AGENCY FOR INTERNATIONAL DEVELOPMENT
WASHINGTON, D. C. 20523

BIBLIOGRAPHIC INPUT SHEET

FOR AID USE ONLY
BATCH #14

1. SUBJECT CLASSIFICATION	A. PRIMARY: Agriculture	AA00-0000-0000
	B. SECONDARY: General	

2. TITLE AND SUBTITLE
The uses of Monte Carlo techniques and response surface methodology in complex simulation models

3. AUTHOR(S)
Turnquist,Mark; Manetsch,T.J.

4. DOCUMENT DATE	5. NUMBER OF PAGES	6. ARC NUMBER
1971	42p.	ARC

7. REFERENCE ORGANIZATION NAME AND ADDRESS
Mich.State

8. SUPPLEMENTARY NOTES *(Sponsoring Organization, Publishers, Availability)*
(In Project working paper 71-1)

9. ABSTRACT

10. CONTROL NUMBER PN-RAA-723	11. PRICE OF DOCUMENT
12. DESCRIPTORS Monte Carlo method	13. PROJECT NUMBER
	14. CONTRACT NUMBER CSD-1557 Res.
	15. TYPE OF DOCUMENT

AID 590-1 (4-74)

Agr. M

A SIMULATION MODEL OF THE

NIGERIAN AGRICULTURAL ECONOMY

Contract No. AID/csd-1557

The Uses of Monte Carlo Techniques and Response Surface Methodology in Complex Simulation Models

Project Working Paper 71-1

Prepared By:

Mark Turnquist
Thomas J. Manetsch

February 20, 1971
East Lansing, Michigan

THE USES OF MONTE CARLO TECHNIQUES AND RESPONSE SURFACE METHODOLOGY IN COMPLEX SIMULATION MODELS

Motivation

In the construction and implementation of a complex simulation model, one very difficult task is specifying values for parameters and coefficients of the model. Two of the difficulties are: first, by fixing values, one may be treating deterministically variables which are actually subject to random variation in the "real world," and secondly, that even in variables which are not so much subject to random variation, the specifications of values are often made with something less than total confidence in their accuracy.

In both of these cases it might then be desirable to observe the variations in certain outputs of the model as some of these parameters are allowed to vary over a specified range.

Once one has delineated the set of parameters which he wishes to subject to variation, he must further subdivide this set into those which will be treated as random <u>within</u> runs and those which will be random <u>between</u> runs. Those parameters which seem to exhibit random variation in the real world should probably be treated as random within runs, while those which are fixed, but with less than total certainty in their accuracy, lend themselves to variation between runs.

In order to treat a parameter as random within a simulation run, the model must incorporate the capability for stochastic variation within its

time loop. In the second case, that of parameters random between runs, the model remains deterministic; within each run the parameters are fixed, but they assume different values for each simulation. It should be noted, paranthetically, that these two sets are not necessarily mutually exclusive; it may well be that one may wish to let a parameter vary over some range about its mean within a run, and let its mean vary between runs.

This study is concerned solely with variables random between runs.

Monte Carlo Simulation

Monte Carlo simulation may be defined as the repeated subjection of a stochastic model to the same set of initial conditions in order to generate a probability distribution of outcomes.

It is important to note that it makes no difference whether the model is actually stochastic in nature, i.e., certain variables are drawn from probability distributions within a run, or is actually deterministic, but with random variation of parameters between runs; for the purposes of Monte Carlo studies, it behaves as a stochastic model.

Monte Carlo simulation can be a very powerful tool in the analysis of stochastic models. The theory of this method is sufficiently developed so that one obtains error estimates on statistics of the generated distribution of outcomes as compared with the "real" distribution. Surprisingly, this error is a function only of Q, the number of simulation runs, and not the number of random variables involved. This is an added bonus, especially in economic simulations, where one is often treating relatively large numbers of stochastic parameters.

Thus, if one makes a sufficiently large number of runs, the expected error of the statistics on the outputs may be reduced to any desired level.

However, the expected error of the standard deviation is proportional to $Q^{-1/2}$ and therefore reducing error can be a prohibitively expensive process if the run time of the simulation model is at all substantial. For example, at the 95 percent confidence level, 100 simulations produce standard deviations of outputs with 14 percent error. To reduce the error to 1.4 percent requires 10,000 simulations.[1]

It would then seem to be advantageous to search for another method which would produce comparable statistics with fewer simulations, and thus be more financially feasible in certain instances. One way in which this can be accomplished is through the construction of a response surface.

Response Surface Designs

If we are considering n input statistical variables, a response surface is a function, generally a polynomial, in n+1 space yielding an output variable as a function of the n input variables.

Since it is usually orders of magnitude faster to evaluate a polynomial than to make a simulation run, if one could construct a polynomial for each output variable of interest, and substitute evaluations of these polynomials for most of the simulations involved in establishing statistics on the outputs, one could save a considerable amount of computer time in many instances. (A criterion for deciding when it is advantageous to construct such a surface will be discussed later.)

It is worthwhile to note that the replacement of a simulation run with a response surface polynomial is a process of substituting a static model for a dynamic one, which leads to both advantages and disadvantages.

One obvious advantage is that optimization of one sort or another becomes much more feasible. Static optimization, even though very possibly non-linear, is a much less demanding task than optimization on a dynamic system.

However, one must also consider the major disadvantage--that the response surface adequately represents the simulation model at only one point in time. Therefore, shifting the time horizon of the model requires the construction of a new set of response surfaces.

However, given the desirability in many cases of constructing a response surface, how does one go about finding the required polynomials?

This problem is really twofold; one must generate data points to which a polynomial may be fitted, and then settle upon a method for fitting the polynomial. In many instances, when one is concerned with relating one output to two or three inputs, there are well-developed statistical methods for constructing experiments to obtain the data points to which a polynomial may be fitted. Full factorial, fractional factorial, Latin square, Youden square, Greco-Latin square, and split-plot designs may all be applicable at one time or another.

However, all of these designs may be overpowered by a large number of input statistical variables, as are often encountered in economic simulations. For example, to conduct a two-level full factorial experiment on a model with 10 random inputs would require $2^{10} = 1024$ trials, or simulation

runs. This is certainly no improvement on Monte Carlo methods. The experimental design itself then becomes a major problem in constructing a response surface when there are a large number of input variables.

There are also numerous approaches to fitting polynomials to the data points, once one has generated them. Statistical regression and the construction of an interpolating polynomial are two possible solutions to this problem. The number of statistical inputs and the number of data points may form a basis for the decision of which method to employ.

We will now proceed to describe the solutions that we adopted to the problems of experimental design and polynomial construction, in finding response surfaces for the model of the northern Nigerian beef industry.

The Nigerian Beef Model

For the purposes of this investigation the beef model was considered to have thirteen statistical inputs and eight outputs or performance criteria.

The thirteen random inputs were as follows:

- C1: acres freed of fly/£ expenditure on fly eradication
- C3: #TDN/acre-year on fly-free grazing
- C4: deterioration coefficient for range condition
- C5: #TDN/acre-year from food crop residues
- C6: #TDN/acre-year from cash crop residues
- C7: #TDN/acre-year in fly region
- C8: proportion of fly-infested land grazed during dry season
- C12: proportion of natural deaths which are marketed

C41: coefficient in birth rate function

C42: coefficient in death rate function

PAØ: normal animal price (in the absence of seasonal and secular factors)

AL1: rate of increase of demand for animals

AL2: rate of increase of crop land

The eight criterion variables were:

PFT = final female beef population (traditional)

PMT = final male beef population (traditional)

RCØN = range condition

FARMI = farm income

CF = discounted cash flow criterion function

FØREX = foreign exchange earnings

YA = income from sale of animals

YM = income from sale of milk

The probability distribution from which values for the input variables were drawn was chosen to be a "triangular" or "pseudo-beta" distribution. See figure 1.

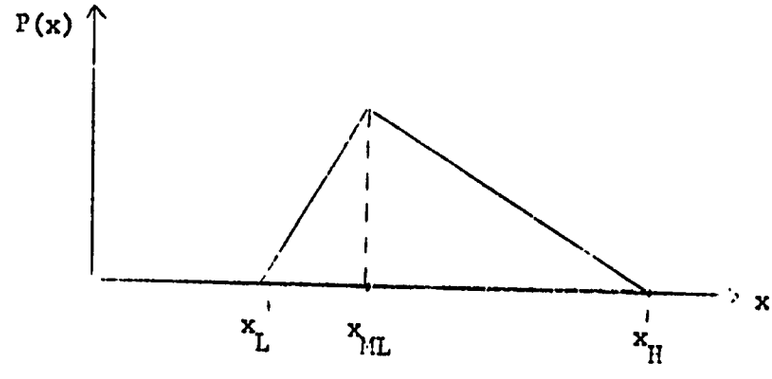


Figure 1. The "Triangular" or "Pseudo-beta" Distribution

Thus, to specify the probability density function (pdf) for any variable one needs three values,

x_L : the lower bound on the range of x

x_{ML}: the most likely value of x, (the mode)

x_U : the upper bound on x.

