OPTIMAL PREDICTION OF STATIONARY TIME SERIES
AND
APPLICATION IN A STOCK MARKET DECISION RULE

by

STUART ALLEN ROONEY

SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE
DEGREE OF BACHELOR OF SCIENCE
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June, 1965

Signature of Author ....
Alfred P. Sloan School of Management, May 3, 1965

Certified by ....
Thesis Supervisor

Accepted by ... .
Chairman, Departmental Committee on Theses

Professor William C. Greene
Secretary of the Faculty
Massachusetts Institute of Technology
Cambridge, Massachusetts 02139

Dear Professor Greene:

In accordance with the requirements for graduation, I herewith submit a thesis entitled "Optimal Prediction of Stationary Time Series and Application in a Stock Market Decision Rule."

I would like to thank Professor Paul H. Cootner and Robert B. Parente for their help with this work.

Sincerely yours,

This work was done in part at the Computation Center at the Massachusetts Institute of Technology, Cambridge, Massachusetts.

Optimal Prediction of Stationary Time Series and
Application in a Stock Market Decision Rule
by Stuart Allen Rooney

ABSTRACT

Submitted to the Alfred P. Sloan School of Management on May 3, 1965, in partial fulfillment of the requirements for the degree of Bachelor of Science.

This thesis encompasses all known methods of prediction and derives a general attack that will best deal with any predictive situation. Both fundamental and technical tools are considered. The standard correlation techniques are found to be optimal for fundamental prediction. Autocorrelation techniques prove far superior to averaging and smoothing methods for technical prediction. The theory evolves and is implemented in MAD for the M.I.T. Computation Center's 7094.

The theory was tested on the most conceivably difficult example, prediction of the New York Stock Exchange. Professor Paul H. Cootner has suggested that stock prices are random, or almost random, in fluctuation.[1] Yet "almost random" means in some way predictable, and with great effort it was found possible to predict NYSE prices.

The theory, tools, and degree of success of this approach are the subject of this work.

Thesis Adviser: Paul H. Cootner
Title: Associate Professor of Finance

Chapter 1

THE THEORY OF STATISTICAL PREDICTION

A general statement of the prediction problem is twofold: determine the statistics of the process, and then minimize a selected error criterion by the calculus of variations. Wiener's[2] theory, employing a squared error criterion, considers weakly stationary random time functions. Such processes are essentially characterized by their second moment properties which must exist, be continuous, and be independent of any time origin.[3] For example:

$$\overline{x(t)*x(t)} = \overline{x(t+T)*x(t+T)}$$

This thesis involves an extention of Wiener's theory to predict finite, time-discrete observations of continuous, stochastic processes[4] (in theory), industrial situations (in general), and the stock market (in particular).

Autocorrelation Functions and their Spectra:

The autocorrelative properties of a continuous function will first be stated. Then the same properties of time-discrete samples of a continuous function will be cited and their ramifications noted. The autocorrelation function is the most useful statistic to describe a stochastic process in the Wiener theory. If fc(t) is a bounded and continuous, stationary, random function of t, then its autocorrelation function is:

$$\not{0}c(x) = \lim_{T\to\infty} \frac{1}{2T} \int_{-T}^{T} fc(t)fc(t-x)dt \qquad (1A)$$

If fs(k) is a bounded, stationary, random vector of samples over t, its autocorrelation vector $\not{0}s(y)$ is:

$$\not{0}s(y) = \lim_{T\to\infty} \frac{1}{2T} \sum_{K=-T}^{T} fs(k)fs(k+y) \qquad (1B)$$

Note that the limits on the above summation are not "$-\infty \to \infty$". Business and stock market data can only be assumed stationary over an even shorter time period than their briefly recorded history. An analagous situation exists in the continuous case. There signals are often terminated at some point in past history by multiplying by a delayed unit step function, $u_{-1}(t-T)$.

