Proceedings of the 1987 Winter Simulation Conference
A. Thesen, H. Grant, W. David Kelton (eds.)

SYSTEMS FOR MONTE CARLO WORK

David Alan Grier
Department of Stat., Comp. and Info. Sciences
George Washington University
Washington DC 20052

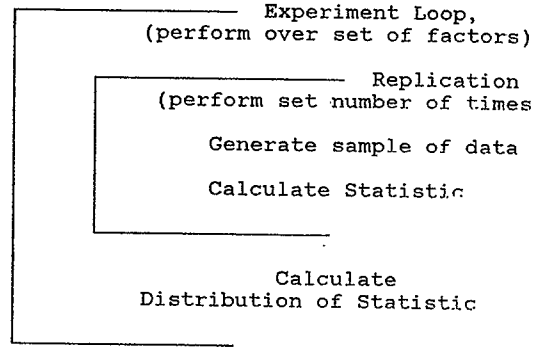
With the proliferation of computers has come a proliferation of simulation. Monte Carlo experiments can now be run by a vast range of programs from simple Basic environments to spreadsheets; yet little attention has been paid to the problem of designing a system to do Monte Carlo problems. The ideas for a system described in this paper not only simplifies the problem of programming a Monte Carlo experiment but also attempts to maintain standards of experimental design and to encourage careful analysis.

Section 1. The Problem

Monte Carlo experiments are sampling experiments, performed on a computer, usually done to determine the distribution of a statistic under some set of probabilistic assumptions. These assumptions are such that the distribution of the statistic in question cannot be calculated in closed form. Examples of such assumptions include the assumption of a finite sample size or of nonstandard sampling distributions.

An example of a problem that might be solved using Monte Carlo experimentation is the following: consider the simplified Wilks-Shapiro test for the normal distribution of data, W'. This test attempts to determine if a set of data come from a gaussian distribution. It is the correlation of the sorted data with order statistics from a standard normal distribution. The distribution of this statistic, needless to say, is difficult to determine, and is known only in the case that the data actually come from a normal population in the limiting case as the sample size becomes infinite (Leslie, Stephens, and Fotopoulos, 1986). Knowing the distribution of the statistic in finite sample cases and when the data are not quite normal but come from a contaminated normal or a t distribution would help to understand the power of the statistic. However, calculation these distributions are intractable at present and must be estimated by a Monte Carlo experiment (Gastworth and Grier).

The paradigm for a Monte Carlo experiment is the doubly nested loop program diagrammed in Figure 1. The inner loop of the program is the sampling loop. Within that loop, the computer generates a sample of data from a probability model and calculates the value of a statistic. In our Wilks-Shapiro example, this within this loop the program would repeatedly draw a sample of fixed size from an identified distribution and calculate the simplified Wilks-Shapiro statistic. The distribution of the statistic would be calculated from the set of observed values of the Wilks-Shapiro statistic.



Monte Carlo Program Model
Figure 1

The outer loop in the paradigm is the experiment loop. This loop changes the values of different factors in the experiment. One common factor is the size of the sample. One factor in the Wilks-Shapiro example would be the distribution of the data. In programming this experiment, we might vary the distribution over a set containing the gaussian distribution, certain members of the t family of distributions and certain members in the contaminated normal family of distributions.

While the paradigm described above is typical of Monte Carlo Experimentation, it may be transformed to accommodate the needs of the problem. Often the amount of computing may be reduced by modifying the basic program. A common technique is to invert the two loops so that a single sample may be reused. Another technique involves moving only part of the outer loop inside of the replication loop, as, for example, in the case of determining the true rejection rate of a test. In that case, the test may be preformed at 4 or 5 different levels on a common data set and the true rejection rate computed at each level.