There are two reasons why this distribution was chosen: 1) Since the parameters could reasonably be expected to vary over only a finite range, it was desired to have a distribution with finite "tails," as opposed to the normal, for instance, and 2) the triangular type distribution was preferred to the beta distribution because it allows for a wider variance than the beta, and hence is more "pessimistic" about the behavior of the random variable under construction.

The three parameters of the distribution for each input variable were then specified (see Appendix A) and the distributions were sampled through the use of the random number generator on the computer.

For the purposes of making a set of simulation runs, a policy (initial conditions of the system) was set by specifying four parameters:

PFT_0 = the initial female beef population (traditional)

PMT_0 = the initial male beef population (traditional)

C18 = expenditures on tsetse-fly eradication

CSFT = a feedback parameter in sales of females (traditional).

It was determined to make a series of 200 Monte Carlo simulations for each of five different policies to serve as a standard by which to judge the results of our response surface investigations. The number 200 was chosen

because that is the number of simulations required to reduce the expected error of standard deviation on outputs to 10 percent at the 95 percent confidence level.[1]

At ten-run intervals, the means and standard deviations of the eight outputs were calculated and printed, in order to observe the rates of their convergence.

We were now ready to generate a surface to compare with our Monte Carlo results. Actually, since there are eight output variables involved, there are eight distinct surfaces to be constructed, each one yielding one output as a function of the thirteen statistical inputs and/or combinations thereof.

After investigating the possibilities of a structured experimental design, it was decided that since we were faced with such a large number of input statistical variables, most of the various factorial methods were not applicable. We then decided to use a series of random runs of the model as data points for the fitting of polynomials.

The next question was how to go about fitting a polynomial to these points; what variables to include, etc. As an initial attempt we decided to try the linear terms, their squares, and their cubes. No cross-product terms were included on this trial. Thus we had 39 independent variables, plus a constant term for a total of 40.

We took the data from 40 random runs of the model and fed it into the stepwise-addition linear regression routine, LSADD, available on the MSU computer.[2] This routine selects, one at a time, those independent variables which are most significant in "explaining" the variance of a dependent variable, until a preset stopping criterion is satisfied. The routine does

not calculate least-squares coefficients as it goes, as do many stepwise-addition routines.

Once this set of "significant" variables had been obtained, the coefficients for the polynomial were calculated as follows, using the Lagrangian approach:

If we let y_i denote the value of the output variable y on the ith trial, and $a_1 \ldots a_j$ the coefficients on the j statistical inputs, $x_{i1} \ldots x_{ij}$, where the i again indicates the ith trial, we may write:

$$
\begin{aligned}
y_1 &= a_1x_{11} + a_2x_{12} + \ldots + a_jx_{1j} \\
y_2 &= a_1x_{21} + a_2x_{22} + \ldots + a_jx_{2j} \\
&\vdots \\
y_i &= a_1x_{i1} + a_2x_{i2} + \ldots + a_jx_{ij} \\
&\vdots \\
y_j &= a_1x_{j1} + a_2x_{j2} + \ldots + a_jx_{jj}
\end{aligned}
$$

or in matrix notation:

$$
\begin{bmatrix} y_1 \\ \vdots \\ y_i \\ \vdots \\ y_j \end{bmatrix} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1j} \\ \vdots & & & \\ x_{i1} & x_{i2} & \cdots & x_{ij} \\ \vdots & & & \\ x_{j1} & x_{j2} & \cdots & x_{jj} \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ \vdots \\ a_i \\ \vdots \\ a_j \end{bmatrix}
$$

or $Y = XA$, which implies $A = X^{-1}Y$ as long as X is non-singular.

The reason for using this matrix inversion approach instead of directly calculating least-squares coefficients is that it is much faster to invert a matrix as long as the rank of the matrix remains within certain bounds. The inversion was done using a generalized matrix inversion routine, MATINV, for which the time required is proportional to n^3, where n is the number of rows in the matrix.[3]

When using this matrix inversion technique with random data points, it is possible that two of the rows of the matrix may be almost linearly dependent, or the matrix may be nearly singular. This leads to some very spurious coefficients for the response surface. The problem can usually be solved by using a slightly different set of random runs for data points.

Once the a's in the column matrix A are determined, the generation of the y's is simply a matter of inputting a set of values for the x's and multiplying.

For the purposes of comparison, the surface was subjected to the exact same set of inputs as the actual model for 200 runs and evaluated on an individual run basis by calculating total sum of squared errors for the eight output criteria:

$$TSS = \sum_{i=1}^{8} \left(\frac{y_i - \hat{y}_i}{y_i}\right)^2$$

where:

$\hat{y}_i$ = value of ith output of surface

y_i = value of ith output of simulation model

In addition, the means and standard deviations of all outputs over the 200 trials were calculated and compared to the same measures calculated on the outputs of the simulation model.

As mentioned before, the first attempt at specification of variables to be included in the set of possibly significant variables was to include the linear terms, their squares, and their cubes. The surface thus generated was highly unsatisfactory and it was decided to include paired interactions between linear terms also. Thus all terms of the form $x_i x_j$ became part of the set of possible independent variables.

This greatly improved the fit of the surfaces generated, and it is very possible the inclusion of higher order interactions may be beneficial also, but a problem arises in evaluating this set of possible independent variables with the statistical routines on the computer. Since we had 13 distinct input statistical variables, the inclusion of paired interactions added $\binom{13}{2} = 78$ variables to the set of possible independent variables. Thus one soon pushes the limits of the capacity of the LSADD routine, which is about 120 variables.

Evaluation of Results

Since the objective of constructing response surfaces is to reduce computer time while retaining as much as possible the ability to evaluate the effects of random variations in parameters of the model, we should now evaluate the ability of our surface to do just that.

Essentially, by making a series of Monte Carlo runs with the simulation model itself, we obtain three kinds of useful information: 1) we can display

the variation in outputs caused by random variation in parameters; 2) we can display the change in output statistics induced by changing the distribution function of an input; and 3) we can evaluate the effects of different policies on the output means and standard deviations.

These first two functions are a comparison of one group of runs with another under the same policy (or initial condition). In order to make this same type of comparison with a response surface, it is important that the surface "preserve order of runs." That is to say, if a given output criterion is viewed as a random variable, and the set of values of this variable generated by a series of simulation runs as a sample from this random variable, then the order statistics must be preserved between the sample obtained from the model and the sample obtained from the surface. If the jth member of the sample from the model constitutes the ith order statistic of that sample, the jth member of the sample from the responce surface must also be the ith order statistic of that sample.

The surfaces we constructed proved incapable of reliably preserving order in this sense. Some surfaces did fairly well; others failed miserably. Our method of surface construction seems to fall somewhat short in this respect.

However, perhaps the most important information is the effect of different policies on the output means and standard deviations. The results of the surface in this respect are very encouraging. The surface is capable of very reliably exhibiting the effects of different policies. (See tabulated results in Appendix B.)

The next question is, "How much computer time can we save?"

In general, we may express the time requirement for a series Carlo runs as:

$$T_1(r) = s + xr$$

where:

$T_1 \equiv$ time

$s \equiv$ setup time (compilation, loading, etc.)

$x \equiv$ run time of model

$r \equiv$ number of runs

Typically, the setup time will be about 25-30 seconds for a large FORTRAN model, and the run time may vary from less than a second to several minutes.

For the Nigerian beef model, $x \approx 2.6$ secs. Thus, 200 runs requires slightly over 9 minutes of time on the CDC 3600.

On the other hand, to obtain the same information with a response surface, one must find "significant" independent variables using the LSADD routine, and then calculate coefficients and evaluate polynomials, which we did with a program called SRFCE. (See listing in Appendix C.)

The time required to find significant variables on LSADD is primarily a function of the number of outputs, y, and thus the number of times it must go through the list of possible independent variables. As a fairly good approximation we may write:

$$T_2(y) \approx 70 + 10y$$

given a 40 run sample for data points. The time is also dependent upon the number of data points to be considered, but this number is generally constant for a given application and thus its effect is absorbed into the coefficient on y. It is interesting to note that the time for execution of LSADD appears to be relatively insensitive to the total number of independent variables specified. Intuitively, it would seem that there should be some dependence here, but its effect seems to be completely swamped out by the dependence on the number of dependent variables which must be regressed against this independent set. It should also be noted, however, that our tests did not incorporate a very wide range of numbers of independent variables, and so even if some relationship is in effect, it may not have been apparent from our results. Thus 8 outputs requires about 2:30 of computer time.

The time requirement for SRFCE is again largely a function of the number of runs, r, and may be approximated as:

$$T_3(r) \simeq 17 + .06r$$

Thus 200 runs require about 29 seconds. We see that in the case of the Nigerian beef model, the Monte Carlo runs required 9:07 of computer time, while the surface approach required 3:10 for 40 runs of the model, plus 2:30 for LSADD, and :29 for SRFCE, a total of 5:09, a considerable saving.

It should be obvious that as the model run time increases the potential for saving time becomes greater and greater, and conversely, as the run time decreases, one stands to gain much less. Indeed, if the run time is small enough, it would even be faster to do Monte Carlo analysis.

It would then seem reasonable to calculate a "break-even" point for run time of the model as one criterion for deciding whether or not response surface techniques may be helpful.

