 0.1 | 10/29/2007 |
| 2702 | 10/29/2007 | (33) MEM SRW differentials add -0.38 cents/bu. (Effective 10/29/2007) | MEM | SRW | add | US | | | | | -0.38 | 10/29/2007 |
| 2703 | 10/29/2007 | (4) KCM HRW differentials add -0.40 cents/bu. (Effective 10/29/2007) | KCM | HRW | add | US | | | | | -0.4 | 10/29/2007 |
| 2704 | 10/29/2007 | (63) GLF YSB differentials in LA, AL, MS add +0.07 cents/bu.(Effective 10/29/2007) | GLF | YSB | in | | LA | AL | MS | | 0.07 | 10/29/2007 |
| 2705 | 10/30/2007 | (19) TKO YC differentials add -0.14 cents/bu. (Effective 10/30/2007) | TKO | YC | add | US | | | | | -0.14 | 10/30/2007 |
| 2706 | 10/30/2007 | (9) MEM YSB differentials add -0.30 cents/bu. (Effective 10/30/2007) | MEM | YSB | add | US | | | | | -0.3 | 10/30/2007 |
| 2707 | 10/31/2007 | (19) TKO YC differentials add -0.17 cents/bu. (Effective 10/31/2007) | TKO | YC | add | US | | | | | -0.17 | 10/31/2007 |
| 2708 | 10/31/2007 | (24) NCD SOR differentials add -0.10 cents/cwt. (Effective 10/31/2007) | NCD | SOR | add | US | | | | | -0.1 | 10/31/2007 |
| 2709 | 10/31/2007 | (8) TKO YSB differentials add -0.36 cents/bu. (Effective 10/31/2007) | TKO | YSB | add | US | | | | | -0.36 | 10/31/2007 |
| 2710 | 10/31/2007 | (9) MEM YSB differentials add -0.40 cents/bu. (Effective 10/31/2007) | MEM | YSB | add | US | | | | | -0.4 | 10/31/2007 |
| 2711 | 11/1/2007 | (6) GLF YSB differentials add -0.65 cents/bu. (Effective 11/1/2007) | GLF | YSB | add | US | | | | | -0.65 | 11/1/2007 |
| 2712 | 11/2/2007 | (19) TKO YC differentials add -0.20 cents/bu. (Effective 11/2/2007) | TKO | YC | add | US | | | | | -0.2 | 11/2/2007 |

9

Fig. 5. Input of temporary differentials from FSA website was automated in June 2007. This is an Excel file with the output – for documentation.

## CONCLUSION

This paper shows how to SAS-automate a manual process of extracting data from text files that have inconsistent structure and format. Automation of this tedious and error-prone process helps reduce typos and allows for automatic verification. The automation saves a significant amount of time and reduces the likelihood of errors.

## ACKNOWLEDGMENTS

Thanks to Paul Dorfman for talking to me about SAS from time to time, and also to Ian Whitlock, Howard Schreier, and Mike Rhoads for giving me useful comments on my NESUG 2009 paper. Thanks to Joy Harwood, the Economic and Policy Analysis Staff director at the Farm Service Agency, USDA; and to Terry Hickenbotham, my supervisor at the Farm Loan Analysis Group, for allowing me to work on and present this paper at SAS meetings and conferences.

## CONTACT INFORMATION

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

Anastasiya Osborne
Farm Service Agency, USDA
1400 Independence Ave, S.W.
Washington, DC 20250-0506
Work Phone: 202-690-0446. E-mail: [Anastasiya.Osborne@wdc.usda.gov](mailto:Anastasiya.Osborne@wdc.usda.gov)