For this system, a new looping operator is introduced to handle the paradigm of Monte Carlo programs. This operator loops over a set produced by creating an experimental design from the input parameters, which are themselves sets. For example, suppose that we have three input parameters, A, B, and C, and that they each can take a value from a set of three items: $\{a_1, a_2, a_3\}$ for A, $\{b_1, b_2, b_3\}$ for B and $\{c_1, c_2, c_3\}$ for C. If we use the new operator to combine these sets into a factorial design, it will cause the program to loop over the set:

$\{\{a_1,b_1,c_1\}, \{a_1,b_1,c_2\}, \{a_1,b_1,c_3\},$
$\{a_1,b_2,c_1\}, \{a_1,b_2,c_2\}, \{a_1,b_2,c_3\},$
$\{a_1,b_3,c_1\}, \{a_1,b_3,c_2\}, \{a_1,b_3,c_3\},$
$\{a_2,b_1,c_1\}, \{a_2,b_1,c_2\}, \{a_2,b_1,c_3\},$
$\{a_2,b_2,c_1\}, \{a_2,b_2,c_2\}, \{a_2,b_2,c_3\},$
$\{a_2,b_3,c_1\}, \{a_2,b_3,c_2\}, \{a_2,b_3,c_3\},$
$\{a_3,b_1,c_1\}, \{a_3,b_1,c_2\}, \{a_3,b_1,c_3\},$
$\{a_3,b_2,c_1\}, \{a_3,b_2,c_2\}, \{a_3,b_2,c_3\},$
$\{a_3,b_3,c_1\}, \{a_3,b_3,c_2\}, \{a_3,b_3,c_3\}\}$

If we use the new operator to combine these sets into a latin square design, it will cause the program to loop over the set:

$\{\{a_1,b_1,c_1\}, \{a_1,b_2,c_2\}, \{a_1,b_3,c_3\},$
$\{a_2,b_1,c_2\}, \{a_2,b_2,c_3\}, \{a_2,b_3,c_1\},$
$\{a_3,b_1,c_3\}, \{a_3,b_2,c_1\}, \{a_3,b_3,c_2\}\}$

Section 2 Organization of System

The need for a system to do Monte Carlo Experimentation has become increasingly evident. Monte Carlo experimentation has become an important tool in statistical research and yet virtually all of the Monte Carlo work is done by custom higher level language programs. A recent survey of published Monte Carlo work (Hauck and Anderson, 1984) shows that the bulk of published Monte Carlo studies need the systematization that statistical analysis packages, such as SAS, S and SPSS have brought to statistical analysis. In most Monte Carlo experiments, the only sections of codes that are unique to the problem at hand are the section of code that generates the random data set and the section that evaluates the statistic of study. The rest of the code just gathers the results of the statistic and supports the double loop structure of the standard Monte Carlo program as defined in Section 1 above.

In published Monte Carlo studies, it is common to find questionable random number generators, unanalyzed results and poorly designed experiments (Hauck and Anderson, 1984); and it is these problems that a Monte Carlo System must be prepared to address. The problem of the questionable number generators is the easiest to solve by having the Monte Carlo System provide a library of carefully chosen, well tested random number generators. The problems of unanalyzed results and poorly designed experiments are harder to address. It is common, in the literature, to read of experiments that have been performed over an unrepresentative sample of the factor space and that have results that have been simply tabulated without being analyzed, leaving the reader to fathom the results. (See, for example, the results of an otherwise good experiment in Bartels, 1982) This problem cannot be handled in a foolproof manner by any system, just as a statistical analysis cannot be said to be perfect just because it was done with one of the standard packages. However, this system attempts to encourage well designed experiments and careful analyses by casting the whole problem of Monte Carlo Experimentation into the mold of classical statistical design of experiments.

Our Monte Carlo System gives researchers a tool that generates a Monte Carlo program, called a target program, in the form given in Section 1 above. The researcher has to provide three things:

1. A routine to generate the random data set, usually assembled from the library of random number generators included with the system.
2. A routine to calculate the statistic of study.
3. An experimental design.

