Martin Schatzoff

Design of Experiments in Computer Performance Evaluation

Techniques of statistical design of experiments have been successfully employed for many decades in a variety of applications in industry, agriculture, medicine, psychology, and other physical and social sciences. Their aim is to provide scientific and efficient means of studying the effects, on one or more variables of interest, of varying multiple controllable factors in an experiment. These techniques have not been widely used in the study of computer systems, although they can potentially have as large an impact as they have had in other fields. The purpose of this paper is to review some of the basic concepts underlying the statistical design and analysis of experiments and to illustrate them by means of examples drawn from studies of computer system performance. The examples include comparisons of alternate page replacement and free storage management algorithms, optimization of a scheduler, and validation of a system simulation model.

1. Introduction

Performance measurement frequently has as its purpose to evaluate the effects of changes to a system. Such changes might include software options (*e.g.*, choice of access methods, scheduling parameters, system generation options) as well as hardware changes (*e.g.*, amount of main storage, number of channels, etc.). In some instances such evaluations are carried out on fixed benchmark workloads, while in others they may be carried out with real users. Purposes of such experimentation may be to compare the performance implications of the changes to the system or to optimize or tune the system. In either event, we would like to carry out the experimental program as quickly and economically as possible, while at the same time being able to assure ourselves of the accuracy and validity of the results.

When dealing with complex computer systems, many factors, controllable by the experimenter, can have major effects on system performance. The theory of statistical design of experiments provides methodology for designing and analyzing experiments involving simultaneous variation of multiple factors. Historically, there has existed a major communication gap between computer scientists, system programmers, and designers, on the one hand, and statisticians, on the other. As a result, statistical design and analysis of experiments is rarely applied in computer studies. The consequences may include unnecessary expense, undue time delays, loss of information, misinformation, and incorrect conclusions.

The purpose of this paper is to introduce some of the basic ideas underlying the statistical design of experiments and to illustrate these by means of real examples drawn from computer performance evaluation work undertaken at this location over a number of years. In all experimental design work, we envision a statistical model which relates some measurable response (or responses) to the factors varied in the experiment. In this paper, we confine our attention to the so-called fixed effects analysis of variance model, which expresses the response as a linear function of the factors, plus an additive random error term. This is essentially a linear regression model, in which the independent variables are dummy variables indicating the presence or absence of a specified value, or level, of a corresponding factor. The parameters, or coefficients, of the linear model then measure the effects of variations in the factors, both singly (main effects) and in combination (interaction effects). The general form of this model is given in Eq. (1).

Given this form of the model, the design of experiments is concerned with procedures which will enable us to

estimate these parameters efficiently and draw statistically valid conclusions, while minimizing or eliminating the effects of extraneous factors that may be beyond our control. Additionally, we must be able to test the validity of the model itself, based upon the resulting data. Thus, the entire process is illustrative of the scientific method itself, wherein we may go through iterative cycles of stipulating hypotheses, carrying out experiments, testing the hypotheses and the model upon which they are based, and possibly reformulating the model and trying again.

The subject begins to take on real significance when we are dealing with multiple factors. Then, it is always better to vary the factors simultaneously, rather than one at a time (as frequently done by engineers unfamiliar with experimental design); design techniques are available for drastically reducing the total number of combinations that must be run, resulting in savings in cost and/or time. In the ensuing sections, we develop the ideas of randomization, factorial experiments, fractional replication, blocking, and response surface exploration. These are illustrated by real examples ranging from benchmark experiments to experiments carried out under actual operating conditions and live workloads. The objective is not to train the non-statistician to become a statistician in one easy lesson, but rather to help him gain an appreciation for the subject, the potential power it offers, and its applicability to real performance evaluation problems.

2. Some basic concepts

The performance of a computing system is a function of the hardware configuration, the operating system and associated application software support required to run it, and the workload imposed on it by its users. A key objective of performance evaluation is to understand the relationships among these constituent system elements from the standpoint of how they affect system performance. The ideal way of meeting this objective would be to have a detailed mathematical model which explicitly displays the nature of these relationships. Such a model could then be used to study the effects of hardware, software, and workload variations upon system performance, and thus provide a means of predicting and optimizing such performance with respect to these factors.

When mathematical system models of sufficient detail and accuracy cannot be readily derived, the only alternatives are to conduct experiments on the system itself, or on a simulation model of it. The objectives remain the same, but the methodological approach now involves derivation of a suitably parameterized empirical model, where the model is an equation whose parameters are to be estimated from the data. The statistical design of experiments provides techniques for designing and analyzing experiments in such a way as to derive such models efficiently and make statistically valid inferences about the underlying mechanisms.

In a designed experiment, variables that are to be controlled are referred to as *factors*. The different values or states of a factor are referred to as *levels* of that factor. For example, in an experiment aimed at comparing the performance of two different page replacement algorithms, say A and B, the variable called *algorithm* is a factor, and its two possible states, A and B, are its levels. Some factors may be quantitative in nature. Thus, suppose that a scheduling algorithm makes decisions based upon whether or not the page steal rate observed in the system over the past minute exceeds a specified threshold. The possible numerical values assigned to that threshold in the experiment are levels of the page steal threshold factor.

A multifactor experiment is one in which two or more factors are varied simultaneously. A particular combination of factor/levels for an experimental run is called a *treatment*, deriving from agricultural experiments in which different combinations of chemicals were administered to different plants. An *experimental design* defines the treatments, or factor/level combinations, at which experimental runs are to be carried out. The theory of design of experiments is concerned with principles for constructing experimental designs in such a way as to be able to estimate the *effects* of the different treatments on the response variable(s) of interest as efficiently as possible (*i.e.*, minimizing the required number of observations, the cost, and/or the time required to conduct the experiment). The theory is predicated on an underlying mathematical model that expresses the response variable as a function of the treatments, with unknown parameters, which represent the treatment effects, to be estimated from the data. The model that we discuss is called the fixed effects analysis of variance model, which is essentially a special case of the well known least squares linear regression model

$$Y_i = \theta_0 + X_{1i}\theta_1 + X_{2i}\theta_2 + \cdots + X_{pi}\theta_p + \varepsilon_i \qquad (i = 1, 2, \cdots, n) \qquad (1)$$

where Y_i are the values of the n random response variables, X_{ji} ($j = 1, 2, \cdots, p$) are the known values of the p independent variables for each of the n design points, θ_j are the $p + 1$ unknown regression coefficients, and ε_i are random error terms having the properties

$$E(\varepsilon_i) = 0 ,$$

$$E(\varepsilon_i \varepsilon_k) = \delta_{i,k}\sigma^2 ,$$

where σ^2, the variance of the error term, is an unknown constant, and the operator $E(z)$ represents the expected value of z.

