

Proving it Works: Using PROC COMPARE to Verify an Analysis Converted into SAS® Software

by Lauren Haworth, MA, Programmer,
and Njeri Karanja, Ph.D., R.D., Nutrition Scientist
Kaiser Permanente Center for Health Research, Portland, Oregon

Abstract

When SAS® Software is used to replace legacy software systems, programmers are often asked to recreate analyses run on the old system. The challenge for the programmer is to make the results from the new system match the results from the old system, so analyses will be comparable over time.

This paper explores the use of PROC COMPARE to prove that a new analysis system written in SAS produces results that exactly match those produced by a legacy software system. The example in this paper involves the analysis of a survey form called the Food Frequency Questionnaire (FFQ), which is used in nutrition research to capture information about a respondent's typical daily diet.

Introduction

The frequency with which food is consumed is one of a variety of methods for assessing the dietary intake of groups and individuals.¹ The frequency method asks respondents to report the frequency with which they consume each food, from a pre-selected list of foods, over a specified time period.^{2,3} The information is then used to develop a dietary profile for the individual respondent or the group.

There are many food frequency instruments developed by different groups to meet a variety of research needs. One such instrument is the Food Frequency Questionnaire that is part of the Health Habits and History Questionnaire developed by the National Cancer Institute (NCI).⁴ The NCI FFQ has been calibrated and validated against other dietary assessment methods in different populations.^{5,6}

The FFQ is analyzed using special NCI-designed software called DietSys.⁷ DietSys software was originally developed in the 1980s as an analysis tool for the FFQ. Both the FFQ and DietSys were again updated in 1992. DietSys estimates the intake of 33 nutrients and up to 20 user-defined food groups. The software allows users to enter and verify data, standardize editing, and calculate a variety of health habits such as the frequency of restaurant eating and vitamin usage.⁸

The FFQ was used to assess baseline food and nutrient intake in the Dietary Approaches to Stop Hypertension

(DASH) study.⁹ The study tested the impact of dietary change on blood pressure.

The problem

The NCI recently discontinued its policy of updating DietSys and providing direct technical support to users of the software. This leaves research scientists with a software package that is unable to support newer versions of the FFQ, resulting in a limited ability to assess changes in the food supply and eating habits. Additionally, DietSys has always had a few inherent operating glitches that make analysis inefficient. For example, DietSys does not provide much flexibility for running custom analyses. One has to run the program repeatedly to test different analysis options, and there is no support for automation or batch processing of food records. For these reasons, we decided to develop a more flexible and upgradeable system using SAS.

For our new program to be credible, we needed to demonstrate that results obtained using our new SAS program (called CHRFFQ) were comparable to those that would be obtained with the DietSys program. To accomplish this we conducted a test run of the DASH baseline dietary data on CHRFFQ and DietSys and used PROC COMPARE to validate the results. This paper will describe the development and testing of CHRFFQ, showing how SAS was an invaluable tool in the process.

Step 1: Rewriting the program in SAS

Our first step in building CHRFFQ was to use the DietSys documentation to write SAS code that we thought would match the DietSys results. Thankfully, the DietSys software is in the public domain, so we did not have to deal with any copyright issues.

The analysis of FFQ forms involves computing nutrient intakes by multiplying the reported frequency for a given food with the amount of nutrient in a specified quantity of that food. Total daily nutrient intake is the sum of the product of the frequency and nutrient amounts in all the foods.

The FFQ also asks more general questions such as how many vegetables one eats, what fats one uses in cooking, favorite brands of cereal, and what vitamins one takes. These results are used to adjust nutrient calculations. The

result is an extremely complex analysis that is not easily reproduced.

The DietSys documentation outlined in great detail each of the nutrient calculations and the effect of each of the adjustments. It also explained how each of the analysis options was implemented. This documentation proved invaluable to the development process.