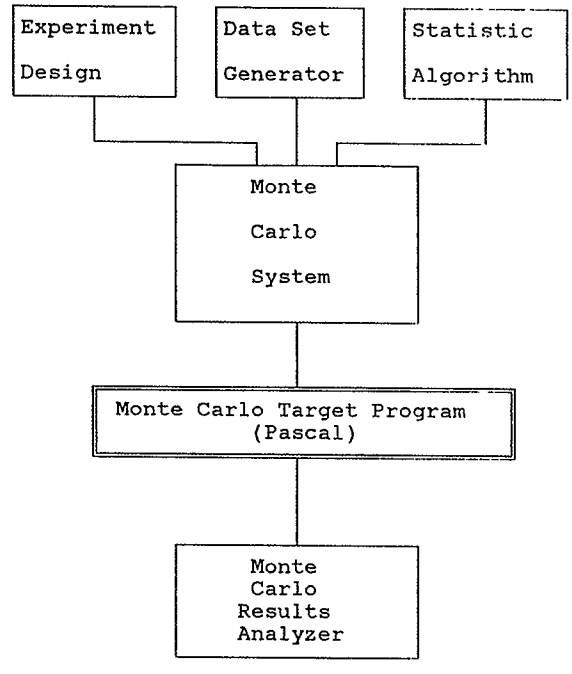
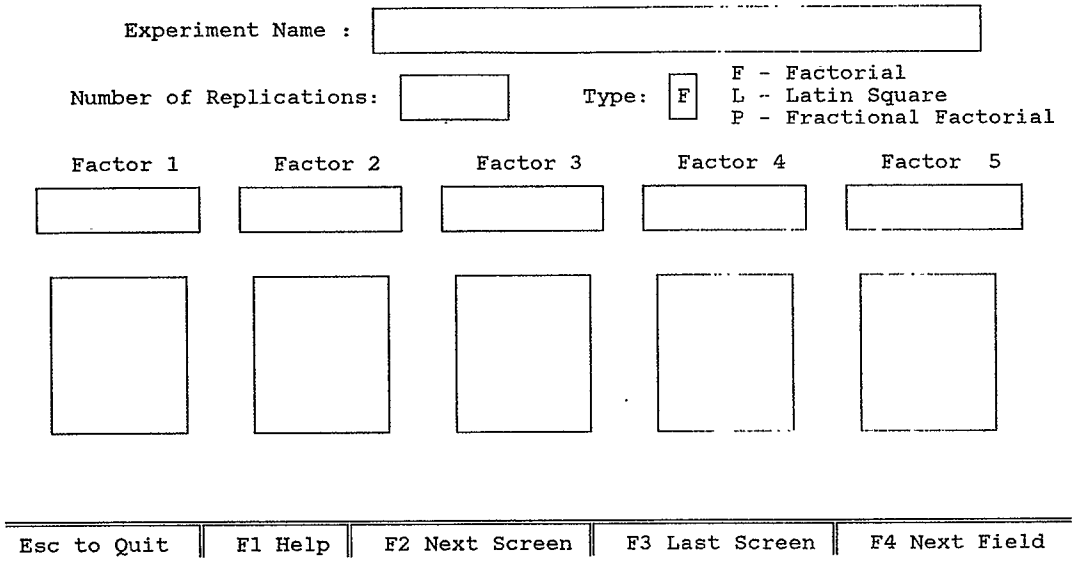


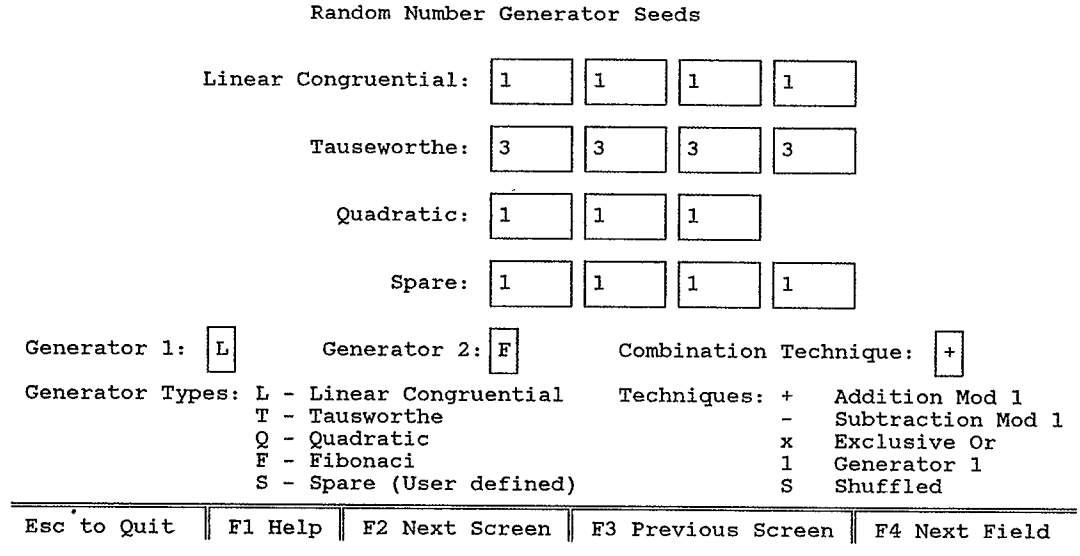
Diagram of Monte Carlo Processor
Figure 2