The further assumption that the ε_i are normally distributed leads to normal distribution theory for inferences concerning the unknown θ_j and σ^2. The normality assumption can be tested by examining the residuals from the fitted model.

In the special case of designed experiments, the independent variables in Eq. (1) are defined to be indicator variables having the values one or zero, depending on whether the effects θ_j are present or absent in the experimental conditions corresponding to the respective observations. An effect corresponding to variation of a single factor is called a *main effect*. An effect corresponding to simultaneous variation of two or more factors is called an *interaction effect*. It is represented in the model by introducing a new independent variable which is the product of those independent variables corresponding to the given interaction.

A *complete factorial experiment* is a multifactor experiment in which every possible factor/level combination is included. If there are two or more observations at each experimental design point, the experiment is said to be *replicated*. If the experimental design has the same number of observations at each design point, the design has the property of *orthogonality*. That is, the vectors $\mathbf{X}_j = \{X_{j1}, X_{j2}, \cdots, X_{jn}\}$ and $\mathbf{X}_k = \{X_{k1}, X_{k2}, \cdots, X_{kn}\}$, corresponding to the levels of any two factors, are orthogonal to one another. Orthogonality in a designed experiment is very important, because it permits independent estimation of all of the effects.

Replication in an experiment provides a direct means of estimating the unknown variance σ^2 of the random error term, since according to the model (1), the only difference between two observations taken at the same data point is that their random error terms are different in value. Thus, for two replicated observations Y_i and Y_k, $(Y_i - Y_k) = (\varepsilon_i - \varepsilon_k)$ is a random variable which has expectation equal to zero and variance equal to $2\sigma^2$. Hence, $(Y_i - Y_k)^2/2$ provides an *unbiased* estimate of σ^2. Estimates of this form, computed at all replicated points in an experiment, may be averaged together to provide a better estimate of σ^2.

The importance of estimating σ^2 accurately is that it provides a yardstick against which the statistical significance of experimental effects may be assessed. Thus, if a calculated estimate of variation due to a particular factorial effect is sufficiently larger than the estimated error variance $\hat{\sigma}^2$, one would infer that the effect is real—that is, it is significantly larger than would be expected if the experimental treatment had no real effect, and we were measuring only random variation.

Since economy is an important aspect of an experimental program, replication should be avoided where possible, since additional observations represent a duplication of effort. In some instances, the experimenter may have estimates of the experimental error obtained from previous experiments. For example, suppose that a fixed benchmark has been used in previous experiments from which σ^2 has been estimated. New experiments involving the same benchmark may be assumed to have the same error variance, provided that other experimental conditions have not changed materially. Replication at a small number of design points can be employed for confirmation.

When previous estimates of σ^2 are not available, it can frequently be assumed that higher order interaction terms are insignificant, and hence that they essentially measure random variation. In such cases, these higher order interaction terms are used to provide estimates of the error variance.

In order to make these ideas more precise, we reparameterize the regression model (1) to reflect more succinctly the situation encountered in factorial experiments. For illustrative purposes, we consider the simple case of a two factor experiment, in which factor A is varied at I levels and factor B at J levels. We assume also that there are K replications at each point. Since the X_{ji} in (1) are equal to zero or one, we can eliminate those terms for which $X_{ji} = 0$ and suppress the symbol X_{ji} when it is equal to one. Also the parameters $\theta_0, \cdots, \theta_p$ in the regression equation (1) are replaced by the notation in Eq. (2), which is commonly used in experimental design models. Equation (1) thus becomes

$$Y_{ijk} = \mu + \alpha_i + \beta_j + \gamma_{ij} + \varepsilon_{ijk} \quad (i = 1, 2, \cdots, I,\ j = 1, 2, \cdots, J,\ k = 1, 2, \cdots, K), \tag{2}$$

where

μ = general mean,
α_i = main effect of factor A at level i,
β_j = main effect of factor B at level j,
γ_{ij} = interaction effect of factor A at level i and factor B at level j.

The model generalizes in an obvious manner as the number of factors increases, so that for experiments with

p factors, there are interaction terms of order 1, 2, 3, $\cdots$, $p - 1$, where a kth order interaction refers to the interaction of $k + 1$ terms. The terms of Eq. (2) are estimated by least squares. That is, we minimize the sum of squared deviations of the observations about their fitted values by differentiating the expression

$$\Sigma_{i=1}\Sigma_{j=1}\Sigma_{k=1}(y_{ijk} - \mu - \alpha_i - \beta_j - \gamma_{ij})^2 \qquad (3)$$

with respect to the unknown parameters. Since the various main effects and interactions are deviations about a mean, side conditions such as

$$\Sigma_i\alpha_i = 0,\ \Sigma_j\beta_j = 0,\ \Sigma_i\Sigma_j\gamma_{ij} = 0$$

must be introduced. The resulting formulas, for cases of orthogonal designs, are provided in most textbooks dealing with the subject of designed experiments. For example, the estimate $\hat{\mu}$ of the general mean μ is given by the sample mean of all the observations:

$$\hat{\mu} = Y_{\ldots} = (\Sigma_{i=1}\Sigma_{j=1}\Sigma_{k=1}Y_{ijk})/IJK\ , \qquad (4)$$

while the estimate $\hat{\alpha}$ of α is given by

$$\hat{\alpha}_i = Y_{i\ldots} - Y_{\ldots} \qquad (i = 1, 2, \cdots, I)\ , \qquad (5)$$

where $Y_{i\ldots} = (\Sigma_{j=1}\Sigma_{k=1}Y_{ijk})/JK$.