Let us then attempt to express $\hat{x}$, the "break-even" run as a function of r, the number of runs desired, y, the number of outputs, and r', the number of data points to be generated:

$$\hat{x} = f(r,y,r')$$

Equating total times:

$$T_1(r) = T_1(r') + T_2(y) + T_3(r)$$

or $$s + \hat{x}r = s + r'\hat{x} + 70 + 10y + 17 + .06r$$

which implies $$\hat{x}(r-r') = 87 + 10y + .06r$$

which yields $$\hat{x} = \frac{87 + 10y + .06r}{r-r'}$$

Thus, for r = 200, y = 8, and r' = 40, $\hat{x}$ = 1.05 sec/run

Suggestions for Further Work

Almost certainly the weakest area of this investigation was the method used to obtain data points for the LSADD routine. There is certainly a need for much effort in trying to arrive at an experimental design more suitable than 40 random points.

The use of the LSADD routine itself may be less than an optimal way of arriving at a set of "significant" independent variables. The routine is quite time consuming and is limited to consideration of about 120 candidates for independent variables. For larger models, this size restriction is a very real constraint.

As it now stands, our method requires the construction of a completely new surface for each policy one wishes to evaluate. If the policy variables themselves could be incorporated into the surface, this would become unnecessary, and the response surface would be much better adapted to comparisons of policies than the Monte Carlo method, which requires a whole new set of runs for each policy specification.

APPENDIX A

Data used for specifying the probability density functions for the statistical inputs:

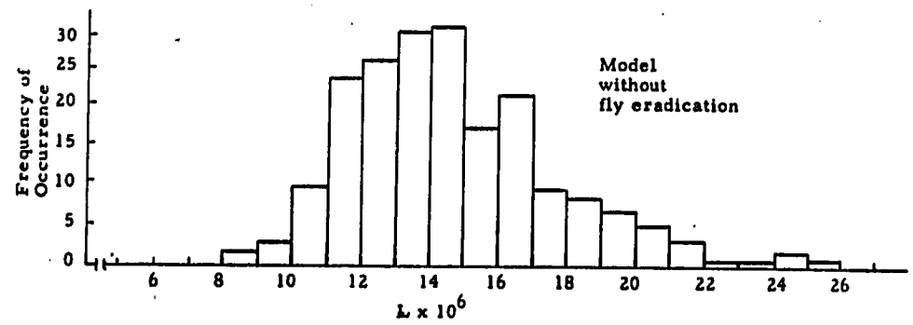
Variable	lower bound	mode	upper bound
C1	.15	.4	.4
C3	130.	130.	248.
C4	0.	0.	.009
C5	150.	325.	350.
C6	40.	40.	100.
C7	200.	200.	500.
C8	.1	.1	.4
C12	.1	.2	.3
C41	.71	.94	1.4
C42	.77	.97	1.5
PAO	12.	15.	18.
AL1	.02	.03	.04
AL2	.02	.035	.05

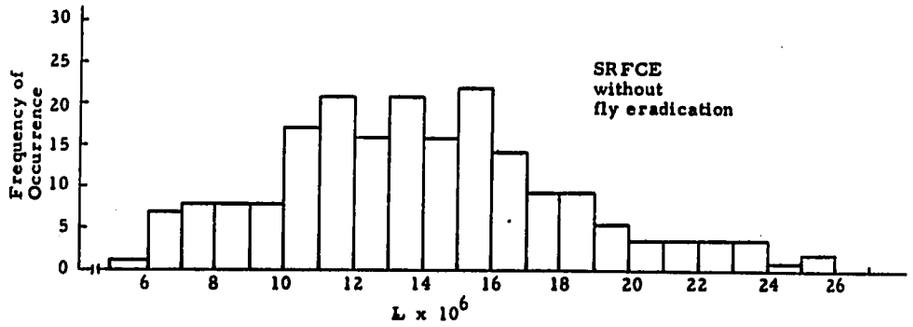
Policies:	PFT	PMT	C18	CSFT
1)	5100.	2400.	0.0	0.0
2)	5100.	2400.	0.0	1.0
3)	5100.	2400.	1000.	0.0
4)	5100.	2400.	1000.	1.0
5)	3825.	1800.	0.0	1.0

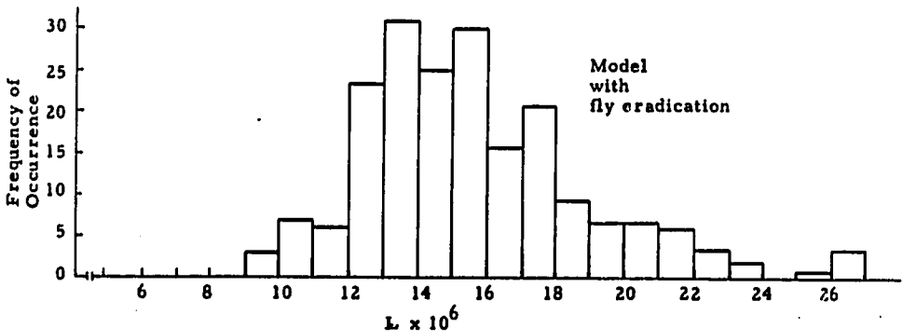
			PFT	PMT	RCON	FARMI	CF	FOREX	YA	YM
1.	MEAN	Model	3835.	1832.	.7814	119400.	1196000.	98420.	14590.	28660.
		Surface	3759.	1828.	.7799	119500.	1205000.	98820.	13740.	28210.
	ST DEV	Model	780.9	379.0	.1557	7437.	53290.	4981.	2974.	4823.
		Surface	808.1	388.5	.1461	7114.	52510.	6026.	4064.	4987.
2.	MEAN	Model	3029.	1591.	.8494	114500.	1141000.	99370.	14720.	23680.
		Surface	3012.	1583.	.8471	114200.	1143000.	103300.	13840.	23810.
	ST DEV	Model	469.2	279.0	.1172	5363.	49130.	5355.	3082.	3069.
		Surface	477.1	283.7	.1130	5317.	51370.	11130.	4272.	2917.
3.	MEAN	Model	4155.	1991.	.8176	122200.	1192000.	99510.	15400.	30650.
		Surface	4060.	2007.	.7482	122300.	1191000.	98360.	15120.	30060.
	ST DEV	Model	822.9	401.3	.1330	7839.	53980.	5111.	3131.	5082.
		Surface	1497.	398.5	.1415	7989.	56110.	15810.	3996.	9110.
4.	MEAN	Model	3412.	1771.	.8875	117800.	1139000.	100500.	15610.	26040.
		Surface	3417.	1754.	.8737	109500.	1133000.	108900.	16120.	26010.
	ST DEV	Model	526.8	295.5	.09768	5667.	50060.	5574.	3288.	3253.
		Surface	547.0	499.2	.09832	16740.	48000.	13520.	4367.	3352.
5.	MEAN	Model	3035.	1594.	.8569	114600.	1108000.	99410.	14750.	23710.
		Surface	3055.	1585.	.8740	105900.	1121000.	103100.	13950.	23850.
	ST DEV	Model	496.7	279.1	.1133	5365.	47640.	5385.	3086.	3067.
		Surface	472.5	285.4	.1101	19440.	47620.	10550.	4132.	2908.

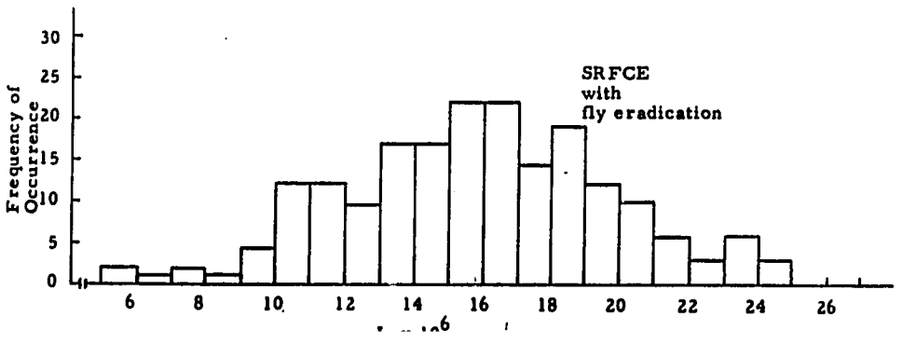
COMPARISON OF OUTPUT STATISTICS OF SIMULATION MODEL
AND RESPONSE SURFACE BASED ON 200 RUNS

Frequency Plots of Income from Sale of Animals (YA).