Our SAS code was written to use the same calculations and adjustments. The resulting program had over three thousand lines of code. About half of the CHRFFQ development time was spent building the basic program and getting it to run error-free.

Step 2: Setting up the test

Once the SAS program was running without errors, we started the long process of getting it to match the results produced by DietSys. Though we tried to copy the DietSys algorithm, in many cases it was unclear from the documentation how DietSys was calculating certain results. The only way to see if we had gotten it right was to test CHRFFQ against DietSys.

We set up a test dataset to use to compare the two systems. Over 400 FFQ forms from the DASH study were entered into DietSys. This produced an ASCII file that was accepted by DietSys, and could be read into SAS using an INPUT statement.

Once the data file was ready, we ran it through the DietSys analysis and the CHRFFQ analysis using the same option settings. For the first test run, we left all of the options at their simplest setting (in most cases this meant turning the option off).

Next we converted the DietSys results into a SAS dataset. Now we were ready to compare the results. Our goal was to keep testing and revising CHRFFQ until we could get to the point where any differences between the CHRFFQ and DietSys results were less than 1% of each outcome measure.

Step 3: Running PROC COMPARE

PROC COMPARE is the perfect tool for comparing two files. By specifying a common variable as an identifier, it conducts an observation by observation, variable by variable comparison to see if each of the data points is the same. It then produces a detailed report outlining the differences. It tells you which observations are on one file but not the other, which variables are on one file but not the other, and which variables have different values in the two files. The latter check was the most useful for our purposes. We used this feature to make sure that both systems gave the same result for each of the nutrients for each FFQ form.

For the first pass, we tried running PROC COMPARE with the default option settings. We specified the base dataset (DIETSYS), the comparison dataset (CHRFFQ)

and the variable to use to identify the matching observations (ID):

```
PROC COMPARE BASE=DIETSYS COMPARE=CHRFFQ;
  ID ID;
RUN;
```

This proved to be a mistake. At this stage of the development process, there were too many differences between the two files. PROC COMPARE produced hundreds of pages of output.

To highlight the major problems, we changed the PROC COMPARE settings. Instead of running every possible comparison at once (the PROC COMPARE default setting), we used a step-by-step approach. The first step was to confirm that both files had the same variables. To do this, you specify the LISTVAR option and the NOVALUES option. LISTVAR specifies a comparison of variables, and NOVALUES turns off the value comparison:

```
PROC COMPARE BASE=DIETSYS COMPARE=CHRFFQ
  LISTVAR NOVALUES;
  ID ID;
RUN;
```

This run pointed out a couple of variables we had forgotten to compute in CHRFFQ. We added the variables and re-ran PROC COMPARE to make sure we had fixed the problem. This time PROC COMPARE produced no output (that is what PROC COMPARE does when the two files are identical for each of the comparisons specified in the options settings).

The next step of our testing was to make sure that both systems produced the same number of observations on the output file. To do this, you specify the LISTOBS option:

```
PROC COMPARE BASE=DIETSYS COMPARE=CHRFFQ
  LISTOBS NOVALUES;
  ID ID;
RUN;
```

This run pointed out that the two systems were using different criteria to select invalid observations for deletion. By looking at the PROC COMPARE output, we could identify which records had been incorrectly dropped, figure out what caused them to be dropped, and correct the problem.

Having confirmed that the two files had the same variables and observations, we were ready to start comparing values in the two files. We removed the NOVALUES option to turn on the values comparison. Also, in order to limit the amount of output in each run, we made three additional changes to the options settings.

First, we limited the comparison to ten observations by using the OBS= parameter on each of the two datasets. Ten observations were enough to spot major differences at this point. Second, we changed the comparison method

to METHOD=PERCENT and CRITERION=20. This means that only values that were off by more than 20% were reported. Third, we limited the comparison to one variable: total calories (CALs). Since this variable summarizes the respondent's diet, it was a useful tool for testing all facets of the program. The new run used the following code:

```
PROC COMPARE BASE=DIETSYS (OBS=10)
  COMPARE=CHRFQ (OBS=10)
  METHOD=PERCENT CRITERION=.20;
  ID ID;
  VAR CALS;
RUN;
```

With this setup, PROC COMPARE ran more quickly and produced a much smaller report showing the major differences between the two files. By looking at the report, we were able to identify which part of the program was producing incorrect results.

