ABSTRACT
This paper is concerned about the speed of inputting and sorting large datasets using SAS. Usually people use the only one SAS procedure Proc SORT to input data and sort data as well. However, this procedure is not efficient when the dataset is large. A two-step method by which first the data is input using DATA step and then is sorted using procedure SORT can enhance the speed of inputting and sorting a large dataset. Experiments on large clinical datasets show that the two-step method reduces the run time greatly.

Key words: inputting, sorting, one-step method, two-step method, large dataset, CPU time.
Platform: OS/390.
SAS product: SAS/Basic version 8.2.

INTRODUCTION
In the process of applying SAS, the first steps are usually inputting and sorting data. We can use two methods to input and sort a dataset. The first method is to use one integrated SAS procedure PROC SORT to do that. The second is a two-step method. It uses DATA step to input data and then uses procedure PROC SORT to sort data. Both methods use the same procedure SORT. But in the second method, a DATA step is used before procedure SORT is called. Lots of people prefer the first one-step method since it appears that it runs faster than two-step method. But for large datasets the situation is different.

This paper compares the two methods by experiments on large datasets. The results show that the two-step method runs faster than the one-step method.

The dataset we used for the experiment is clinical lab test data consisting of more than 1 million records and dozens of variables. The key variables are study and patient id numbers.

COMPARISON OF THE TWO METHODS

TWO COMMON METHODS OF INPUTTING AND SORTING DATA
The most widely used one-step method inputs and sorts dataset using only one procedure SORT. Typical code is as follows:

```sas
PROC SORT DATA=IN.LABS(WHERE=(LAB_TYPE='SI')) OUT=LAB1;
   BY STUDY PATIENT LABCODE VISIT SAMPSTD SAMPSEQ;
RUN;
```

This method uses one single procedure PROC SORT to both input and sort the dataset.

One alternative method is the so-called two-step method. It uses applies DATA step first and then call the procedure SORT to sort the input dataset. The typical code looks like:

```sas
DATA LAB2;
   SET IN.LABS;
   IF LAB_TYPE='SI';
RUN;
PROC SORT DATA=LAB2;
   BY STUDY PATIENT LABCODE VISIT SAMPSTD SAMPSEQ;
RUN;
```

Comparing the codes of the two methods, we can see that the one-step method has only the second part of the two-step method.

EXPERIMENT RESULTS
In order to compare the speed of the two methods, the above programs are tested using large clinic datasets on platform OS/390, respectively. The difference of the speed of two methods can be seen from the log files of the two programs.

The log file from the one-step contains the following notes:

```
NOTE: There were 1481583 observations read from the dataset IN.LABS.
   WHERE LAB_TYPE='SI';
```
The total CPU time used by this method is 39.77 seconds.

The log file from the two-step method contains the following notes:

NOTE: There were 1690444 observations read from the dataset IN.LABS.
NOTE: The dataset WORK.LABS has 1481583 observations and 43 variables.
NOTE: The DATA statement used 6.63 CPU seconds and 9494K.
NOTE: The PROCEDURE SORT used 12.79 CPU seconds and 10039K.

By this method, the CPU time used by DATA step and SORT procedure are 6.63 seconds and 12.79 seconds, respectively. Therefore the total CPU time used is 19.42 seconds. In the experimental dataset, there are about 1.5 million observations and 43 variables. The two-step method saves about 51% CPU time.

CONCLUSIONS
The experiment was conducted on multi-tasking platform OS/390, so the CPU time may fluctuate. In order to minimize the effect of sharing of CPU with other users, we repeated the experiment several times at different time. Table 1 lists the CPU time for the two different methods, conducted at different time.

<table>
<thead>
<tr>
<th></th>
<th>CPU time (in seconds) records for the two methods</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>One-step method</td>
</tr>
<tr>
<td>39.77</td>
<td>6.63+12.79=19.42</td>
</tr>
<tr>
<td>37.91</td>
<td>6.60+13.18=19.78</td>
</tr>
<tr>
<td>38.49</td>
<td>6.30+13.23=19.53</td>
</tr>
<tr>
<td>38.53</td>
<td>6.35+13.25=19.60</td>
</tr>
<tr>
<td>37.71</td>
<td>6.21+13.09=19.30</td>
</tr>
</tbody>
</table>

The results in Table 1 were obtained a clinic data set that has mbout 1.5 million observations and 43 variables. This table shows that the two-step method saves about 50% CPU time. Similar experiments are conducted on other datasets and similar conclusions are obtained. Therefore we conclude that the two-step method can save CPU time significantly.

ACKNOWLEDGMENTS
The author thanks valuable discussions and suggestions from Dr. Yili Lu, Mr. Bruce Spotts, Mrs. Lingling Xie and Eli Lilly and Company.

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

Lin Yan
Consultant at Eli Lilly and Company
Indianapolis, IN 45021
Work Phone: (317) 433-0674
Email: yan_lin_nonlilly@lilly.com