APPENDIX C

```
      PROGRAM MCSIM
      DIMENSION X(13,3),XVAL(13),PFT1(200),PMT1(200),RCON(200),
     1      FARMI(200),CF(200),FOREX(200),YA(200),YM(200)
      KSWTCH = 0
      NR = 200
      NP = 1
   11 IREAD = 0
      IPRNT = 0
      DO 30 I=1,NP
      NRC = 0
      READ 1000, C18,PMT,PFT,CSFT
      PRINT 1003, PFT,PMT,C18,CSFT
      IF(KSWTCH.EQ.1) GO TO 1
      IF(IREAD.EQ.0) READ 1001, ((X(K,J),J=1,3),K=1,13)
      IREAD = 1
    1 DO 20 II=1,NR
      IF(KSWTCH.EQ.1) GO TO 3
      DO 2 L=1,13
    2 CALL BETA (XVAL(L),X(L,1),X(L,2),X(L,3))
      C1=XVAL(1)
      PAO = XVAL(2)
      C3=XVAL(3)
      C4=XVAL(4)
      C5=XVAL(5)
      C6=XVAL(6)
      C7=XVAL(7)
      C8=XVAL(8)
      C12=XVAL(9)
      C41=XVAL(10)
      C42=XVAL(11)
      AL1=XVAL(12)
      AL2=XVAL(13)
      PUNCH 1002, II,C1,PAO,C3,C4,C5,C6,C7
      PUNCH 1002, II,C8,C12,C41,C42,AL1,AL2
      GO TO 4
    3 READ 1000, C1,PAO,C3,C4,C5,C6,C7,C8,C12,C41,C42,AL1,AL2
    4 CALL BEEF(C1,PAO,C3,C4,C5,C6,C7,C8,C12,C41,C42,AL1,AL2,PFT,PMT,
     1     CSFT,C18,PFT1(II),PMT1(II),RCON(II),FARMI(II),CF(II),
     2     FOREX(II),YA(II),YM(II),II,NR)
      NRC = NRC + 1
      IF(KSWTCH .EQ. 1) GO TO 30
      PUNCH 1002,     PFT1(II),PMT1(II),RCON(II),FARMI(II),CF(II),
     1     FOREX(II),YA(II),YM(II)
      IPRNT = IPRNT + 1
      IF(IPRNT .NE. 10) GO TO 20
      PFTM=XMEAN(PFT1,NRC)
      PMTM=XMEAN(PMT1,NRC)
      RCONM=XMEAN(RCON,NRC)
      FARMIM=XMEAN(FARMI,NRC)
      CFM=XMEAN(CF,NRC)
      FOREXM=XMEAN(FOREX,NRC)
      YAM=XMEAN(YA,NRC)
      YMM=XMEAN(YM,NRC)
      PFTS = SQRT(VAR(PFT1,PFTM,NRC))
      PMTS = SQRT(VAR(PMT1,PMTM,NRC))
      RCONS = SQRT(VAR(RCON,RCONM,NRC))
```

```
      FARMIS = SQRT(VAR(FARMI,FARMIM,NRC))
      CFS = SQRT(VAR(CF,CFM,NRC))
      FOREXS = SQRT(VAR(FOREX,FOREXM,NRC))
      YAS = SQRT(VAR(YA,YAM,NRC))
      YMS = SQRT(VAR(YM,YMM,NRC))
      PRINT 1004
      PRINT 1005, PFTM,PMTM,RCONM,FARMIM,CFM,FOREXM,YAM,YMM,PFTS,
     1      PMTS,RCONS,FARMIS,CFS,FOREXS,YAS,YMS
      IPRNT = 0
   20 CONTINUE
   30 CONTINUE
 1000 FORMAT(8F10.3)
 1001 FORMAT(3F10.3)
 1002 FORMAT(8E10.3)
 1003 FORMAT(*1PFT = *,E10.3,3X,*PMT = *,E10.3,3X,*C18 = *,E10.3,3X,
     1    *C3FT = *,E10.3)
 1004 FORMAT(*0*,4X,*PFTM*,9X,*PMTM*,9X,*RCONM*,8X,*FARMIM*,7X,*CFM*,
     1    10X,*FOREXM*,7X,*YAM*,10X,*YMM*/5X,*PFTS*,9X,*PMTS*,9X,
     2    *RCONS*,8X,*FARMIS*,7X,*CFS*,10X,*FOREXS*,7X,*YAS*,10X,*YMS*)
 1005 FORMAT(1H0,8(E10.3,3X))
   31 END
```

```
      SUBROUTINE BEEF(C1,PAO,C3,C4,C5,C6,C7,C8,C12,C41,C42,AL1,AL2,
     1   PFT1,PMT1,CSFT,C18,PFT,PMT,RCON,FARMI,CF,FOREX,YA,YM,II,LR)
      REAL LGF,LG,LC1,LC2,LC3,LGM,LGT
      DIMENSION VAL1(9),VAL2(9),VAL3(9),VAL4(9),CROUT1(3),CROUT2(3),
     1     CROUT3(3),VAL5(9)
C CONSTANTS
      NR=1
      DT=.1
      RLENTH = 20.
      NCPP=10
      NITOL=RLENTH/(NCPP*DT)
      INTRVL = 5
      DI=850.
      AL3=.06
      AL4=.02
      LC10=8500.
      C9 = 263.
      C10 = 3700.
      C13 = 0.
      C17 = 0.
      C19 = 100.
      C20 = .023
      C21 = .29
      C22 = .47
      C23 = .3
      C24 = .6
      C25 = .57
      C26 = .7
      C29=60.
      C30=.035
      C31=42.
      C32=.0056
      C33=0.
      C34=0.
      C35 = .208
      ELAS1=1.
      ELAS2=1.
      GRDEPD=10.
      TDNAD=3000.
      LG = 37000.
      RLTT = 0.
      ISPRS = 0
      GRE = 8.0
C
C RUN LOOP
      DO 22 M=1,NR
C
      C2=5.75
      C11=1.15
      C14= .036
      C15= 17.
      C16=.000333
      C27=1.5
      C28=1.5
      C36=.1
      C37=.285
```

```
      C38=.100
      C39=.000
      C40M=.6
      C40T=.6
      GRGDEL=3.
      FEGDEL=3.
      PAD=3.
      VAL1(1)=.06
      VAL1(2)=.19
      VAL1(3)=.27
      VAL1(4)=.33
      VAL2(1)=.08
      VAL2(2)=.29
      VAL2(3)=.44
      VAL2(4)=.54
      VAL3(1)=.55
      VAL3(2)=.22
      VAL3(3)=.13
      VAL3(4)=.11
      VAL3(5)=.1
      VAL3(6)=.09
      VAL3(7)=.08
      VAL4(1)=.5
      VAL4(2)=.17
      VAL4(3)=.1
      VAL4(4)=.08
      VAL4(5)=.07
      VAL4(6)=.06
      VAL4(7)=.05
      VAL5(1)=1.
      VAL5(2)=1.
      CROUT1(1)=0.
      CROUT1(2)=0.
      CROUT1(3)=0.
      CROUT2(1)=0.
      CROUT2(2)=0.
      CROUT2(3)=0.
      CROUT3(1)=0.
      CROUT3(2)=0.
      CROUT3(3)=0.
      LGF=74000.
      D1=1.
      D2=3.
      D3=1.
      D4=1.
      D5=3.
      PFCAT=.65
      PFCAM=.65
      YMAT=950.
      YMAM=1200.
      PRMT=.01
      PRMM=.01
      TDNTAR = 2700.
C INITIAL CONDITIONS
      PA = 15.0
      PMT = PMT1
```

```
      PFT = PFT1
      RCON=1.
      CF=0.
      CF1=0.
      CF2=0.
      LGM=1000.
      TDNM=C9*LGM
      PFM=36.5
      PMM=13.5
      A2T=.25*PFT
      A1PT=0.
      DFT=.1*PFT
      DMT=.1*C40T*PMT
      A2M=.41*PFM
      A1PM=0.
      ERM=.2
      DFM=.07*PFM
      DMM=.07*PMM
      ERT=.08
      LC1=8500.
      LC2=3000.
      LC3=0.
      LC20=3000.
      T=0.
      FARMIA=0.
      FOREXA=0.
      SFT=C20*PFT
      SFM=0.
      GRCAP=0.
      TDNAM = 2400.
      TDNT = 14000000.
 1075 DO 224 I=1,NITOL
      DO 236 J=1,NCPP
      T=T+DT
C POLICY VARIABLES
  181 RLC2=C17+LC20*AL4*EXP(AL4*T)
  182 RLC3=RLTT-C17
  183 EFE=C18
  184 EXGR=C19
  185 SFT=PFT*(C20+ELAS1*SFT*(PA-PAO)/(PFT*PA))+(CSFT*(PFT-TDNT/((1+C22)
     1*TDNTAR)))
  186 SMT=C21*(PMT-C22*PFT)+C35*PMT
  187 IF (SMT) 188,188,190
  188 SMT=0.
  190 SFM=C23*(PFM-TDNM*C16*C26)+.5*A2M-DFM+ELAS2*SFM*(PA-PAO)/PA
 1900 IF (SFM) 1901,1901,191
 1901 SFM=0.
  191 SMM=C24*(PMM-C25*PFM)+.5*A2M-DMM
  192 IF (SMM) 193,193,194
  193 SMM=0.
  194 PA=PAO*(1.+C33*T+C34*SIN(6.2816*T))
 1941 PAA=.8*PA
C  MODEL STRUCTURE
  195 LC3=LC3+DT*RLC3
  196 AUX3=EXGR*C2
  197 CALL DELAY(AUX3,AUX4,CROUT2,GRGDEL,DT,3)
```

```
 1971 RTDN=AJX4*C9+RLC3*C10
 1972 RAA=RTDN*C16+C36*(TDNAD)
 1973 RFTT=RAA*(PFT/(PFT+PMT))
 1974 RMTT=RAA-RFTT
 1975 RFTM=-RFTT
 1976 RMTM=-RMTT
      AUX1 = EFE*C1*C7/C3
  199 CALL DELAY(AUX1,AUX2,CROUT1,FEGDEL,DT,3)
  200 DEM=DI*EXP(AL1*T)
  201 RLC1=AL2*LC10*EXP(AL2*T)
  202 LG=LG+DT*(-RLC1-RLC2-RLC3+AUX2)
  203 LGM=LGM+DT*AUX4
  204 LGT=LG-LGM
  205 GRT=LGT/(PFT+PMT)
  206 RCON=RCON+DT*C4*(GRT-GRE)
 2061 IF(RCON.LT..1) RCON=.1
  207 TDNGT=RCON*C3*LGT
  208 TDNRES=C5*LC1+C6*LC2
  209 TDNT=TDNGT+TDNRES+C7*C8*LGF
  210 TDNAT=TDNT/(PFT+PMT)
  211 TDNGM=LGM*C9
  212 TDNFC=LC3*C10
  213 TDNM=TDNGM+TDNFC
  214 TDNAM=TDNM/(PFM+PMM)
  215 CALL DEMOG(SFT,SMT,TDNAT,PFT,PMT,ERT,VAL1,680.,680.,3,VAL3,680.,34
     10.,6,D1,D2,D3,D4,D5,DMT,DFT,A2T,A1PT,RFTT,RMTT,DT,BRT,DRT,ERPT,A1T
     2,BRDELT,A3T,A4T,VAL5,QMT,PFCAT,YMAT,PRMT,YMT,BFT,BMT,C40T,C41,C42)
  216 CALL DEMOG(SFM,SMM,TDNAM,PFM,PMM,ERM,VAL2,680.,680.,3,VAL4,680.,34
     10.,6,D1,D2,D3,D4,D5,DMM,DFM,A2M,A1PM,RFTM,RMTM,DT,BRM,DRM,ERPM,A1M
     2,BRDELM,A3M,A4M,VAL5,QMM,PFCAM,YMAM,PRMM,YMM,BFM,BMM,C40M,C41,C42)
 2161 SRT=PMT/PFT
 2162 SRM=PMM/PFM
 2163 TMM=2*SRM/(BRM-2*DRM*SRM)
 2164 TFM=2/(BRM-2*DRM)
  217 SUPT=SFT+SMT+C12*(DFT+DMT)
 2171 SUPM=C11*(SFM+SMM)+C13*(DFM+DMM)
 2172 SUP=SUPT+SUPM
  218 CIMP=DEM-SUP
 2181 YAT=SUPT*PA*PAD*(DMT+DFT)
 2182 YAM=SUPM*PA*PAD*(DMM+DFM)
  219 YA=YAT+YAM
 2191 YM=YMT+YMM
 2192 QM=QMT+QMM
  220 COGR=LGM*C14
  225 LC1=LC1+DT*RLC1
  221 LC2=LC2+DT*RLC2
  223 YCC=C15*LC2