In simple words, the ith main effect of factor A is estimated as the difference between the mean of all observations taken at level i of factor A and the general mean. The dot (.) notation employed in the subscripts of Y in the above equations denotes averaging over the subscript indices that have been replaced by the dots. Similarly, the jth main effect of factor B, β_j, is estimated as the difference between the mean of all observations taken at level j of factor B and the general mean. The AB interaction effect is a bit more complicated. In essence, it measures the extent to which deviations of the Y_{ijk} about $Y_{i\ldots}$ are dependent upon the level of factor B (and vice versa). Formally, the AB interaction effect when factor A is at level i and factor B is at level j is given by $\hat{\gamma}_{ij} = Y_{ij\cdot} - Y_{i\ldots} - Y_{\cdot j\cdot} + Y_{\ldots}$.

The technique used to analyze data from a designed experiment is called the Analysis of Variance, frequently referred to by its acronym, ANOVA. This is an algebraic decomposition of the sum of squares of the observations about their general mean into additive sums of squares due to the various effects. Thus, in our two factor example we have

$$\Sigma_{i=1}\Sigma_{j=1}\Sigma_{k=1}(Y_{ijk} - Y_{\ldots})^2 = JK\Sigma_{i=1}\hat{\alpha}_i^2 + IK\Sigma_{j=1}\hat{\beta}_j^2 + K\Sigma_{i=1}\Sigma_{j=1}\hat{\gamma}_{ij}^2 + \Sigma_{i=1}\Sigma_{j=1}\Sigma_{k=1}(Y_{ijk} - Y_{ij\cdot})^2 \qquad (6).$$

Since the terms in each sum of squares in (6) represent deviations about estimated means, the "degrees of freedom" (or ranks of the corresponding sub-spaces) are given, respectively, by

Source	*Degrees of freedom*
Total	$IJK - 1$
A	$I - 1$
B	$J - 1$
AB	$(I - 1)(J - 1)$
ERROR	$IJ(K - 1)$

(7)

Thus, the degrees of freedom for the total experiment are given by the total number of observations, IJK, minus 1. Similarly, for a main effect such as A, the degrees of freedom are given by the number of levels of factor A, in this case I, minus 1. The "total" number of degrees of freedom is equal to the sum of the degrees of freedom for the individual sources of variation. Each sum of squares in (6) when divided by its corresponding degrees of freedom in (7) is called a mean square and is an estimate of the variation due to its corresponding source.

An important principle inherent in factorial experiments is that it is better to vary factors simultaneously than one at a time. The advantages are twofold. First, all of the observations from a factorial experiment can be used to estimate each of the effects as well as the error variance. Second, simultaneous variation also permits estimation of the interactions. To illustrate these points, we consider as an example the simplest type of factorial experiment, one in which each factor is varied at two levels. Experiments of this type are called 2^n factorials, reflecting the fact that there are 2^n experimental runs when each of n factors is varied at two levels.

Figure 1 provides a geometric representation of a 2^3 experiment. Each pair of parallel faces on the cube represents four observations at the indicated levels for a particular factor. For example, the vertices on the top face of the cube represent the "high" level of factor B, while those on the bottom face represent the "low" level. Furthermore, each of these faces contains two vertices at which each of the other factors is at the "high" level and two at which they are at the "low" level. Thus, in comparing the observations on any two parallel faces, the effects of the factors represented by the remaining faces are balanced out. This is an illustration of the orthogonality principle, which ensures that the various main effects and interactions can be estimated independently of one another.

For example, half the difference in the average value of the four observations on the top face and those on the

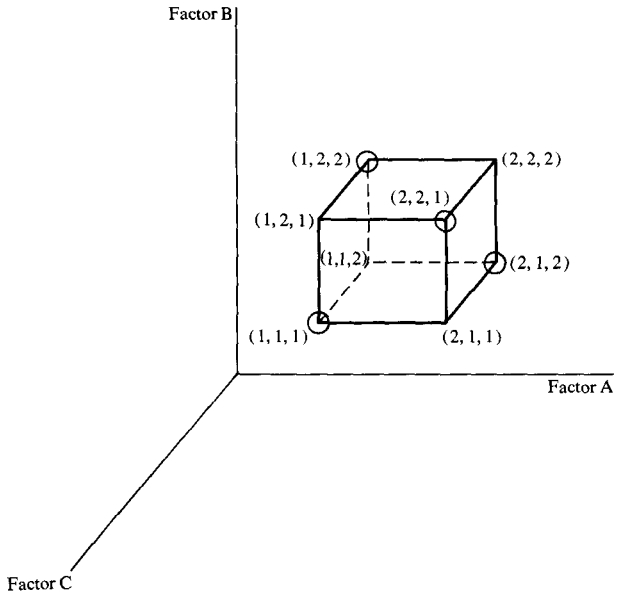

Figure 1 One-half replicate of a 2^3 fractional factorial design.

bottom face measures the main effect of factor B, independent of factors A and C. Similar comparisons on the other pairs of parallel faces provide estimates of the main effects of their corresponding factors. The interaction between factors A and B is measured by half the difference of the effect of factor A when factor B is at the high level and its effect when factor B is at the low level.

As the number of factors to be included in an experiment increases, the total number of observations required for a full factorial grows geometrically. However, if we are willing to ignore the effects of high order interactions, which frequently prove to be negligible, it is possible to carry out a fraction of a full experiment, while still maintaining orthogonality and the ability to estimate main effects and low order interactions. In Fig. 1, the circled vertices represent a balanced one-half fraction of the full 2^3, while the uncircled vertices represent the complementary one-half fraction. Either fraction could be used to estimate the main effects of the three factors, assuming their interactions to be nonexistent. Instead of having four observations on each of any pair of parallel faces of the cube, we now have only two. The design is still balanced, however, since at each level of a particular factor, there is one observation at each level of the other two factors.