To this point, the phrases "truncation" and "truncation error" have been painstakingly avoided. A nonrigorous explanation for this is: the error bears little relation to the time series truncation; it is more the change in the Fourier transform of the time series. What is lost in Shannon's[5] sense is the additional information about the statistics of the process contained in the truncated portion of the frequency domain of the series transform. If a tighter bound, dependent on the truncation of the series, could be found, an optimal vector length and sampling rate could be determined. The tightest bound I can develop is:

$$0 \leq E\left[x(t+a)*e(t)\right] \leq E\left[x(t+a)\right] \qquad (2)$$

This is only dependent upon the prediction period, a. In the stock market example as large a number of samples as possible was taken to avoid this problem.

The value of the autocorrelation function at the origin is the mean square value of the time function:

$$\phi(0) = \overline{f^2(t)} = \overline{f^2(k)} \qquad (3)$$

The value of the autocorrelation function for large arguments of the dependent variable approaches the D.C. power of the time function.

$$\phi(\infty) = \overline{f(t)}^2 = \overline{f(k)}^2 \qquad (4)$$

The greatest value of the autocorrelation function is at the origin.

$$\phi(0) \leq |\phi(T)| \qquad \forall T \qquad (5)$$

The autocorrelation function is even or symmetric about the origin.

$$\phi(-N) = \phi(N) \qquad (6)$$

This fact allows us to express the Fourier transform in terms of cosines only and introduces the next equations.

The continuous autocorrelation function and the power density spectrum are Fourier transforms of each other.

$$\phi c(T) = \int_{-\infty}^{\infty} \phi c(w) \cos wT \, dw \qquad (7A)$$

and

$$\phi c(w) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \phi c(T) \cos wT \, dT \qquad (7B)$$

Higher Order Correlates:

Similar statements can be made of all the higher order correlates. The nth order correlates of a function are the averaged nth order integral of the n + 1st order product of the shifted function. For example, the third order correlates are computed as follows:

$$\phi c(x,y,z) = \lim_{W\to\infty} \frac{1}{8W^3} \int_{-W}^{W}\int_{-W}^{W}\int_{-W}^{W} fc(t)fc(t-x)fc(t-y)fc(t-z)dxdydz \quad (8)$$

As usual, the sampled data case follows the same pattern with integrals going to summations and functions to vectors.

The Prediction Problem:

Wiener[6] suggests that the input-output relation of any nonlinear, time-variant system may be represented by a Volterra Functional Power Series. Thus:

$$\begin{aligned} y(t) = ho &+ \int_{-\infty}^{\infty} h_1(s_1)x(t-s_1)ds \\ &+ \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} h_2(s_1,s_2)x(t-s_1)x(t-s_2)ds + \ldots \\ &\ldots + \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\ldots\int_{-\infty}^{\infty} h_n(s_1,s_2,\ldots,s_n)x(t-s_1)x(t-s_2)\ldots \\ &x(t-s_n)ds\ ds \ldots ds \qquad (9) \end{aligned}$$

where x(t) is the input to the system

y(t) is the output

ho is a constant

and the kernel hm is a function of m variables and $\{hm\}$ characterizes the system.

If fi(t) is the input to a nonlinear system and fo(t) is its output, the minimization of the error, e, between input and output with respect to the characteristic kernel of the system will yield an optimal continuous predictor for the period "a" according to any selected criterion.

$$e = fo(t) - fi(t+a) \qquad (10)$$

The solution of the above for a linear, time-invariant filter yields the familiar Wiener-Hopf equation when a squared-error criterion is chosen.