The target program is a separate program, written in a high level language. (The current language of the target program is Pascal.) This program is then compiled and run. The output of the program can then be read by another element of the system, the analysis program, that can analyze the results of the experiment, tabulate the results or write them in a form that may be read by a statistical analysis program such as S, SPSS, or SAS. The organization of the system is seen in Figure 2.

The experimental design is entered in a menu, as pictured in Figure 3. The experimenter enters the names of the factors and the possible numerical values that they may hold, the number of replications for each point in the experiment and the type of experiment design (factorial, fractional factorial or latin square are the current options). The algorithms for data generation and for the evaluation of the statistic are entered in Pascal in an interactive editor. The Monte Carlo system also has a menu screen to allow control of the random number generator. Figure 4 shows pare of the menu for controlling the random number generators.



Experimental Design Menu
Figure 3

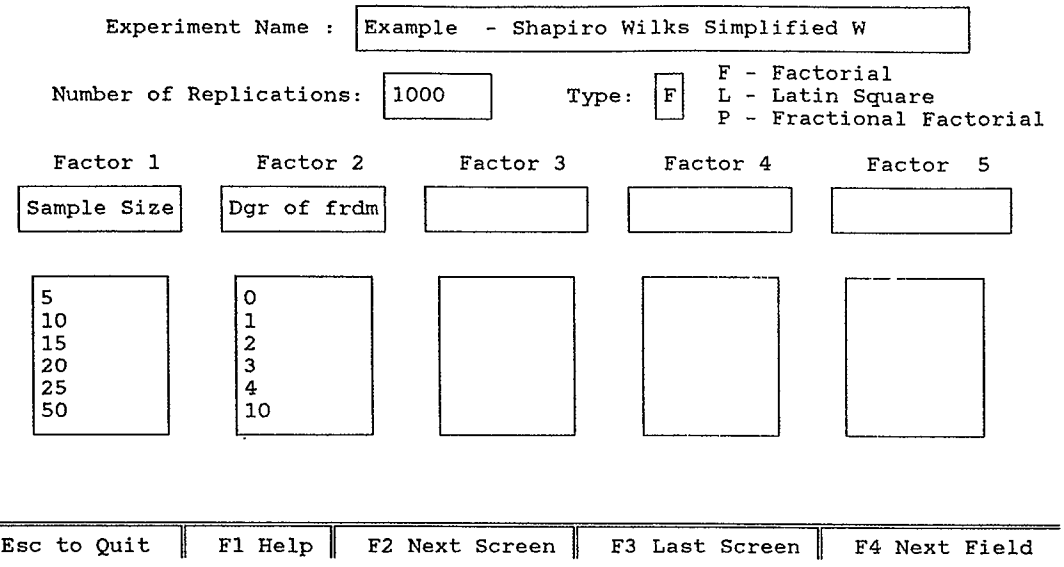


Random Number Control Menu
Figure 4

Section 3 Example

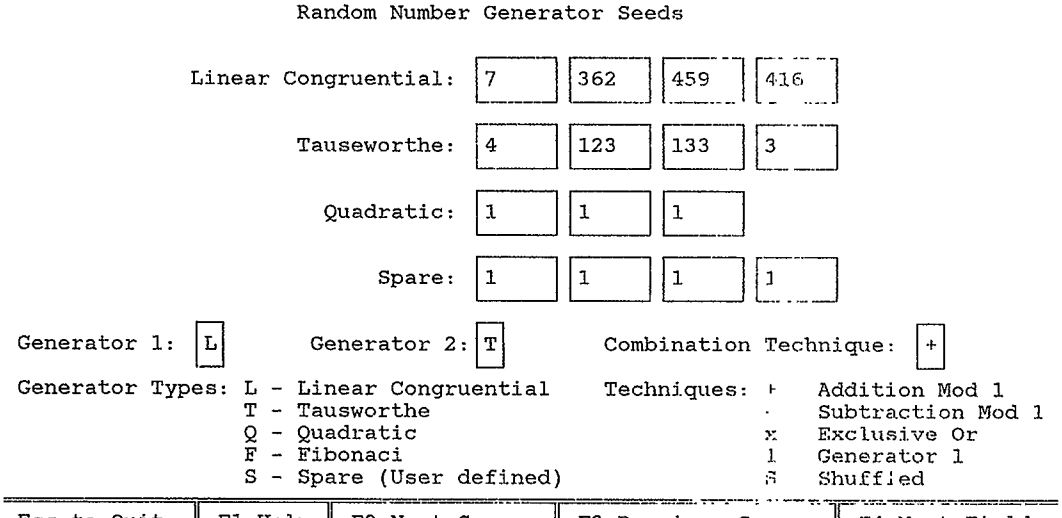
For this example, let us return to the experiment involving the Shapiro-Wilks W´. Let´s prepare an experiment that will calculate the expected value of the statistic for finite sample size and nongaussian distribution. For our experiment, we will use six different sample sizes: 5, 10, 15, 20, 25, and 50. For our nongaussian distributions, we will use five members of the t family with 1, 2, 3, 4 and 10 degrees of freedom. We will also use the gaussian distribution for a reference.

The experiment is presented in Figure 5 on the experiment control screen. There are two factors, Sample size and Degree of Freedom, each with a set of six possible values. The value of 0 for the Degree of Freedom factor is used to indicate a gaussian distribution; see Figure 7 with the code. The experiment is a factorial experiment and it will be replicated 1000 times for each experiment point.



Example Experiment.
Figure 5