 2231 CALL DELAY(EXGR,GRDEP,CROUT3,GRDEPD,DT,3)
  226 CF=CF+DT*((YA+YCC+YM-COGR-EFE-GRDEP*EXGR)*EXP(-AL3*T))
 2261 CF1=CF1+DT*((YAM+YMM)*EXP(-.06*T))
 2262 CF2=YM
 2263 CFM=C36*CF1+C37*CF2
  227 CO=EFE+COGR
  228 CAPE=EXGR
  229 FARMI=YA+YM+YCC
  230 FARMIA=FARMIA+DT*FARMI
```

```
      SUBROUTINE SUMMARY (MN,NR,T,     W1,W2,W3,W4,W5,W6,W7,W8,W9,W10,
     1  W11,W12,Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8,Q9,Q10,Q11,Q12,I1,I2,I3,I4,I5,
     2  I6,I7,I8,I9,I10,I11,I12,J1,J2,J3,J4,J5,J6,J7,J8,J9,J10,J11,J12,
     3  NV,NP)
      DIMENSION IM(30),V1(30),V2(30),V3(30),V4(30),V5(30),V6(30),V7(30),
     1  V8(30),V9(30),V10(30),V11(30),V12(30),P1(30),P2(30),P3(30),
     2  P4(30),P5(30),P6(30),P7(30),P8(30),P9(30),P10(30),P11(30),
     3  P12(30)
      M1 = (MN-1)/10
      M = MN-10*M1
      IM(M)=MN
      V1(M)=W1
      V2(M)=W2
      V3(M)=W3
      V4(M)=W4
      V5(M)=W5
      V6(M)=W6
      V7(M)=W7
      V8(M)=W8
      V9(M)=W9
      V10(M)=W10
      V11(M)=W11
      V12(M)=W12
      P1(M)=Q1
      P2(M)=Q2
      P3(M)=Q3
      P4(M)=Q4
      P5(M)=Q5
      P6(M)=Q6
      P7(M)=Q7
      P8(M)=Q8
      P9(M)=Q9
      P10(M)=Q10
      P11(M)=Q11
      P12(M)=Q12
      IF(MN .EQ. NR) GO TO 9
      IF(M .NE. 10) RETURN
    9 PRINT 1, T
      GO TO(10,10,11,12,13,14,15,16,17,18,19,20)NV
   10 PRINT 2,I1,I2                                          $GO TO 21
   11 PRINT 2,I1,I2 ,I3                                      $GO TO 21
   12 PRINT 2,I1,I2 ,I3,I4                                   $GO TO 21
   13 PRINT 2,I1,I2 ,I3,I4,I5                                $GO TO 21
   14 PRINT 2,I1,I2 ,I3,I4,I5,I6                             $GO TO 21
   15 PRINT 2,I1,I2 ,I3,I4,I5,I6,I7                          $GO TO 21
   16 PRINT 2,I1,I2 ,I3,I4,I5,I6,I7,I8                       $GO TO 21
   17 PRINT 2,I1,I2 ,I3,I4,I5,I6,I7,I8,I9                    $GO TO 21
   18 PRINT 2,I1,I2 ,I3,I4,I5,I6,I7,I8,I9,I10                $GO TO 21
   19 PRINT 2,I1,I2 ,I3,I4,I5,I6,I7,I8,I9,I10,I11            $GO TO 21
   20 PRINT 2,I1,I2 ,I3,I4,I5,I6,I7,I8,I9,I10,I11,I12
   21 GO TO(22,22,23,24,25,26,27,28,29,30,31,32)NP
   22 PRINT 3,J1,J2                                          $GO TO 33
   23 PRINT 3,J1,J2 ,J3                                      $GO TO 33
   24 PRINT 3,J1,J2 ,J3,J4                                   $GO TO 33
   25 PRINT 3,J1,J2 ,J3,J4,J5                                $GO TO 33
   26 PRINT 3,J1,J2 ,J3,J4,J5,J6                             $GO TO 33
```

```
 27 PRINT 3,J1,J2 ,J3,J4,J5,J6,J7                                  $GO TO 3
 28 PRINT 3,J1,J2 ,J3,J4,J5,J6,J7,J8                               $GO TO 3
 29 PRINT 3,J1,J2 ,J3,J4,J5,J6,J7,J8,J9                            $GO TO 3
 30 PRINT 3,J1,J2 ,J3,J4,J5,J6,J7,J8,J9,J10                        $GO TO 3
 31 PRINT 3,J1,J2 ,J3,J4,J5,J6,J7,J8,J9,J10,J11                    $GO TO 3
 32 PRINT 3,J1,J2 ,J3,J4,J5,J6,J7,J8,J9,J10,J11,J12
 33 DO 60 I=1,M
    GO TO(34,34,35,36,37,38,39,40,41,42,43,44)NV
 34 PRINT 4,IM(I),V1(I),V2(I)                                      $GO TO 4
 35 PRINT 4,IM(I),V1(I),V2(I),V3(I)                                $GO TO 4
 36 PRINT 4,IM(I),V1(I),V2(I),V3(I),V4(I)                          $GO TO 4
 37 PRINT 4,IM(I),V1(I),V2(I),V3(I),V4(I),V5(I)                    $GO TO 4
 38 PRINT 4,IM(I),V1(I),V2(I),V3(I),V4(I),V5(I),V6(I)              $GO TO 4
 39 PRINT 4,IM(I),V1(I),V2(I),V3(I),V4(I),V5(I),V6(I),
   1  V7(I)                                                        $GO TO 4
 40 PRINT 4,IM(I),V1(I),V2(I),V3(I),V4(I),V5(I),V6(I),
   1  V7(I),V8(I)                                                  $GO TO 4
 41 PRINT 4,IM(I),V1(I),V2(I),V3(I),V4(I),V5(I),V6(I),
   1  V7(I),V8(I),V9(I)                                            $GO TO 4
 42 PRINT 4,IM(I),V1(I),V2(I),V3(I),V4(I),V5(I),V6(I),
   1  V7(I),V8(I),V9(I),V10(I)                                     $GO TO 4
 43 PRINT 4,IM(I),V1(I),V2(I),V3(I),V4(I),V5(I),V6(I),
   1  V7(I),V8(I),V9(I),V10(I),V11(I)                              $GO TO 4
 44 PRINT 4,IM(I),V1(I),V2(I),V3(I),V4(I),V5(I),V6(I),
   1  V7(I),V8(I),V9(I),V10(I),V11(I),V12(I)
 45 GO TO(46,46,47,48,49,50,51,52,53,54,55,56)NP
 46 PRINT 5,P1(I),P2(I)                                            $GO TO 60
 47 PRINT 5,P1(I),P2(I),P3(I)                                      $GO TO 60
 48 PRINT 5,P1(I),P2(I),P3(I),P4(I)                                $GO TO 60
 49 PRINT 5,P1(I),P2(I),P3(I),P4(I),P5(I)                          $GO TO 60
 50 PRINT 5,P1(I),P2(I),P3(I),P4(I),P5(I),P6(I)                    $GO TO 60
 51 PRINT 5,P1(I),P2(I),P3(I),P4(I),P5(I),P6(I),P7(I)              $GO TO 60
 52 PRINT 5,P1(I),P2(I),P3(I),P4(I),P5(I),P6(I),P7(I),
   1  P8(I)                                                        $GO TO 60
 53 PRINT 5,P1(I),P2(I),P3(I),P4(I),P5(I),P6(I),P7(I),
   1  P8(I),P9(I)                                                  $GO TO 60
 54 PRINT 5,P1(I),P2(I),P3(I),P4(I),P5(I),P6(I),P7(I),
   1  P8(I),P9(I),P10(I)                                           $GO TO 60
 55 PRINT 5,P1(I),P2(I),P3(I),P4(I),P5(I),P6(I),P7(I),
   1  P8(I),P9(I),P10(I) ,P11(I)                                   $GO TO 60
 56 PRINT 5,P1(I),P2(I),P3(I),P4(I),P5(I),P6(I),P7(I),
   1  P8(I),P9(I),P10(I) ,P11(I) ,P12(I)
 60 CONTINUE
C
  1 FORMAT(1H0,24X*SUMMARY OF OUTPUT FOR KEY VARIABLES,AT TIME T = *,
   1  F4.1,* YEARS*/)
  2 FORMAT(1H0,*RUN*,2XA8,11(3XA8))
  3 FORMAT(6XA8,11(3XA8))
  4 FORMAT(1H0,I3,1XE10.3,11(1XE10.3))
  5 FORMAT(5XE10.3,11(1X,E10.3))
    RETURN
    END
```

```
  231 FOREX=C27*YCC-C28*CIMP*PA
  232 FOREXA=FOREXA+DT*(FOREX)
  233 ANPROT=C29*SUP+C30*QM
  234 CERPROT=C31*LC1
  235 GRCAP=GRCAP+DT*(EXGR-GRDEP)
  236 VALCAP=(PFT+PMT+C11*(PFM+PMM))*PAA+GRCAP
C
C     PREPARE SUMMARY TABLE
 1392 IF(I.NE.NITOL) GO TO 224
      CALL SUMMARY (II,LR,RLENTH,PFT,PMT,RCON,FARMI,CF,FOREX,
     1 YA,YM,C1,PA,TDNAT,PFM,C3,C4,C5,C6,C7,C8,C12,C41,C42,AL1,AL2,PFCAM
     2,     8HPFT     ,8HPMT     ,8HRCON    ,8HFARMI   ,8HCF      ,
     3      8HFOREX   ,8HYA      ,8HYM      ,8HC1      ,8HPA      ,
     4      8HTDNAT   ,8HPFM     ,8HC3      ,8HC4      ,8HC5      ,
     5      8HC6      ,8HC7      ,8HC8      ,8HC12     ,8HC41     ,
     6      8HC42     ,8HAL1     ,8HAL2     ,8HPFCAM   ,11,11)
  224 CONTINUE
C
   22 CONTINUE
C
      RETURN
      END
```



```
      SUBROUTINE DEMOG(SF,SM,TDNA,PF,PM,ER,VALB,SMALLB,DIFFB,KB,VALD,SMA
     1LLD,DIFFD,KD,D1,D2,D3,D4,D5,DM,DF,A2,A1P,RFT,RMT,DT,BR,DR,ERP,A1,B
     2RDEL,A3,A4,VAL5,QM,PFCA,YMA,PRM,YM,BF,BM,C40,C41,C42)
      DIMENSION VALB(9),VALD(9),VAL5(9)
    1 BR=C41*TABLIE(VALB,SMALLB,DIFFB,KB,TDNA)
    2 DR=C42*TABLIE(VALD,SMALLD,DIFFD,KD,TDNA)
    3 ERP=PF*BR/(PF+PM)-DR
    4 A1=BR*PF
    5 A1P=A1P+(DT/.3)*(A1-A1P)
    6 IF (A1-A1P) 7,7,9
    7 BRDEL=D1
    8 GOTO 10
    9 BRDEL=D2
   10 A2=A2+(DT/BRDEL)*(A1-A2)
   11 BF=.5*A2
   12 BM=BF
   13 A3=PF*DR
   14 DF=DF+(DT/D3)*(A3-DF)
   15 A4=PM*DR *C40
   16 DM=DM+(DT/D4)*(A4-DM)
   17 ER=ER+(DT/D5)*(ERP-ER)
   18 PF=PF+DT*(BF-DF-SF-RFT)
   19 PM=PM+DT*(BM-DM-SM-RMT)
   20 QM=PF*PFCA*YMA*TABLIE(VAL5,1360.,1360.,1,TDNA)
   21 YM=QM*PRM
   22 RETURN
      END
```