$$\phi(t+a) = \int_0^\infty h(x)\phi(t-x)dx \qquad (11)$$

Parente[7] proved that the error, e = $y-\hat{y}$, between a desired output, $\hat{y}$, and an actual output, y, for a given input, x, IS A MINIMUM MEAN SQUARE ERROR REPRESENTATION IF, AND ONLY IF, ITS KERNELS $k_n(S), n \in N$ ARE SUCH THAT

$$E\left[y(t) * X^m(t-R)\right] = \sum_{n \in N} \int_{A}^{n} k_n(S)\left[E\left\{X^n(t-S)\ X^m(t-R)\right\}\ ds\right] \qquad (12)$$

for each $m \in N$ and all $R \in A$

where E denotes an ensemble average

and $X^n(t-S)$ denotes the nth order shifted product of the function X as follows:

$$X^n(t-S) = \prod_{T=1}^{n} x(t-s_i) \qquad (13)$$

and $\int_{A}^{n}$ ds denotes the nth order integral as follows:

$$\int_{A}^{n} dS = \int_{a_{n-}}^{a_{n+}} \int_{a_{n-1-}}^{a_{n-1+}} \cdots \int_{a_{2-}}^{a_{2+}} \int_{a_{1-}}^{a_{1+}} ds_1 ds_2 \ldots ds_n \qquad (14)$$

where the a's denote the bounds of the vector space

and the selected summation, n N, is as follows:

$$\sum_{n \in N} f_n = f_{n1} + f_{n2} + f_{n3} + \cdots + f_{nl} \qquad (15)$$

Equation (9) is rewritten in the above notation as a demonstration of its use.

$$y_n(t) = \sum_{n \in N} \int_{A}^{n} k_n(S) * x^n(t-S)\, dS \qquad (16)$$

The solution to the above equation (12) for some input function, x(t), and desired output, y(t) = x(t + a), yields an optimal nonlinear, time-invariant predictive filter for the period "a." The next section derives nonlinear prediction in the sampled data case. This is done under the assumption of a strictly stationary time series, a constraint which could be relaxed further.

Nonlinear Prediction:

A general expression, selected for its adaptability to the computer, for the nonlinear prediction of the next sample, x(S), of an ergodic and stationary time series from the previous M samples, x(1)...x(M), is:

$$x(S) = Ao+\sum_{K=1}^{M} A_1 (M+1-K)*x(S-K)$$

$$+\sum_{I=1}^{M}\sum_{J=1}^{M} A_2(M+1-I,\ M+1-J)*x(S-I)*x(S-J)$$

$$+\sum_{F=1}^{M}\sum_{G=1}^{M}\sum_{H=1}^{M} A_3(M+1-F,\ M+1-G,\ M+1-H)*x(S-F)*x(S-G)*x(S-H)$$

$$+\ldots \qquad (17)$$

The general expression for linear prediction in the above format is:

$$x(S) = A_0 + \sum_{K=1}^{M} A_1(M+1-K)*x(S-K) \qquad (18)$$

The general expression for quadratic prediction as above is:

$$X(S) = A_0 + \sum_{K=1}^{M} A_1(M+1-K)*x(S-K)$$

$$+\sum_{I=1}^{M}\sum_{J=1}^{M} A_2(M+1-I,\ M+1-J)*x(S-I)*x(S-J) \qquad (19)$$

In the stock market application the relative size of the predicted terms is taken as an indicator of the economic utility of considering higher order correlates in the prediction. In practice this means to try the next order prediction and see if there is significant error reduction. Also, *a priori* knowledge of the statistics of the market suggests that third order nonlinear prediction, which corresponds very roughly to the acceleration in the rate of change of the predicted price, may not be of great value. Moreover, since there are M^n equations to be solved in an M-term, nth-order prediction, it is rarely practical to examine further than the quadratic (second order) prediction case. Choosing the time average of the square error as a criterion, we next calculate the weighting coefficients (the A's). The general method is to solve each of the orders of correlation separately using only the residual data after the last phase. Thus an orthogonal functional representation of the signal is developed that is the best that can be done to the selected order of correlation.

$$A_0 = \left[\sum_{K=1}^{M} x(K) \right] / M \qquad (20)$$

Next, transform the time series vector by subtracting A , yielding a new vector with a zero mean. Applying orthogonality and limiting the problem to quadratic prediction, this new data vector has only linear and quadratic terms.