We then began an iterative process of finding a problem, fixing the program, and then running PROC COMPARE again, each time coming closer to matching the DietSys results.

In most cases, differences between the two systems were caused by errors in our CHRFQ program. But in some cases, it turned out that CHRFQ was computing the results correctly, and DietSys was in error.¹⁰ Since we couldn't fix the problems with DietSys, in these cases we wrote code to reproduce the errors in CHRFQ. We used comments in the SAS code to mark the location of the alteration. For example, here is one of the errors we replicated:

```
*****
*
*ERROR REPRODUCED FOR VALIDATION
PURPOSES:
* (SEASONAL ADJUSTMENT NOT IMPLEMENTED)
*CORRECT CODE:
* NUT=FREQ*SERVSIZE*NUTPERG*SEAS
*INCORRECT CODE TO USE FOR NOW:
*****;
  NUT=FREQ*SERVSIZE*NUTPERG;
```

With this code in place, we could re-run the CHRFQ analysis and it would replicate the error we found in DietSys. Eventually, PROC COMPARE was run and no differences were reported.

At this point we were ready to run PROC COMPARE on the full dataset, instead of just 10 observations. The CRITERION setting was left unchanged, and the comparison was still limited to one variable:

```
PROC COMPARE BASE=DIETSYS COMPARE=CHRFQ
  METHOD=PERCENT CRITERION=.20;
  ID ID;
  VAR CALS;
RUN;
```

This identified more differences between the two datasets, caused by situations that did not occur with the 10 test observations. Some errors only showed up in foods that only a few people reported eating, so the problems did not appear until we analyzed hundreds of forms.

Once we had fixed these problems, and PROC COMPARE reported no differences for any observation, we began running the comparison on all variables:

```
PROC COMPARE BASE=DIETSYS COMPARE=CHRFQ
  METHOD=PERCENT CRITERION=.20;
  ID ID;
RUN;
```

This round of tests identified some problems with individual nutrients that did not affect the CALS variable.

Step 4: Fine-tuning PROC COMPARE

Now we were ready to start looking for more subtle differences between the two files. So far, we had only looked for values that varied by more than 20%. To look for smaller differences, we began reducing the CRITERION level. First we ran it at 10%. Then, as we continued to find and correct errors, the level was lowered to 5% and finally to 1%, our original goal.

When we got to the 1% level, we discovered an interesting problem. The two systems were producing different results for some nutrients due to rounding errors. DietSys was only carrying two decimal places in most of the calculations. Of course SAS was doing its computations at a much higher level of precision. So for nutrients where the result was a very small number, a difference of .01 (the rounding error), could be larger than the 1% CRITERION value.

Replicating the rounding errors in SAS would have been a major challenge since so many variables and computations were involved (it would have required hundreds of LENGTH statements), so instead we reset the comparison method in PROC COMPARE.

In the previous runs, we used the METHOD=PERCENT to look for values that were not within a specified percentage of each other. But there is also a setting called ABSOLUTE that compares the absolute difference between two values. The difference is only reported if it is at least as large as the CRITERION setting. For example, if the CRITERION setting is 1, METHOD=ABSOLUTE tells SAS that two values are different only if $ABS(A-B) > 1$.

Since we knew that DietSys should match CHRFQ up to the limits of its rounding (two decimal places), we set the METHOD to ABSOLUTE and the CRITERION to 0.01, which eliminated our problem:

```
PROC COMPARE BASE=DIETSYS COMPARE=CHRFFQ
  METHOD=ABSOLUTE
  CRITERION=0.01;
  ID ID;
RUN;
```

Any differences reported by this PROC COMPARE run would be true differences, and not caused by rounding. Eventually, we reached a point where PROC COMPARE was no longer reporting any differences at this level of precision.