The preparation of the random number generator is seen in Figure 6. We are using an additive combination of the linear congruential and the tausworthe generator. The seeds for each generator are set on the top of the screen.



Random Number Control
Figure 6

The Pascal code that performs the experiment is seen in Figure 7. The code performs the experiment once, all looping for the experiment is handled by the system. The code:

1. Generates approximate Normal Order statistics
2. Generates a random sample
3. Sorts the sample
4. Computes the correlation between the sample and the Normal order statistics, which is W'
5. Accumulates the result.

The procedure rnorm generates normal random variables. The procedure rt generates t random variables. The qsort procedure is a quick sort and cor computes the correlation. The procedure accum_ave is a system procedure that accumulates the average of the 1000 replications of each experiment. The system array fact[] contains the current values of the factors. The system variable changed_fact, gives the number of the last factor that changed; each time that changed fact is equal to 1, a new set of order statistics must be generated.

```
Var
    os          : x_array;              {hold normal order stats}
    x           : x_array;              {hold random data}
    i           : Integer;              {index}
    c           : Real;                 {correlation}
    sample_size : Integer;              {sample_size}



Begin
sample_size := trunc(fact[1]);

                        {first prepare normal order stats}
If (changed_fact = 0) Or (changed_fact = 1) Then
    For i := 1 To sample_size Do os[i] := inv_norm((i-0.5)/fact[1]);

For i := 1 To sample_size do
    If (fact[2] = 0) Then          {0 means use normal}
       x[i] := rnorm
    Else
       x[i] := rt(trunc(fact[2])); {generate random t variates}

qsort(x,1,sample_size);             {sort data}

c := cor(x,os,sample_size);        {compute Shapiro-Wilks}

accum_ave(c*c,1);                   {accumulate it in location 1}

End;        {main_procedure}
```

Experiment Code
Figure 7

Boxplots of the results can be seen in Figure 8. These side by side boxplots are made across the Degrees of Freedom factor.