To minimize the selected criterion and select $A_1(1) \ldots A_1(M)$, we first consider the general square error term.

$$\langle e^2 \rangle = \left\langle \left[X(S) - \sum_{K=1}^{M} A_1(M+1-K) * X(S-K) \right]^2 \right\rangle \qquad (21)$$

Next we set the derivative of $\langle e^2 \rangle$ with respect to A_1 equal to zero.

$$\frac{\partial \langle e \rangle}{\partial A_1(P)} = \left\langle 2\left[X(S) - \sum_{K=1}^{M} A_1(M+1-K)*X(S-K)\right] * \left[-X(S-P)\right] \right\rangle = 0 \quad \forall P \qquad (22)$$

Then defining,

$$Y(P) = \langle X(S)*X(S+P) \rangle \qquad (23)$$

we solve

$$\frac{\partial^2 \langle e \rangle}{\partial A_1(P)^2} = 2y(0) \qquad (24)$$

Since Y(0) is always positive (square average), the values of $A_1(P)$ found as solutions to the above M simultaneous linear equations yield a minimum error according to the selected criterion.

Therefore solve:

$$Y(-P) = \sum_{K=1}^{M} A_1(M+1-K)*Y(K-P) \qquad \forall P \qquad (25)$$

To write out this solution we will change the index of the A_1's by subtracting S and adding M. We thus attempt to predict X_{MH} from M samples, X(1)...X(M), where X(M) is the most recent. The A_1 index is likewise changed. Finally, the ergodic theorem is applied and the time averages are taken as equal to the ensemble averages. For example:

$$\langle x_1 x_2 \rangle = \overline{x_1 x_2}. \qquad (26)$$

The M linear equations now take on the following form:

$$A_{1,1}\overline{X_1^2} + A_{1,2}\overline{X_1X_2} + A_{1,3}\overline{X_1X_3} + \dots + A_{1,m}\overline{X_1X_m} = \overline{X_1X_{m+1}}$$

$$A_{1,1}\overline{X_2X_1} + A_{1,2}\overline{X_2^2} + A_{1,3}\overline{X_2X_3} + \dots + A_{1,m}\overline{X_2X_m} = \overline{X_2X_{m+1}}$$

$$A_{1,1}\overline{X_3X_1} + A_{1,2}\overline{X_3X_2} + A_{1,3}\overline{X_3^2} + \dots + A_{1,m}\overline{X_3X_m} = \overline{X_3X_{m+1}}$$

$$A_{1,1}\overline{X_mX_1} + A_{1,2}\overline{X_mX_2} + A_{1,3}\overline{X_mX_3} + \dots + A_{1,m}\overline{X_m^2} = \overline{X_mX_{m+1}}$$

The solution to the above equations (27) yield the M linear coefficients of correlation, $A_{1,1} \ldots A_{1,M}$. To this point we have transformed the time series vector by subtracting A_o from each term. Now we further transform it by subtracting the linear prediction from each term. This is the previously described orthogonal approach. To proceed with quadratic prediction we must deal with the following residual time series term:

$$\chi_j = A_{i,j}(X_j - A_o) \quad . \tag{28}$$

We desire to predict the next term of this new series:

$$\hat{\chi}_{m+1} = \sum_{i=1}^{M} \sum_{j=1}^{M} A_{2,i,j}\, \chi_i \chi_j \quad . \tag{29}$$

The general squared error term is then:

$$E\left[\epsilon^2\right] = \sum_{i=1}^{M} \sum_{j=1}^{M} \sum_{k=1}^{M} \sum_{\ell=1}^{M} A_{2,i,j}\, A_{2,k,\ell}\, \overline{\chi_i \chi_j \chi_k \chi_\ell} + \overline{\chi_{m+1}^{\,2}} - 2 \sum_{i=1}^{M} \sum_{j=1}^{M} \overline{\chi_{m+1} A_{2,i,j}\, \chi_i \chi_j} \quad . \tag{30}$$