Step 5: Varying the analysis

The tests we had done so far were based on running DietSys and CHRFFQ with all of the analysis options turned off. The final step in our process was to turn the options on one at a time and run PROC COMPARE to be sure that they were the same in both systems.

Once we fixed the last few errors, we were able to show that PROC COMPARE would find both files to be similar or identical, even with all of the analysis options turned on.

Now we had our necessary proof in hand. We can use the PROC COMPARE output to show that CHRFFQ can replicate DietSys results to within 0.01 on every variable. The output from the final PROC COMPARE run is shown in Appendix A.

Validating CHRFFQ was a lengthy process. We spent as much time testing with PROC COMPARE and revising the code as we did writing it in the first place.

Step 6: Documentation

Once the validation process was complete, we created an archive that holds: (1) the PROC COMPARE output showing the matching results, (2) the test input dataset, (3) the DietSys results file, (4) the CHRFFQ results file, (5) the SAS code exactly as it was when the results matched, and (6) the same SAS code with the replicated errors removed (the comments were left in place where each error occurred to leave a documentation trail of the differences between CHRFFQ and DietSys). This allows us to prove that our program works, and it served as a reference for the next phase of our development effort: adding features that were lacking in DietSys.

Step 7: Enhancements

After confirming that the basic CHRFFQ was comparable to DietSys, we were in a position to add new features that were lacking in DietSys. We have now modified the code to work with a 1995 version of the FFQ portion of the HHHQ (Block Dietary Systems, unpublished communication). The new FFQ has been updated to reflect the greater supply of low-fat foods and more vegetarian foods, and includes 8 additional foods. The DietSys nutrient reference database only had room to add a limited number of new foods, and we had already used all of

those slots. With CHRFFQ, we were able to add unlimited new foods to the reference database.

Other changes that we made included adding new variables to the results file. We wanted to be able to create more than the 18 standard and 2 optional food groups computed by DietSys. In addition, we wanted to obtain results that showed how study participants were eating relative to current eating guidelines. In the time since DietSys was created, healthy eating guidelines have changed from the food group format to the food pyramid guide.¹¹ Using the flexibility afforded by CHRFFQ, we were able to expand our results file to include the old food groupings as well as the food groupings defined in the food pyramid.

With these enhancements in place, CHRFFQ is no longer an exact replication of DietSys. But with our testing process, we have proved that it is as close as possible to the old methodology. We can even use PROC COMPARE to show what the differences are between DietSys and the enhanced version of CHRFFQ, by running the validated SAS output against the enhanced SAS output in another PROC COMPARE.

Conclusions

We can now be comfortable that CHRFFQ can correctly analyze the DASH FFQ forms, and that our results will be comparable to those of prior studies that used DietSys for analysis.

Though our goal was to build an FFQ analysis system, we also have shown that SAS provides a useful tool for validating your work. PROC COMPARE is good way to show that your new programs correctly replicate the work of the programs they replace.

PROC COMPARE gave us enough information to identify and correct each of the differences between the two analysis systems, and it could be customized to produce an appropriate level of checking at each stage of the process. The ability to start by looking for gross errors and then gradually refine our checks to look for more subtle errors was invaluable.

This validation process should be a routine step any time you use SAS to replicate the work of a legacy system. Instead of saying “My new program *should* produce comparable results” you can say “I can *prove* that my new program produces comparable results.”

And while this example involved using SAS to replace another software package, you could also use this approach to validate a new SAS process against an old SAS process. We encourage you to try this approach on your next major programming project.

Acknowledgments

Linda Coyle, a programmer at the National Cancer Institute, provided technical assistance and documentation related to DietSys.