As the number of factors increases, the degree of possible *fractional replication* also increases, permitting much larger economies. For example, in a 2^7 experiment, a one-fourth replicate (32 out of 128 possible combinations) will yield estimates of all main effects and first order interactions. While the concept of fractional factorial experiments is most readily understood in the case of the 2^n series, such designs also exist for experiments in which some factors are varied at more than two levels. Actual collections of designs, as well as algorithms for constructing them, may be found in a number of books such as Cochran and Cox [1] and Davies [2]. An excellent discussion of fractional factorials in the 2^n series is provided by Box and Hunter [3]. The original idea was introduced by Finney [4].

Other classes of designs aimed at producing limited information in a minimum number of runs include the Plackett-Burman [5] (1946), Addelman-Kempthorne [6], and Addelman [7] plans. The 2^{n-p} fractional factorials are a subset of the Plackett-Burman designs. Still other classes of designs achieve economies by sacrificing orthogonality, as described by Margolin [8]. It is beyond the scope of this paper to describe the principles and properties underlying these various schemes; however, the interested reader may wish to refer to some of the above referenced works. Also recommended is an expository paper by Kleijnen [9], which discusses and compares the properties of a number of experimental designs for screening large numbers of factors.

Once a specific experimental design has been selected for a given situation, the next step is that of assigning the experimental *treatments* to the experimental *units*. In computer experiments, this process usually refers to the method used for ordering the set of experimental runs. The basic principle underlying this allocation, or sequencing, process is that of randomization. That is, a randomization procedure is employed to decide which treatments are allocated to which experimental units, in order to give each unit the same chance of receiving any treatment. The idea is to protect against inadvertent introduction of systematic, uncontrollable effects that might be associated with an allocation procedure governed by some nonrandom process.

In carrying out experiments on real systems, we distinguish between two cases, namely, that in which we specify the workload and hold it fixed during the period of experimentation, and that in which the experimentation is carried out during periods of actual system operation, with real, uncontrollable workloads. The first approach includes benchmark tests on real systems as well as experiments carried out on system simulators. It has the advantage of providing a completely controlled environment, with reproducible results. However, conclusions drawn from such experiments are dependent on the

particular workload, and it may be very difficult to extrapolate the results to other workloads, including real ones encountered at a particular installation. In the second instance, we are sampling the workload of a given installation during periods of actual system operation, and hence can make valid inferences about the effects of varying hardware and/or software factors in that installation. However, since the workload is uncontrollable, the results can be affected by variations in the workload over the period of experimentation. Fortunately, design and analysis techniques are available for eliminating, or substantially reducing, the effects of such uncontrolled variation. One such technique is that of randomization, as described above.

To illustrate, we consider an example reported by Margolin, Parmelee, and Schatzoff [10] in which two different free storage algorithms for IBM's CP-67 were to be compared under live loads. It was decided that each of two different versions of the system would be run on different days, over a two week period. In order to ensure an equal number of trials under each condition, we could use a table of pseudo random numbers to randomly select five digits and associate these with the corresponding days of the two week period. Thus, if the random selection yielded the digits 8, 3, 4, 5, 1, we would run one of the two versions on Monday, Wednesday, Thursday, and Friday of week one and Wednesday of week two. The other version would be run on the remaining five days. Furthermore, we could decide, by randomization, which version of the system would be run on which set of days.

An obvious shortcoming of this procedure is that there might be significant differences in workload between week 1 and week 2 or among given days of the week. The above selection lacks balance with respect to both of these factors. The actual plan used in this experiment is reproduced in Table 1, which exhibits the following properties:

1. Each version is run the same number of times in each week.
2. Each version is run once on each day of the week.
3. The design provides protection against any possible linear or quadratic time trend in the data.

Another way of summarizing the above properties would be to state that sums of squares for day and week, or for linear and quadratic trends, can be isolated from the treatment (algorithm) sum of squares. Once the design has been constructed, randomization is used only to decide which version corresponds to A and which to B.

Another technique for isolating uncontrollable effects is that of *blocking*, a device for grouping together experimental units that are similar in nature and applying treatments randomly to units within blocks. This terminology, again, stems from agricultural experimentation, in which areas of land were laid out in rectangular blocks, and specified treatment combinations were applied to plants within each block. As in the case of fractional replication, the basic idea is to assume that high order interactions are negligible, and then design the experiment in such a way as to *confound* these selected interaction effects with the block effects. That is, the calculation of sums of squares between blocks will be exactly the same as the calculation of the sums of squares for those interaction effects with which the block effects are confounded. Also, mutual orthogonality will be maintained among the main effects and the block effects.

Table 1 Experimental design for free storage experiment.

	Monday	*Tuesday*	*Wednesday*	*Thursday*
Week 1	A	B	B	A
Week 2	B	A	A	B

A simple example of the use of this technique is provided in a paper by Bard [11]. The objective was to compare the performance of two different page replacement algorithms under live load conditions. The system was instrumented in such a way that the algorithms could be switched automatically at prescribed time intervals. Thus, instead of running different algorithms on different days, as in the free storage experiment, the page replacement algorithms were switched back and forth every five minutes. Each ten minute interval then provided a comparison of the two algorithms, and workload variations between ten minute intervals, or blocks, could be eliminated from the comparison. Measurements were taken at one minute intervals. In order to eliminate the effect of switching between algorithms, the first observation in each five minute interval was eliminated from the analysis. At fixed intervals during the experiment, a small FORTRAN benchmark program was executed and timed in order to measure the responsiveness of the system.

Randomization was not employed in this experiment, since it was not felt that there was any possibility of systematic or biased behavior in workloads over successive ten minute intervals. A summary of the resulting data is presented in Table 2, which clearly shows that algorithm A was superior in performance to algorithm B, with respect to each of the measured variables (average benchmark completion time in seconds, problem state time, and page read rate), for each day of the experiment. Howev-

Table 2 Results of paging experiment.

Day	*Average benchmark completion time*		*Percentage problem state time*		*Page reads per second*	
	Alg. A	*Alg. B*	*Alg. A*	*Alg. B*	*Alg. A*	*Alg. B*
1	7.0	8.4	45.6	42.6	29.0	29.7
2	7.6	8.5	49.7	43.6	30.0	32.6
3	7.7	10.4	39.5	37.5	26.9	32.1
4	12.1	16.6	42.8	39.3	33.1	36.1
Average	8.6	11.0	44.4	40.7	19.7	32.6

Table 3 Analysis of variance for paging experiment.