We now employ calculus to solve for the A_2's that will yield a minimum square error.

$$\frac{\partial E[\epsilon^2]}{\partial A_{2,\alpha,\beta}} = \sum_{i=1}^{M} \sum_{j=1}^{M} \sum_{k=1}^{M} \sum_{\ell=1}^{M} \overline{\chi_i \chi_j \chi_k \chi_\ell} \left(A_{2,i,j}\, \delta_{\alpha,k}\, \delta_{\beta,\ell} + A_{2,k,\ell}\, \delta_{i,\alpha}\, \delta_{j,\beta}\right) - 2 \sum_{i=1}^{M} \sum_{j=1}^{M} \delta_{i,\alpha}\, \delta_{j,\beta}\, \overline{\chi_{m+1} \chi_i \chi_j}$$

(equation continued)

$$\frac{\partial E[\epsilon^2]}{\partial A_{2,\alpha,\beta}} = \sum_{i=1}^{M} \sum_{j=1}^{M} A_{2,i,j} \overline{\chi_i \chi_j \chi_\alpha \chi_\beta}$$

$$+ \sum_{k=1}^{M} \sum_{=1}^{M} A_{2,k,\ell} \overline{\chi_k \chi_\ell \chi_\alpha \chi_\beta} - 2 \overline{\chi_{m+1} \chi_\alpha \chi_\beta}$$

Setting the derivative equal to zero:

$$2 \sum_{i=1}^{M} \sum_{j=1}^{M} A_{2,i,j} \overline{\chi_\alpha \chi_\beta \chi_i \chi_j} - 2 \overline{\chi_{m+1} \chi_\alpha \chi_\beta} = 0$$

$$1 \leq \alpha , \beta \leq M \quad .$$

Thus, the two following equations allow us to solve for the A_2's :

$$\sum_{i=1}^{M} \sum_{j=1}^{M} A_{2,i,j} \overline{\chi_i \chi_j \chi_\alpha \chi_\beta} = \overline{\chi_{m+1} \chi_\alpha \chi_\beta} \qquad 1 \leq \alpha , \beta \leq M \tag{31}$$

$$A_{2,i,j} = A_{2,j,i} \tag{32}$$

The following similar attack is the solution for the cubic case:

$$E[\epsilon^2] = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} \sum_{\ell=1}^{N} \sum_{m=1}^{N} \sum_{n=1}^{N} A_{3,i,j,k} A_{3,\ell,m,n}$$

$$\overline{\chi_i \chi_j \chi_k \chi_\ell \chi_m \chi_n} + \overline{\chi_{n+1}}^{\,2}$$

$$- 2 \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} \overline{\chi_{n+1} A_{3,i,j,k} \chi_i \chi_j \chi_k}$$

$$\frac{\partial E}{\partial A_{3,r,s,t}} = \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} \sum_{\ell=1}^{N} \sum_{m=1}^{N} \sum_{n=1}^{N} \overline{\chi_i \chi_j \chi_k \chi_\ell \chi_m \chi_n}$$

$$\left(A_{3,i,j,k}\, \delta_{r,i}\, \delta_{s,j}\, \delta_{t,k} + A_{3,\ell,m,n}\, \delta_{r,\ell}\, \delta_{s,m}\, \delta_{t,n} \right)$$

$$- 2 \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} \overline{\chi_{n+1} \chi_i \chi_j \chi_k}\, \delta_{r,i}\, \delta_{s,j}\, \delta_{t,k}$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} \overline{A_{3,i,j,k}\, \chi_i \chi_j \chi_k \chi_r \chi_s \chi_t}$$