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

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

Notes

¹ Thompson FE, Byers T, Kohlmeier L (1994). Dietary assessment resource manual. *J. Nutr* 124 (115): 22475-22495.

² Willett WC (1990). *Nutritional Epidemiology*. Oxford University Press, New York.

³ Zulkifli SN, Yu SM (1992). The food frequency method for dietary assessment, *J. Am. Dietet. Assoc.* 59:1715-1745.

⁴ Block G, Hartman AM, Dresser CM, Carroll MD, Gannon J, Gardner L (1986). A data based approach to diet questionnaire design and testing. *Am. J. Epidemiol.* 124:453-469.

⁵ Block G, Thompson FE, Hartman AM, Larkin FA, Guire KE (1992). Comparison of two dietary questionnaires validated against multiple dietary records collected during a 1-year period. *J. Am. Diet. Assoc.* 6:686-693.

⁶ Coates RJ, Eley JW, Block G, Guner EW, Sowell AL, Grossman C, Greenberg RS (1991). An evaluation of a food frequency questionnaire for assessing dietary intake of specific carotenoids and vitamin E among low income black women. *Am. J. Epidemiol.* 134:658-671.

⁷ Block G., Coyle LM, Hartman AM, Scoppa SM (1993). *HHHQ-DietSys Analysis Software*, version 3.0. National Cancer Institute, Bethesda, MD.

⁸ Block G, Coyle LM, Hartman AM, Scoppa SM (1994). Revision of dietary analysis software for the health habits and history questionnaire. *Am. J. Epidemiol.* 139:1190-1196.

⁹ Appel LJ, Moore TJ, Obarzanek E, Vollmer WM, Svetkey LP, Sacks FM, Bray GA, Vogt TM, Cutler JA, Windhauser MM, P Lin, Karanja N for the DASH Collaborative Research Group (1997). A clinical trial of the effects of dietary patterns on blood pressure. *The New England Journal of Medicine.* 336:1117-1124.

¹⁰ Problems encountered included an incorrect adjustment for foods eaten seasonally (fresh fruit), an incorrect adjustment from monthly to weekly frequencies, and incorrect serving-size adjustments for milk.

¹¹ Department of Agriculture, Department of Health and Human Services. *Nutrition and your Health: Dietary Guidelines for Americans*. 4th ed. Washington, D.C.: Government Printing Office, 1995.

References

SAS Institute Inc., *SAS Procedures Guide, Version 6, Third Edition*, Cary, NC: SAS Institute Inc., 1990.

Contacting the Authors

Please direct any questions or feedback to the authors at:

Kaiser Permanente Center for Health Research
3800 N. Kaiser Center Drive
Portland, OR 97227-1098

E-mail: haworthla@chr.mts.kpnw.org

Appendix A: Final PROC COMPARE Output

If you run the following code ...

```
PROC COMPARE BASE=TEMPDIET COMPARE=TEMPCHR
      METHOD=PERCENT CRITERION=.01 LIST TRANSPOSE;
      ID ID;
RUN;
```

... you get the following output (once all of the significant differences have been resolved):

Data Set Summary

Dataset	Created	Modified	NVar	NObs
WORK.TEMPDIET	21NOV97:13:00:58	21NOV97:13:00:58	34	405
WORK.TEMPCHR	21NOV97:13:02:38	21NOV97:13:02:38	34	405

Variables Summary

Number of Variables in Common: 34.
 Number of ID Variables: 1.

Observation Summary

Observation	Base	Compare	ID
First Obs	1	1	ID=101
Last Obs	405	405	ID=901

Number of Observations in Common: 405.
 Total Number of Observations Read from WORK.TEMPDIET: 405.
 Total Number of Observations Read from WORK.TEMPCHR: 405.

Number of Observations with Some Compared Variables Unequal: 0.
 Number of Observations with All Compared Variables Equal: 405.

NOTE: No unequal values were found. All values compared are exactly equal.