Source of variation	*Percent problem state time*			
	Sum of squares (SS)	*Degrees of freedom (DF)*	*Mean squares (MS)*	*F ratio*
Blocks, day 1	5.02	26	0.19	17.04
Blocks, day 2	5.28	26	0.20	17.91
Blocks, day 3	6.40	26	0.25	21.71
Blocks, day 4	6.59	26	0.25	22.35
Days	0.81	3	0.27	23.73
Algorithms	0.29	1	0.29	25.97
Replications	7.34	648	0.0113	
Lack of fit	4.14	107	0.0387	3.41
Total	35.87	863		

er, it is interesting to note that the uncontrolled workload variation was sufficiently large to have produced the wrong conclusions had the experiment consisted of running algorithm A on day 4 and algorithm B on day 2. For example, the average benchmark completion time for algorithm A on day 4 was 12.1 seconds, which is substantially higher than the benchmark completion time for algorithm B on day 2, which was 8.5 seconds.

The resulting analysis of variance is presented in Table 3. It shows that about two thirds of the variability in the experiment, as measured by the sum of squares of blocks within days and that between days, was due to workload variations. The estimate of error variance, derived from the replication sum of squares, is an under-estimate of the true sum of squares, since the replications are positively correlated in time. A more appropriate estimate is given by the "lack of fit" mean square, which represents the interactions, both of blocks and days, with algorithms. There is no reason to expect these interactions to be significant, and hence they may be assumed to provide valid estimates of experimental error. Experiments carried out on fixed benchmarks, as well as those performed on simulators, can benefit also from the randomized block approach. Usually, such experiments consist of relatively lengthy runs made under a given set of conditions, without replication or other means of estimating error, and hence assessing significance of results. Since workloads typically exhibit very high variability, the sequential blocking scheme can produce substantial savings in testing time. Alternatively, the process may be viewed as one which can provide information on a number of factors in a single run. As in the paging study just described, a given experiment may be replicated many times by using sufficiently short blocking intervals, thus providing many independent estimates of the treatment effects.

Thus far, we have discussed design techniques for reducing, or isolating, the effects of uncontrollable variations, usually associated with workload factors. Such techniques are employed during the conduct of the experiment and are dictated in advance by the experimental plan. After the experiment has been completed, such effects may be further reduced by analysis techniques. The basic idea is to capture workload related data that might have an effect on the measured performance responses, and to use these data to adjust the results. For example, in our work with VM/370 performance data, we usually plot performance variables such as response times, CPU throughput, overhead, and paging rates as functions of the number of active users on the system. A plot of this type for the paging experiment described previously is shown in Fig. 2. Although there are other workload dependent factors that may influence system performance, we have at least eliminated the effect of a single major one.

In terms of our linear model, we may in fact introduce terms explicitly representing such concomitant factors, or covariates, as in Eq. (8) below:

$$Y_{ijk} = \mu + \mathbf{X}_i\boldsymbol{\beta} + T_j + B_k + \varepsilon_{ijk}\,, \tag{8}$$

where $\mathbf{X}_i$ is a vector of observations on a set of concomitant factors, or covariates, $\boldsymbol{\beta}$ is a corresponding set of regression coefficients, T_j represents the treatment effects, and B_k represents the block effects.

This model is known as an analysis of covariance model. The analysis itself is performed in two steps. First, a stepwise regression analysis is performed on the set of covariates, and then an analysis of variance is carried out on the adjusted treatment and block terms.

This has the effect of eliminating any linear effects of the selected covariates, or uncontrolled factors, from the analysis, in order to better estimate the effects of the controlled experimental factors. In computer evaluation studies on live workloads, reasonable choices for covariates might include workload dependent variables, such as number of users, or linear or quadratic time trends. The latter may be introduced by means of artificially constructed variables, namely, first and second degree orthogonal polynomials (see Anderson [12, Chapter 3]).

3. Further applications of designed experiments

• *Model validation*

A problem encountered at all levels of modeling, both analytic and simulative, is that of validation. In a sense, complex system models can never be completely validated, since it is virtually impossible to check the model against the modeled system under all conceivable conditions of workload variation and system configuration. The most that can be hoped for in most instances is to be able to demonstrate that the model represents the system satisfactorily for the purposes intended by its builders.

Any model, whether analytic or simulative, has a defined set of input and output variables. The problem of validation is that of verifying that the model accurately predicts the set of outputs for a wide range of input conditions. The dilemma posed by this problem is that of specifying an adequate set of input variable values, as well as an acceptable and measurable level of predictive accuracy. A reasonable solution is to carry out identical multifactor experiments on both the system and its model. Analysis of the results of both experiments will then indicate whether or not there is agreement as to which main effects and interactions are statistically significant. Furthermore, the degree of accuracy can be assessed in terms of the error variance of the model, which can be measured by means of replicated measurements on the real system.

An analysis of this type was carried out by Schatzoff and Tillman [13], who carried out identical half-replicates of a 2^5 experiment on CP-67 and on a detailed simulation model driven by reduced full instruction trace streams. Specifically, a forty user benchmark workload, consisting of different combinations of assemblies, compilations, and interactive sessions, was used to drive the real system, and compressed data derived from full instruction traces of these same workloads were used to drive the simulation model. Since the main reason for developing the simulator was to evaluate different resource management algorithms, it was felt that a satisfactory basis for validation would be provided by identical experiments in which parameters of the dispatching and scheduling algorithms were varied as design factors. Each such experiment consisted of a one-half fraction of a 2^5 factorial. The main effects and first order interactions for three response variables, proportion problem state time, proportion supervisor state time, and page reads per second, are shown for both the actual system and the simulator in Table 4. The factors represented on-off switches for five different control mechanisms, as follows:

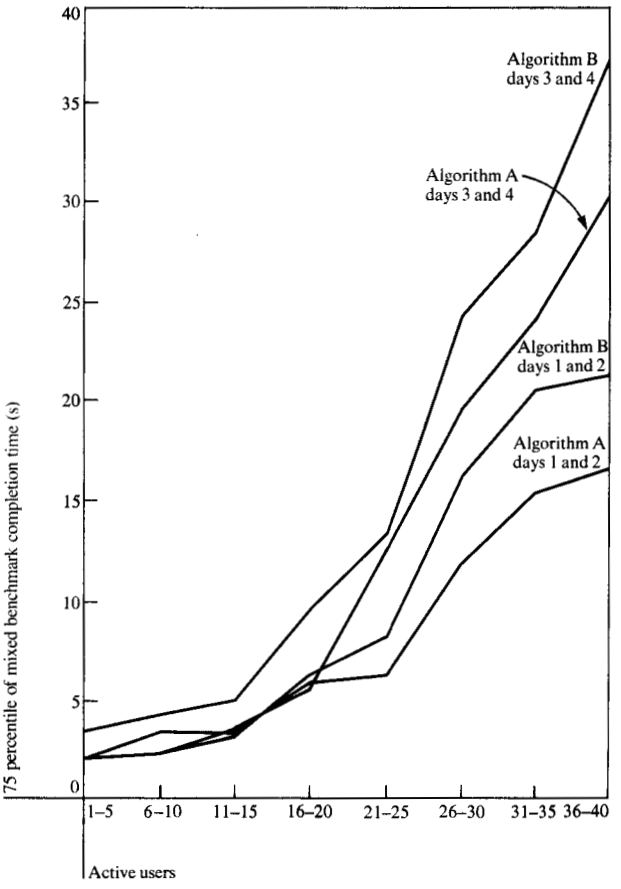

Figure 2 Page replacement algorithms—comparison of results by day.

X_1 Paging penalty,
X_2 CPU usage penalty,
X_3 Maximum page I/O preemption,
X_4 Maximum allowable multiprogramming level,
X_5 Two level Q1 (interactive dispatching queue).

The corresponding estimates of the main effects and first order interaction effects are denoted by $\hat{\theta}_i$ and $\hat{\theta}_{i,j}$ ($i = 1, \cdots, 5; j = 1, \cdots, 5$), as indicated in column 1 of Table 4. The overall mean is indicated by $\hat{\theta}_0$. Underlined values in

Table 4 Factorial effects.

	Proportion problem state		*Proportion supervisor state*		*Page reads per second*	
	Act.	*Sim.*	*Act.*	*Sim.*	*Act.*	*Sim.*
$\hat{\theta}_0$	0.3684	0.4111	0.2814	0.2730	58.9133	65.3156
$\hat{\theta}_1$	0.0026	−0.0049	0.0011	0.0009	0.3438	0.4656
$\hat{\theta}_2$	0.0018	0.0023	−0.0031	0.0034	−1.0032	0.1869
$\hat{\theta}_3$	0.0268	0.0247	−0.0105	−0.0116	−6.3814	−4.9806
$\hat{\theta}_4$	−0.0359	−0.0478	0.0466	0.0414	22.0428	21.5206
$\hat{\theta}_5$	0.0034	0.0012	−0.0100	−0.0105	−3.2128	−4.3431
$\hat{\theta}_{1,2}$	0.0065	0.0031	0.0039	−0.0022	1.6413	−0.7831
$\hat{\theta}_{1,3}$	0.0050	0.0004	0.0010	−0.0030	0.6251	−0.6506
$\hat{\theta}_{1,4}$	0.0001	−0.0073	0.0009	0.0058	0.1856	0.4456
$\hat{\theta}_{1,5}$	0.0029	−0.0043	−0.0022	0.0006	−0.5896	0.2019
$\hat{\theta}_{2,3}$	0.0029	−0.0003	0.0013	0.0023	0.2233	−0.2294
$\hat{\theta}_{2,4}$	−0.0010	0.0042	−0.0024	−0.0020	−0.8409	−0.1531
$\hat{\theta}_{2,5}$	0.0000	0.0047	0.0000	0.0011	−0.0428	1.2031
$\hat{\theta}_{3,4}$	0.0195	0.0281	−0.0100	−0.0082	−5.3464	−4.5631
$\hat{\theta}_{3,5}$	0.0040	−0.0064	−0.0046	0.0016	−2.0133	0.2781
$\hat{\theta}_{4,5}$	−0.0076	0.0033	0.0050	0.0036	2.0857	1.7844

Table 4 represent statistically significant effects. All effects involving X_3 and X_4 were significant well beyond the 1% level, while the X_5 effects were significant near the 2% level. The error variance, estimated by means of replicated measurements on the real system, was used as the basis for assessing statistical significance. A separate analysis, reported in the paper, showed that discrepancies in estimated effects between the system and the simulator were no greater than would be expected in replicated measurements on the real system.

The results shown in Table 4 are striking, because they show complete agreement in picking out the significant effects in both the system and its model. Furthermore, these effects are almost identical in all cases.

- *System tuning*

Given that a complex system has many adjustable parameters whose values can affect various aspects of system performance, an obvious problem of interest, both to system designers and installation managers, is that of finding a combination of parameter values that will optimize some specified measure of system performance. Since the number of parameters may be quite large, the first step is to carry out an initial screening experiment to isolate that subset of parameters which affect the desired performance measure significantly. As previously noted, a very good exposition of such procedures is given by Kleijnen [9]. The second step consists of sequential procedures to find values of the parameter subset of step one that will optimize the performance measure of interest. The methodology for this step is based on the response surface methodology of Box and Wilson [14].

The basic idea underlying this technique is to assume that the performance measure can be approximated by a quadratic function of the parameter values. Then, an experiment is designed for purposes of estimating the coefficients of the quadratic function, or surface, and a hill-climbing procedure is employed to find the maximum (or minimum) point of the function. Box and Wilson have invented a class of experimental designs, called central composite designs, which have certain desirable statistical properties for estimating quadratic surfaces. Their procedure calls for carrying out an initial experiment to estimate the quadratic function, and then employing a "steepest ascent" method to determine a direction of increased yield. One or more confirmatory observations are taken along the path of steepest ascent, followed possibly by a second experiment. The process can continue in an evolutionary manner until the experimenter is satisfied with the results, or no further improvement seems likely.