$$+ \sum_{\ell=1}^{N} \sum_{m=1}^{N} \sum_{n=1}^{N} A_{3,\ell,m,n}\, \overline{\chi_\ell \chi_m \chi_n \chi_r \chi_s \chi_t}$$

$$- 2\, \overline{\chi_{n+1} \chi_r \chi_s \chi_t}$$

$$= 2 \sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} A_{3,i,j,k}\, \overline{\chi_i \chi_j \chi_k \chi_r \chi_s \chi_t}$$

$$- 2\, \overline{\chi_{n+1} \chi_r \chi_s \chi_t} \doteq 0$$

$$A_{3,i,j,k} = A_{3,j,i,k} = A_{3,k,i,j} = A_{3,k,j,i}$$

$$= A_{3,j,k,i} = A_{3,i,k,j} \tag{33}$$

$$\sum_{i=1}^{N} \sum_{j=1}^{N} \sum_{k=1}^{N} A_{3,i,j,k} \overline{\chi_i \chi_j \chi_k \chi_r \chi_s \chi_t} = \overline{\chi_{n+1} \chi_r \chi_s \chi_t} \quad . \tag{34}$$

The A's are thus optimally derived, under the assumptions and constraints stated, in terms of the statistics of the time series. Unfortunately, no simplifying assumptions of any great magnitude can be made past the linear case, which was solved by Wiener and Hopf. The greatest problem proved to be elimination of redundant terms among the correlates, which, if allowed to exist, caused singular matrices. A solution was reached, but not without many complications, brute force techniques, and the M. I. T. 7094.

Chapter 2

Implementation of Statistical Prediction

First, a brief word about the organization of this chapter. It begins with a general introduction to the programs printed in the appendix. These are a representative sample of the fifty to one hundred working programs developed in the course of this thesis. Then, with reference to these programs, a chronological picture of the problems and pitfalls encountered is presented. Since the programs were written in the Michigan Algorithm Decoder (MAD) without use of abbreviations, an interested reader can read them as he would English text for a full description of what was attempted.

The Programs:

NONLIN is representative of a class of foreground (real time, on-line) programs run on the Compatible Time Sharing System (CTSS).[8] This is a general predictive program not at all related to the Stock Market or any other application. It emphasizes man-machine interaction with points of rapport between the program and the user. (The program as listed in the appendix is the actual calculation flow. However, because it is so long, it will not fit all at once into the core of the 7094.) A working version of this program with enlarged dimension statements and the necessary program manipulation would have been hopelessly confusing to any reader. The technique for this manipulation will be discussed in the section on problems and pitfalls. For the present, it is sufficient to say that NONLIN is a working program that needs the handling discussed at the end of this chapter.

UTOPIA, like NONLIN, is a foreground CTSS program that needs larger dimension statements and program manipulation to produce correct answers. As it stands, UTOPIA is a complete stock market prediction program without preprocessing of data, but with man-machine communication.

PREDLN is a background (off-line, batch-processed, stacked job) program typical of those on which much of the theory was tested. It is a general prediction program and runs as it stands.

PREDQD is a background quadratic prediction program which accepts the cards punched out by PREDLN and predicts the second order case. It also works as it stands.

EXTRAP is a nonlinear multicorrelation and extrapolation program of a completely general nature. It will solve for weighting constants for up to eleven variables of regression using from linear to fourth-order fit as specified by the user. For instance, one might wish to extrapolate one variable regressed on a second linearly, a third quadratically, and a fourth cubically.

The GO card specification for this (see EXTRAP line 60) would be:

	(observations)	(regressed variables)	(orders of fit)
GO	100	3	1 2 3

SIGTST is a correlation and significance testing program that will be discussed under Prediction of Volume in Chapter 3.

PREDCT is almost the same program as PREDLN except that the preprocessing of the data is contained within it instead of being in another phase.