Section 4 Conclusion

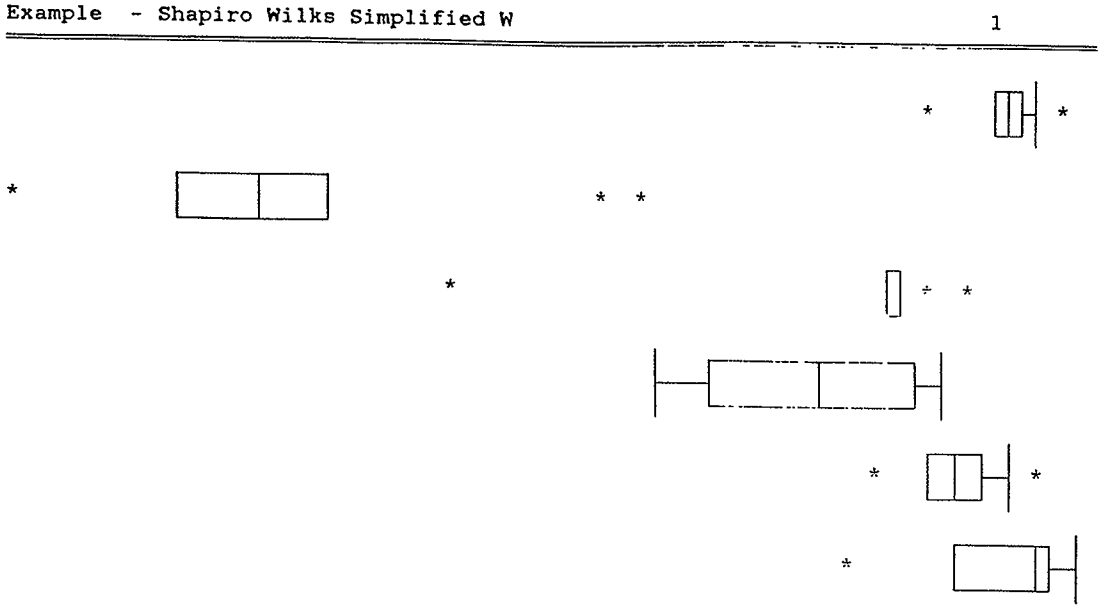
Packages for discrete event simulation have existed for nearly 25 years, yet none have existed for Monte Carlo experimentation. (Friedman and Friedman, 1984) The Monte Carlo System, described in this paper, attempts to systematize the process of preparing Monte Carlo Experimentation. The system prepares most of the program, leaving the researcher to prepare only the experimental design and the sections of code unique to the given experiment. This system not only speeds the process of preparing an experiment but it also encourages a careful design of the experiment and the thorough analysis of the results.

Availability

An experimental version of the software is available from the author at:

The George Washington University
Department of Statistics,
Computer and Information Sciences
Funger Hall
Washington, DC 20052

The software runs on the IBM PC, PS/2, and Apple Macintosh architecture machines, with or without a math coprocessor, and requires the Turbo Pascal programing environment (Borland, 1987). A fee of $10.00 is requested to cover the costs of handling and shipping.



Boxplot of the results
Figure 8

Bibliography

Bartels, R. (1982), "The Rank Version of von Neumann's Ratio Test for Randomness," Journal of the American Statistical Society, 77:377, 40-46.

Borland International Incorporated (1987). Turbo Pascal, 4585 Scotts Valley Drive. Scotts Valley CA, 95066.

Friedman, Linda Weiser and Friedman, Hersheyll (1984), "Simulation: State of the Art", Statistical Computing and Simulation, 19:3, 237 - 256.

Gastworth, J. and Grier, D., "An Adaptive Test for Location", to appear.

Grier, David Alan, A System for Monte Carlo Experimentation, PhD Thesis, University of Washington, Department of Statistics. Seattle, WA, 1986.

Grier, David Alan, A Monte Carlo Processor, Technical Report No. 54, University of Washington, Department of Statistics. Seattle, WA, 1984.

Hauck, W. and Anderson, S. (1984, "A Survey Regarding the Reporting of Simulation Studies," Journal of the American Statistical Society, August 1984. (38:3), 214-216.

Leslie, J. R., Stephens, M. A. and Fotopoulos, S. (1986) "Asymptotic Distribution of the Shapiro-Wilk W for Testing Normality," Annals of Mathematical Statistics, (14,4), 1497-1506.