Procedures of this type have been employed successfully on both CP-67 and VM/370. In the first instance, the CP-67 simulator was used to carry out the experiment, as reported by Schatzoff and Bryant [15], based on the work of our colleague, Y. Bard. This experiment involved five parameters of the dispatching and scheduling algorithms. In each case, the parameter was a numerical threshold for some system activity level, such that different actions would be taken depending on whether or not the threshold was exceeded. The problem was to find values for these thresholds that would maximize problem state throughput, as measured by problem state time as a proportion of total CPU time.

Table 5 Response surface exploration—CP/67.

Run no.	*System scheduler parameter settings*					*% Prob. state*
	X_1	X_2	X_3	X_4	X_5	
*	0.10	0.75	0.67	250	25	40.3
1	0.06	0.50	0.11	175	150.0	45.6
2	0.10	0.50	0.11	175	13.5	46.1
3	0.06	0.70	0.11	175	13.5	42.1
.						
14	0.10	0.50	0.19	325	13.5	34.4
:						
25	0.08	0.60	0.15	250	4.0	44.0
26	0.08	0.60	0.15	250	500.0	42.9
27†	0.08	0.60	0.15	250	45.0	42.6
28	0.10	0.50	0.11	166	11.2	45.8
29	0.10	0.49	0.11	144	8.7	44.6
:						
44	0.10	0.43	0.11	161	25.0	47.4‡
45	0.10	0.43	0.11	163	19.1	43.5
46	0.10	0.43	0.11	160	23.2	45.0
47	0.10	0.43	0.11	160	26.2	45.3
48	0.10	0.43	0.11	150	53.1	45.9
49	0.10	0.43	0.11	160	27.4	44.0
:						
69	0.10	0.43	0.11	162	24.6	44.0

*These refer to standard installation parameter settings.
†Runs 1–27 represent the initial experimental design; the remaining runs represent the automatic hill climbing procedure.
‡Maximum.

After running the initial experiment, which consisted of 27 runs, as shown in Table 5, the quadratic function was automatically fit and the next experimental point was determined. The controlling program would then carry out an experiment at this point, add it to the previous set of results, and delete that point which had given the worst results. This procedure was terminated arbitrarily after 67 runs, but automatic stopping procedures could have been employed. The results showed that it was possible to increase the proportion of problem state from 0.40 to 0.47, even though we had started with a well tuned system. Subsequent confirmatory experiments were carried out to substantiate that this order of performance increase prevailed even after randomly perturbing other aspects of the experimental setup, such as initial placement of pages on the paging drum and log-on order. In this instance, a 2^2 factorial was used to verify the correctness of the initial conclusion. A logical next step would have been to carry out an automatic switching experiment on a live workload, as in the paging experiment described in Section 2, to verify that the new set of parameter values resulted in improved performance.

A problem observed with the above procedure was that the measured response was sometimes quite unstable in small neighborhoods of particular points of seemingly good performance. For example, run number 44, which produced the largest observed value of % problem state time (47.4), represented parameter settings that were very close to those of run numbers 46, 47, 49, and 69, which produced somewhat smaller values, ranging from 44 to 45.3. To overcome this difficulty, a new procedure was devised to smooth the observed values prior to fitting. Essentially, each observed value was replaced by a weighted average of all the observed values, with the weights being decaying exponential functions of the euclidean distances between the given point and each of the remaining points. Thus, each point would in effect be approximated by something like an average of those points in its immediate vicinity. Since this weighting process produces an analytic function of the observations, namely, a sum of weighted exponentials, it is no longer necessary to assume a quadratic response surface. Instead, using methods of nonlinear optimization, the maximal point of this function can be found directly. Each new point would then be added to the preceding set, but no deletions would be made, since the procedure would tend to move away from areas of poor observed response, and points in such areas would automatically carry small weights in areas of high response.

A procedure of this type was carried out by Bard [16] on an experimental version of VM/370 employing a scheduling algorithm of his own design, as described in

Table 6 Response surface exploration—VM/370.

Run no.	*System scheduler parameter settings*				*% Prob. state*	
	X_1	X_2	X_3	X_4	*Actual*	*Smoothed*
1	30	30	30	100	35.3	36.5
2	20	20	35	125	51.8	49.4
3	40	40	25	75	20.2	22.7
4	13	30	30	100	35.6	35.6
5	30	47	30	100	34.5	34.5
6	30	30	30	100	36.6	36.5
7	40	40	35	125	52.4	50.0
8	30	30	39	100	56.7	52.0
9	20	40	35	75	50.6	48.2
10	47	30	30	100	35.8	35.8
11	30	30	30	100	40.6	36.5
12	30	30	21	100	17.3	23.5
13	40	20	35	75	51.3	48.8
14	20	20	25	75	20.6	23.0
15	20	40	25	125	18.3	20.7
16	30	30	30	100	31.5	36.5
17	30	13	30	100	39.4	39.4
18	30	30	30	58	38.2	37.1
19	30	30	30	142	34.5	35.9
20	40	20	25	125	18.8	21.2
21†	30	30	30	100	38.7	36.5
22	30	30	47	94	58.2	57.4
23	30	30	55	57	57.4	57.3
24	30	30	52	192	57.7	57.9
25	30	28	50	118	57.6	57.4
26	30	34	50	114	56.5	57.1
27	30	26	49	83	55.9	56.9
28	30	28	60	290	55.6	55.8
29	30	30	48	217	59.1	58.0‡
30	29	35	42	315	56.8	57.0
31	36	25	43	285	56.8	56.9
32	25	26	45	260	56.7	56.9
33	32	31	46	237	59.3‡	58.0‡
34	38	34	48	248	56.9	57.2
35	32	31	45	242	59.0	58.0‡
36	32	31	45	243	56.6	58.0‡
37	31	30	47	235	56.6	58.0‡
38	31	32	47	223	58.0	58.0‡

†Runs 1–21 represent the initial experimental design; the remaining runs represent the automatic hill climbing procedure.
‡Maximum.
Note: Standard system parameter settings as in Table 5 did not exist for this experimental scheduler.