Problems and Pitfalls:

As it has already been noted, the most difficult problem encountered involved the redundancy inherent in the correlates. The switching circuits, lines 1690-2110 of NONLIN, and the external function ARRAY are typical of the tedious solution to this problem. In retrospect, the solution was quite simple, but it took a long time to get this area of the program debugged. If the computer treats Ø(3,2,1), Ø(2,3,1), Ø(1,2,3), etc. all as the same correlate and is programmed to read the terms of the correlate in ascending order, the redundancies that cause a singular solution matrix disappear.

Two functional methods of correlation and autocorrelation of vectors were conceived and tested. The first, dubbed a "fixed window method," can be visualized as the passing of a vector by another which is half as long and dividing the sum of the product of adjacent vector components by the length of the shorter. The amount of shift determines the term of the correlate (e.g. no shift = Ø(0)). The second method, termed "a variable window method," can be thought of as the consecutive shifting and dropping of the last term of the shorter vector. Thus, one sums the multiples of all the adjacent components of the vectors and divides by the current length of the shorter vector. If statements 0380 and 0410 of PREDLN ended in N/2 instead of N-D, the fixed window method would be programmed. The variable window method works better in all cases and is employed.

To predict more than one sampling period in the future, three methods were considered: iteration, varying the sampling rate, and adjustment of the formulae. Of these, only iteration was discarded because of the long program run time and large process error due to round-off in the 32K program. It was found that the best combination of the remaining two was independent of a theoretical error bound such as equation 2. Four 7094 hours of experimentation on hourly common stock prices showed the following table as representative of the area of predictive combinations that yield the lowest relative error in the prediction of the price for their respective prediction periods. Samples were taken on the hour from 10 a.m. to 3 p.m.

Table 1

Prediction Period	Sampling Rate	No. Samples Predicted	Mean Error
1 day	hourly	6	1%
2 days	hourly	12	1%
	bi-hourly	6	1%
3 days	bi-hourly	9	2%
	tri-hourly	6	1%
4 days	bi-hourly	12	2%
	tri-hourly	8	1%
5 days	bi-hourly	15	3%
	tri-hourly	10	2%
6 days	tri-hourly	12	3%
7 days	tri-hourly	14	3%
8 days	tri-hourly	16	4%
	daily	8	4%
9 days	tri-hourly	18	6%
	daily	9	5%
10 days	tri-hourly	20	6%
	daily	10	5%
6 months	bi-daily	64	15%
	tri-daily	42	13%
	4-daily	32	10%
	weekly	25	11%

Stationarity is not really a problem or pitfall in the present sense. When the time series becomes nonstationary, no technical method, whether chart or computer-oriented, will produce anything logical. At this time fundamental considerations must rule. There are two advantages of the computer technique over the chartists'. First, the correlation against the error is right there in EXTRAP. This program is then relieved of its "slavery" to PREDCT and PREDOD and becomes the master prediction program. Second, there is definite proof of nonstationarity as correlates begin to take on large and fast-changing values. Normally, the Ø(M)   Ø(0), and the values of the correlation vector exponentially taper off toward zero. When nonstationary effects begin, such typical behavior is destroyed.

The second major problem uncovered in this thesis seems to have been a stumbling block to previous works.[9] It seems that at least 500 to 1000 terms must be considered in calculating the correlates. Diminishing returns in the reduction of the relative error term come into effect when 2000 samples are used. Because any single correlate appears M times in the normal equations of prediction, and the computation of the correlates requires two-thirds of the total computer time, care must be taken to eliminate their recalculation whenever possible. The slow version of PREDOD includes some recalculation of the third-order correlates to obtain greater accuracy in the final prediction. The fast version of PREDOD contains no redundant calculations.

Finally, major modification of the CTSS executive routines was necessary to fit all of the programs except EXTRAP and the slow version of PREDQD into core memory.

Background operation ran with an executive routine that used unblocked input-output and possessed no Fortran II Post Mortem. The use of unblocked records in I/O required replacement of half of the normal Fortran Monitor System, and increased run time by twenty or thirty per cent.