FOR1.3A 08/21/70

```
      SUBROUTINE DELAY(RINR,ROUTR,CROUTR,DEL,DT,K)
      DIMENSION CROUTR(3)
      DEL1 = DEL/(FLOAT(K)*DT)
      RIN = RINR
      DO 1 I = 1,K
      ABC = CROUTR(I)
      CROUTR(I) = ABC + (RIN-ABC)/DEL1
    1 RIN = ABC
      ROUTR = CROUTR(K)
      RETURN
      END
```

FOR1.3A 08/21/70

```
      FUNCTION TABLIE(VAL,SMALL,DIFF,K,DUMMY)
      DIMENSION VAL(9)
      DUM=AMIN1(AMAX1(DUMMY-SMALL,0.0),FLOAT(K)*DIFF)
      I=1.0+DUM/DIFF
      IF(I.EQ.K+1) I=K
      TABLIE=(VAL(I+1)-VAL(I))*(DUM-FLOAT(I-1)*
     1DIFF)/DIFF+VAL(I)
      RETURN
      END
```

```
      FUNCTION XMEAN (A,NR)
      DIMENSION A(200)
      ASUM=0.
      DO 1 J=1,NR
    1 ASUM=ASUM + A(J)
      XMEAN = ASUM/FLOAT(NR)
      RETURN
      END
```



```
      FUNCTION VAR (A,AM,NR)
      DIMENSION A(200)
      VSUM=0.
      DO 1 J=1,NR
    1 VSUM = VSUM + ((ABS(A(J)-AM))**2.)
      R = FLOAT(NR)
      VAR= VSUM/(R-1.)
      RETURN
      END
```

```
      SUBROUTINE BETA (X, XL, XML, XH)
        B = XH-XL
        R = RANF(-1)
      IF ( R .GT. (XML-XL)/B )  GO TO 10
        X = XL + SQRT( R*B*(XML-XL) )
        RETURN
   10 X = XH - SQRT( (1.-R)*B*(XH-XML) )
      RETURN
      END
```