Bard [17]. Results of this experiment (Table 6) show that a stable area in which problem state throughput was about 58% had been found, after starting at points with very much lower values (20.7–52.0%). In fact, the very first point in the hill-climbing part of the experiment yielded a point that was very close to the experimentally determined optimal (57.4 *vs* 58.0%).

A relevant point of interest, from a systems standpoint, is that the above experiment was carried out automatically through a virtual machine which had privileged access to the control program and hence could change the parameter values. Such a capability would be very desirable for any operating system for purposes of carrying out designed experiments either on benchmarks or live loads. The same facility was used for the live paging experiment previously described. In carrying out automatic tuning procedures on live loads, it is of course very important to prevent the tuning procedure from severely impacting system performance by inadvertently leading it into regions of instability. To avoid such problems, high and low limits for each parameter can be established, with the aid of knowledgeable systems people and data acquired from benchmark experiments. The hill-climbing algorithm can then be programmed in such a way as to prohibit excursions beyond these limits. As a further precaution, since the system is continually monitoring its own performance, it can quickly detect a condition of seriously

impacted performance following a change of parameter values, and revert to the prior setting. The procedure could even be programmed to carry out a small experiment in that region to determine whether the degradation was due to the change of parameter values or to some other coincidental event.

• *Benchmark experiments*

In software development programs, it is customary to benchmark successive program versions under a variety of conditions in order to assess the nature of performance changes. A typical example of such a benchmark is one in which a new version of the IBM Information Management System (IMS) was benchmarked in a 2^4 factorial, where the factors consisted of two different CPU sizes (System/370, Models 158 and 168), two different system configurations (uniprocessor and multiprocessor), two different terminal access methods (BTAM and VTAM), and two different storage access methods (ISAM/OSAM and VSAM). An analysis of variance of the results showed that all interactions were negligible and hence that the same conclusions could have been reached with a half-replicate. Adoption of this procedure in future work could result in savings of 50%.

In other benchmarking programs, the number of factors may be larger, and hence the potential savings realizable through fractional replication will also be larger. Alternatively, additional factors can be introduced without increasing the benchmarking budget, thus providing additional information at no additional cost.

Finally, where practical, automatic switching of factor levels within a run, as in the paging experiment, can further reduce the overall cost of benchmarking. Furthermore, by blocking and measuring in small time intervals, it is possible to determine whether valid conclusions can be drawn with shorter benchmarks.

4. Summary

We have attempted to explain some of the basic principles underlying the statistical design and analysis of experiments and to show by example fruitful areas of application for these ideas in computer performance evaluation work. Opportunities for substantial savings, as well as improved knowledge and insight into complex, poorly understood computing processes, abound at all phases of the product cycle from design to installation management. However, the communication gap between computer systems people and statisticians is still very large, so that such opportunities are seldom realized. In contemplating the many years of successful development and application of principles of experimental design in fields such as agriculture and chemistry, one is quickly led to the conclusion that progress in these fields has come about largely through application of the scientific method. However, computer system development, particularly on the software side, appears not to have followed this path to any noticeable degree. Designed experiments would provide a rewarding first step.

References

1. W. G. Cochran and G. M. Cox, *Experimental Designs* (Second Edition), John Wiley & Sons, Inc., New York, 1957.
2. *The Design and Analysis of Industrial Experiments*, O. L. Davies, Ed., Oliver and Boyd, London, 1954.
3. G. E. P. Box and J. S. Hunter, "The 2^{k-p} Fractional Factorial Designs, Part 1," *Technometrics* **3,** 311–351 (1961).
4. D. J. Finney, "The Fractional Replication of Factorial Arrangements," *Ann. Eugenics* **12,** 291–301 (1945).
5. R. L. Plackett and J. P. Burman, "The Design of Optimum Multifactorial Experiments," *Biometrika* **33,** 328 (1946).
6. S. Addelman and O. Kempthorne, "Orthogonal Main-effect Plans and Orthogonal Arrays of Rank 2," *Ann. Math. Statist.* **32,** 1167–1176 (1961).
7. S. Addelman, "Orthogonal Main-Effect Plans for Asymmetrical Factorial Experiments," *Technometrics* **4,** 21–46 (1962).
8. B. H. Margolin, "Non-orthogonal Main-effect Designs for Asymmetrical Factorial Experiments," *Roy. Statist. Soc. B* **34,** 431 (1972).
9. Jack P. C. Kleijnen, "Screening Designs for Poly-factor Experimentation," *Technometrics* **17,** 487 (1975).
10. B. H. Margolin, R. I. Parmelee, and M. Schatzoff, "Analysis of Free-Storage Algorithms," *IBM Syst. J.* **10,** 283–304 (1971).
11. Y. Bard, "Experimental Evaluation of System Performance," *IBM Syst. J.* **12,** 302–314 (1973).
12. T. W. Anderson, *The Statistical Analysis of Time Series*, John Wiley & Sons, Inc., New York, 1971.
13. M. Schatzoff and C. C. Tillman, "Design of Experiments in Simulator Validation," *IBM J. Res. Develop.* **19,** 252–262 (1975).
14. G. E. P. Box and D. B. Wilson, "On the Experimental Attainment of Optimum Conditions," *Roy. Statist. Soc. B* **3,** 1–45 (1951).
15. M. Schatzoff and P. G. Bryant, "Regression Methods in Perfomance Evaluation: Some Comments on the State of the Art," *Proceedings of Computer Science and Statistics, 7th Annual Symposium on the Interface*, William J. Kennedy, Ed., Iowa State University, Ames, IA, 48–57, October 18 and 19, 1973.
16. Y. Bard, "An Experimental Approach to System Tuning," *Proceedings of the International Symposium on Computer Performance Modeling, Measurement and Evaluation*, Peter P. S. Chen and Mark Franklin, Eds., Harvard University, Cambridge, MA, March 29–31, 1976.
17. Y. Bard, "Application of the Page Survival Index (PSI) to Virtual-memory System Performance," *IBM J. Res. Develop.* **19,** 212–220 (1975).

Received July 21, 1980; revised June 1, 1981

The author is located at the IBM Data Processing Division Scientific Center, 545 Technology Square, Cambridge, Massachusetts 02138.