EXECUTION STARTED AT 1419 -47

```
      PROGRAM SRFCE
      COMMON /BLK1/ XIN(40,13), XOUT(40,40)
      COMMON /BLK2/ VARIN(13), VAROUT(40,8)
      DIMENSION COEF(8,40), NAMVAR(40), NAMOUT(8), T1(40), IT2(40),
     1          IT3(40,2), NV(8), OUTPUT(8), OUT(8), Y(40,8),
     2          RUNSUM(8), XMEAN(8), VSUM(8), STDDEV(8)
      ISWTCH = 0
      JSWTCH = 0
      KSWTCH = 1
      IF(KSWTCH .EQ. 1) GO TO 12
      READ 1004, NOBS
      READ 1002, (NAMOUT(I),I=1,8)
      DO 1 L=1,NOBS
      READ 1001, (XIN(L,J),J=1,13)
    1 READ 1005, (Y(L,K),K=1,8)
      DO 10 I=1,8
      READ 1004, NV(I)
      NVAR = NV(I)
      READ 1002, (NAMVAR(K),K=1,NVAR)
      CALL TRANS1(I,NVAR)
      CALL MATINV(XOUT,NVAR,40,Y,0,T1,IT2,IT3,DETERM)
      DO 5 J=1,NVAR
      SUM = 0.
      DO 4 K=1,NVAR
    4 SUM = SUM + XOUT(J,K)*Y(K,I)
    5 COEF(I,J) = SUM
      PRINT 1003, NAMOUT(I)
      PRINT 1006, (NAMVAR(K),K=1,NVAR)
      PRINT 1007, (COEF(I,J),J=1,NVAR)
      IF(JSWTCH .EQ. 1) PUNCH 1005, (COEF(I,J),J=1,NVAR)
   10 CONTINUE
      IF(ISWTCH .EQ. 1) GO TO 31
   12 READ 1004, NR
      IF(KSWTCH .EQ. 0) GO TO 15
      READ 1004, (NV(I),I=1,8)
      DO 14 N=1,8
      NVAR = NV(N)
   14 READ 1005, (COEF(N,J),J=1,NVAR)
   15 DO 20 I=1,NR
      READ 1001, (VARIN(K),K=1,13)
      READ 1005, (OUTPUT(J),J=1,8)
      CALL TRANS2
      TSS = 0.
      DO 18 J=1,8
      NVAR = NV(J)
      SUM = 0.
      DO 17 K=1,NVAR
   17 SUM = SUM + COEF(J,K)*VAROUT(K,J)
      OUT(J) = SUM
   18 TSS = TSS + ((OUT(J)-OUTPUT(J))/OUTPUT(J))**2
      IF(I .EQ. 1) PRINT 1008
      PRINT 1009, I, (OUT(K),K=1,8), (VARIN(J),J=1,13),TSS
      DO 19 K=1,8
      RUNSUM(K) = RUNSUM(K) + OUT(K)
      XMEAN(K) = RUNSUM(K)/FLOAT(I)
      VSUM(K) = VSUM(K) + ((OUT(K)-XMEAN(K))**2)
```

```
   19 STDDEV(K) = SQRT(VSUM(K)/FLOAT(I-1))
      IF(I-(10*(I/10)))20,191,20
  191 PRINT 1010
      PRINT 1011, XMEAN,STDDEV
   20 CONTINUE
 1001 FORMAT(10X,7E10.3)
 1002 FORMAT(8(A8,2X))
 1003 FORMAT(1H1,*OUTPUT VARIABLE *,A8,*--- COEFFICIENTS*)
 1004 FORMAT(8I5)
 1005 FORMAT(8E10.3)
 1006 FORMAT(1H0,10(5XA8))
 1007 FORMAT(1H0,5X,10(E10.3,3X))
 1008 FORMAT(1H1,*RUN*,2X,*PFT*,8X,*PMT*,8X,*RCON*,7X,*FARMI*,6X,*CF*,
     1        9X,*FOREX*,6X,*YA*,9X,*YM*,9X,*C1*,9X,*PAO*,8X,*C3*/6X,
     2        *C4*,9X,*C5*,9X,*C6*,9X,*C7*,9X,*C8*,9X,*C12*,8X,*C41*,8X,
     3        *C42*,8X,*AL1*,8X,*AL2*,8X,*TSS*)
 1009 FORMAT(1H0,I3,11(1XE10.3)/4X,11(1XE10.3))
 1010 FORMAT(1H0,5X,*PFTM*,9X,*PMTM*,9X,*RCONM*,8X,*FARMIM*,7X,*CFM*,
     1        10X,*FOREXM*,7X,*YAM*,10X,*YMM*/6X,*PFTS*,9X,*PMTS*,9X,
     2        *RCONS*,8X,*FARMIS*,7X,*CFS*,10X,*FOREXS*,7X,*YAS*,10X,
     3        *YMS*)
 1011 FORMAT(1H0,5X,8(E10.3,3X)/6X,8(E10.3,3X))
   31 END
```

```
      SUBROUTINE MATINV(A,N,NMAX,B,M,PIVOT,IPIVOT,INDEX,DETERM)
C     F1 CODA MATINV  MATRIX INVERSION
C     MATRIX INVERSION WITH ACCOMPANYING SOLUTION OF LINEAR EQUATIONS
C
C     MATRIX INVERSION WITH ACCOMPANYING SOLUTION OF LINEAR EQUATIONS
C
      DIMENSION IPIVOT(N),A(NMAX,NMAX),B(NMAX,1),INDEX(N,2),PIVOT(N)
C
C     INITIALIZATION
C
   10 DETERM=1.0
   15 DO 20 J=1,N
   20 IPIVOT(J)=0
   30 DO 550 I=1,N
C
C     SEARCH FOR PIVOT ELEMENT
C
   40 AMAX=0.0
   45 DO 105 J=1,N
   50 IF (IPIVOT(J)-1) 60, 105, 60
   60 DO 100 K=1,N
   70 IF (IPIVOT(K)-1) 80, 100, 740
   80 IF (ABSF(AMAX)-ABSF(A(J,K))) 85, 100, 100
   85 IROW=J
   90 ICOLUM=K
   95 AMAX=A(J,K)
  100 CONTINUE
  105 CONTINUE
  110 IPIVOT(ICOLUM)=IPIVOT(ICOLUM)+1
C
C     INTERCHANGE ROWS TO PUT PIVOT ELEMENT ON DIAGONAL
C
  130 IF (IROW-ICOLUM) 140, 260, 140
  140 DETERM=-DETERM
  150 DO 200 L=1,N
  160 SWAP=A(IROW,L)
  170 A(IROW,L)=A(ICOLUM,L)
  200 A(ICOLUM,L)=SWAP
  205 IF(M) 260, 260, 210
  210 DO 250 L=1, M
  220 SWAP=B(IROW,L)
  230 B(IROW,L)=B(ICOLUM,L)
  250 B(ICOLUM,L)=SWAP
  260 INDEX(I,1)=IROW
  270 INDEX(I,2)=ICOLUM
  310 PIVOT(I)=A(ICOLUM,ICOLUM)
  320 DETERM=DETERM*PIVOT(I)
C
C     DIVIDE PIVOT ROW BY PIVOT ELEMENT
C
  330 A(ICOLUM,ICOLUM)=1.0
  340 DO 350 L=1,N
  350 A(ICOLUM,L)=A(ICOLUM,L)/PIVOT(I)
  355 IF(M) 380, 380, 360
  360 DO 370 L=1,M
  370 B(ICOLUM,L)=B(ICOLUM,L)/PIVOT(I)
```

```
C
C     REDUCE NON-PIVOT ROWS
C
  380 DO 550 L1=1,N
  390 IF(L1-ICOLUM) 400, 550, 400
  400 T=A(L1,ICOLUM)
  420 A(L1,ICOLUM)=0.0
  430 DO 450 L=1,N
  450 A(L1,L)=A(L1,L)-A(ICOLUM,L)*T
  455 IF(M) 550, 550, 460
  460 DO 500 L=1,M
  500 B(L1,L)=B(L1,L)-B(ICOLUM,L)*T
  550 CONTINUE
C
C     INTERCHANGE COLUMNS
C
  600 DO 710 I=1,N
  610 L=N+1-I
  620 IF (INDEX(L,1)-INDEX(L,2)) 630, 710, 630
  630 JROW=INDEX(L,1)
  640 JCOLUM=INDEX(L,2)
  650 DO 705 K=1,N
  660 SWAP=A(K,JROW)
  670 A(K,JROW)=A(K,JCOLUM)
  700 A(K,JCOLUM)=SWAP
  705 CONTINUE
  710 CONTINUE
  740 RETURN
  750 END
```

```
      SUBROUTINE TRANS1(I,NV)
      COMMON /BLK1/ XIN(40,13), XOUT(40,40)
      DO 10 J=1,NV
      XOUT(J,1) = 1.0
      GO TO (1,2,3,4,5,6,7,8)I
    1 XOUT(J,2) = XIN(J,7) * XIN(J,8)
      XOUT(J,3) = XIN(J,1) * XIN(J,5)
      XOUT(J,4) = XIN(J,3) * XIN(J,7)
      XOUT(J,5) = XIN(J,5) * XIN(J,13)
      XOUT(J,6) = XIN(J,4) * XIN(J,7)
      XOUT(J,7) = XIN(J,5) * XIN(J,10)
      XOUT(J,8) = XIN(J,11) * XIN(J,13)
      XOUT(J,9) = XIN(J,4) * XIN(J,8)
      XOUT(J,10) = XIN(J,1) * XIN(J,11)
      XOUT(J,11) = XIN(J,6) * XIN(J,12)
      XOUT(J,12) = XIN(J,2) * XIN(J,8)
      XOUT(J,13) = XIN(J,7) * XIN(J,10)
      XOUT(J,14) = XIN(J,7) * XIN(J,13)
      GO TO 10
    2 XOUT(J,2) = XIN(J,7) * XIN(J,8)
      XOUT(J,3) = XIN(J,5) * XIN(J,10)
      XOUT(J,4) = XIN(J,3) * XIN(J,7)
      XOUT(J,5) = XIN(J,5) * XIN(J,11)
      XOUT(J,6) = XIN(J,1) * XIN(J,8)
      XOUT(J,7) = XIN(J,4) * XIN(J,8)
      XOUT(J,8) = XIN(J,6) * XIN(J,12)
      XOUT(J,9) = XIN(J,13)**3
      XOUT(J,10) = XIN(J,1) * XIN(J,3)
      XOUT(J,11) = XIN(J,7) * XIN(J,10)
      XOUT(J,12) = XIN(J,5) * XIN(J,8)
      GO TO 10
    3 XOUT(J,2) = XIN(J,4) * XIN(J,8)
      XOUT(J,3) = XIN(J,3) * XIN(J,5)
      XOUT(J,4) = XIN(J,7) * XIN(J,13)
      XOUT(J,5) = XIN(J,3) * XIN(J,10)
      XOUT(J,6) = XIN(J,1) * XIN(J,12)
      GO TO 10
    4 XOUT(J,2) = XIN(J,7) * XIN(J,8)
      XOUT(J,3) = XIN(J,2) * XIN(J,10)
      XOUT(J,4) = XIN(J,3) * XIN(J,5)
      XOUT(J,5) = XIN(J,1) * XIN(J,7)
      XOUT(J,6) = XIN(J,3) * XIN(J,11)
      XOUT(J,7) = XIN(J,2) * XIN(J,4)
      XOUT(J,8) = XIN(J,4) * XIN(J,13)
      XOUT(J,9) = XIN(J,10)**3
      XOUT(J,10) = XIN(J,3) * XIN(J,9)
      GO TO 10
    5 XOUT(J,2) = XIN(J,8) * XIN(J,10)
      XOUT(J,3) = XIN(J,3) * XIN(J,7)
      XOUT(J,4) = XIN(J,2)**3
      XOUT(J,5) = XIN(J,5) * XIN(J,10)
      XOUT(J,6) = XIN(J,7) * XIN(J,8)
      XOUT(J,7) = XIN(J,3) * XIN(J,10)
      XOUT(J,8) = XIN(J,11)
      XOUT(J,9) = XIN(J,5) * XIN(J,9)
      XOUT(J,10) = XIN(J,2) * XIN(J,7)
```

```
      XOUT(J,11)= XIN(J,9) * XIN(J,11)
      XOUT(J,12)= XIN(J,2) * XIN(J,9)
      GO TO 10
    6 XOUT(J,2) = XIN(J,5) * XIN(J,10)
      XOUT(J,3) = XIN(J,5) * XIN(J,12)
      XOUT(J,4) = XIN(J,7) * XIN(J,8)
      XOUT(J,5) = XIN(J,2) * XIN(J,11)
      XOUT(J,6) = XIN(J,3) * XIN(J,11)
      XOUT(J,7) = XIN(J,7) * XIN(J,9)
      GO TO 10
    7 XOUT(J,2) = XIN(J,2) * XIN(J,10)
      XOUT(J,3) = XIN(J,7) * XIN(J,8)
      XOUT(J,4) = XIN(J,3) * XIN(J,5)
      XOUT(J,5) = XIN(J,2) * XIN(J,11)
      XOUT(J,6) = XIN(J,7) * XIN(J,9)
      XOUT(J,7) = XIN(J,8) * XIN(J,10)
      XOUT(J,8) = XIN(J,8) * XIN(J,11)
      XOUT(J,9) = XIN(J,1) * XIN(J,9)
      GO TO 10
    8 XOUT(J,2) = XIN(J,7) * XIN(J,8)
      XOUT(J,3) = XIN(J,1) * XIN(J,5)
      XOUT(J,4) = XIN(J,3) * XIN(J,7)
      XOUT(J,5) = XIN(J,5) * XIN(J,13)
      XOUT(J,6) = XIN(J,4) * XIN(J,7)
      XOUT(J,7) = XIN(J,5) * XIN(J,10)
      XOUT(J,8) = XIN(J,11) * XIN(J,13)
      XOUT(J,9) = XIN(J,4) * XIN(J,8)
      XOUT(J,10)= XIN(J,1) * XIN(J,11)
      XOUT(J,11)= XIN(J,6) * XIN(J,8)
      XOUT(J,12)= XIN(J,2) * XIN(J,12)
   10 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE TRANS2
      COMMON /BLK2/ VARIN(13), VAROUT(40,8)
      DO 1 I=1,8
    1 VAROUT(1,I) = -1.0
      VAROUT(2,1) = VARIN(7) * VARIN(8)
      VAROUT(3,1) = VARIN(1) * VARIN(5)
      VAROUT(4,1) = VARIN(3) * VARIN(7)
      VAROUT(5,1) = VARIN(5) * VARIN(13)
      VAROUT(6,1) = VARIN(4) * VARIN(7)
      VAROUT(7,1) = VARIN(5) * VARIN(10)
      VAROUT(8,1) = VARIN(11)* VARIN(13)
      VAROUT(9,1) = VARIN(4) * VARIN(8)
      VAROUT(10,1)= VARIN(1) * VARIN(11)
      VAROUT(11,1)= VARIN(6) * VARIN(12)
      VAROUT(12,1)= VARIN(2) * VARIN(8)
      VAROUT(13,1)= VARIN(7) * VARIN(10)
      VAROUT(14,1)= VARIN(7) * VARIN(13)
      VAROUT(2,2) = VARIN(7) * VARIN(8)
      VAROUT(3,2) = VARIN(5) * VARIN(10)
      VAROUT(4,2) = VARIN(3) * VARIN(7)
      VAROUT(5,2) = VARIN(5) * VARIN(11)
      VAROUT(6,2) = VARIN(1) * VARIN(8)
      VAROUT(7,2) = VARIN(4) * VARIN(8)
      VAROUT(8,2) = VARIN(6) * VARIN(12)
      VAROUT(9,2) = VARIN(13)**3
      VAROUT(10,2)= VARIN(1) * VARIN(3)
      VAROUT(11,2)= VARIN(7) * VARIN(10)
      VAROUT(12,2)= VARIN(5) * VARIN(8)
      VAROUT(2,3) = VARIN(4) * VARIN(8)
      VAROUT(3,3) = VARIN(3) * VARIN(5)
      VAROUT(4,3) = VARIN(7) * VARIN(13)
      VAROUT(5,3) = VARIN(3) * VARIN(10)
      VAROUT(6,3) = VARIN(1) * VARIN(12)
      VAROUT(2,4) = VARIN(7) * VARIN(8)
      VAROUT(3,4) = VARIN(2) * VARIN(10)
      VAROUT(4,4) = VARIN(3) * VARIN(5)
      VAROUT(5,4) = VARIN(1) * VARIN(7)
      VAROUT(6,4) = VARIN(3) * VARIN(11)
      VAROUT(7,4) = VARIN(2) * VARIN(4)
      VAROUT(8,4) = VARIN(4) * VARIN(13)
      VAROUT(9,4) = VARIN(10)**3
      VAROUT(10,4)= VARIN(3) * VARIN(9)
      VAROUT(2,5) = VARIN(8) * VARIN(10)
      VAROUT(3,5) = VARIN(3) * VARIN(7)
      VAROUT(4,5) = VARIN(2)**3
      VAROUT(5,5) = VARIN(5) * VARIN(10)
      VAROUT(6,5) = VARIN(7) * VARIN(8)
      VAROUT(7,5) = VARIN(3) * VARIN(10)
      VAROUT(8,5) = VARIN(11)
      VAROUT(9,5) = VARIN(5) * VARIN(9)
      VAROUT(10,5)= VARIN(2) * VARIN(7)
      VAROUT(11,5)= VARIN(9) * VARIN(11)
      VAROUT(12,5)= VARIN(2) * VARIN(9)
      VAROUT(2,6) = VARIN(5) * VARIN(10)
      VAROUT(3,6) = VARIN(5) * VARIN(12)
      VAROUT(4,6) = VARIN(7) * VARIN(8)
```

```
      VAROUT(5,6) = VARIN(2) * VARIN(11)
      VAROUT(6,6) = VARIN(3) * VARIN(11)
      VAROUT(7,6) = VARIN(7) * VARIN(9)
      VAROUT(2,7) = VARIN(2) * VARIN(10)
      VAROUT(3,7) = VARIN(7) * VARIN(8)
      VAROUT(4,7) = VARIN(3) * VARIN(5)
      VAROUT(5,7) = VARIN(2) * VARIN(11)
      VAROUT(6,7) = VARIN(7) * VARIN(9)
      VAROUT(7,7) = VARIN(8) * VARIN(10)
      VAROUT(8,7) = VARIN(8) * VARIN(11)
      VAROUT(9,7) = VARIN(1) * VARIN(9)
      VAROUT(2,8) = VARIN(7) * VARIN(8)
      VAROUT(3,8) = VARIN(1) * VARIN(5)
      VAROUT(4,8) = VARIN(3) * VARIN(7)
      VAROUT(5,8) = VARIN(5) * VARIN(13)
      VAROUT(6,8) = VARIN(4) * VARIN(7)
      VAROUT(7,8) = VARIN(5) * VARIN(10)
      VAROUT(8,8) = VARIN(11) * VARIN(13)
      VAROUT(9,8) = VARIN(4) * VARIN(8)
      VAROUT(10,8) = VARIN(1) * VARIN(11)
      VAROUT(11,8) = VARIN(6) * VARIN(8)
      VAROUT(12,8) = VARIN(2) * VARIN(12)
      RETURN
      END
RUN,2.0,2100

EXECUTION STARTED AT   1532 *43
```

REFERENCES

1. Logsdon, Thomas S. and Robert C. Africano, "A Modified Monte Carlo Procedure," AIAA Journal, Vol. 6, No. 6, June 1968, pp. 1111-1117.

2. Ruble, William L. and Mary E. Rafter, "LSADD, Stepwise Addition of Variables to Form a Least Squares Equation," Michigan State University Agricultural Experiment Station, STAT Series Description No. 9, March, 1966.

3. Elkin, Sanford, "Matrix Inversion with Solution of Linear Equations," Michigan State University Computer Laboratory Program Documentation No. 00000040, December, 1963.

4. Box, G. E. P. and Norman R. Draper, "A Basis for the Selection of a Response Surface Design," American Statistical Association Journal, September, 1959, pp. 622-65.