Foreground operation was run under a one- or two-tract executive routine. Two such private commands were investigated. The first and simpler attempt combined various phases of the prediction into a master program. A file, RUNRUN BCD, was created in the following form:

```
DELETE .TAPE. 2, .TAPE. 3, .TAPE. 4, .TAPE. 5, .TAPE. 6
LOADGO PHASE1
LOADGO PHASE2
LOADGO PHASE3
LOADGO PHASE4
LOADGO PHASE5
LOADGO PHASE6
LOGOUT
```

The command RUNCOM RUNRUN[10] will cause sequential loading of the various phases. The program of each phase will call from private disk file the needed data in a pseudo tape form. If PHASE1, this is raw data; otherwise, the called pseudo tape has just been written by the previous phase. The phases, except for the data preprocessing of PHASE1, which will be discussed in the next chapter, are simply a split version of PREDCT or NONLIN:

PHASE1 - preprocessing of data

PHASE2 - calculation of linear correlates

PHASE3 - computation of linear coefficients

PHASE4 - calculation of quadratic correlates

PHASE5 - computation of quadratic coefficients

PHASE6 - error analysis

The program tends to run for over an hour, printing out notes of its progress on the console, in the course of predicting fifty times. Execution of this hour of computer time on the M.I.T. CTSS system takes approximately one day. For this reason, the program is set to chain to logout when finished. Sufficient error checks are built into the programs of the various phases that the operation is self-running.

A second faster and extremely complicated executive routine was written. This program directs the chaining of the various phases without writing out intermediate data on pseudo tapes. The chaining procedure is also different in that each phase is chained through for each prediction; that is, a partial core image must be swapped three hundred times for fifty normal predictions. Intermediate data is stored in program common and the executive routine overlays the next program phase over the last. The core image of the intermediate data is preserved between phases. This program takes fifty minutes of computer time for fifty predictions, and runs on CTSS for three hours.

Because of the length of the foreground run time, all theory was checked in background operation.

Chapter 3

SUMMARY OF STATISTICAL PREDICTION IN VIEW OF APPLICATION

In General:

There are two sides to any prediction, the fundamental and the technical. To proceed without a full knowledge of of both is only half preparation. A technical prediction is produced by looking back over past variations of the signal to be predicted and determining those characteristics of the signal that are innately identified with it. A fundamental prediction is developed by looking back over past variations of the signal to be predicted and noting how external signals correlate with these variations.

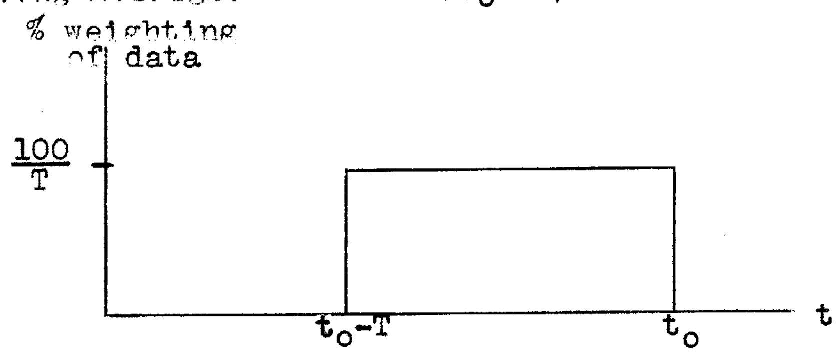
Programming a fundamental prediction by nonlinear multicorrelation techniques is quite simple. Nevertheless, EXTRAP is the first attempt at a completely general fundamental approach. It will solve for regression on an arbitrary number of variables, each considered to arbitrary (programmer selected) powers of fit (e.g. linear, quadratic, cubic, etc.)

General statistical prediction, however, is not easily programmed. For this reason other methods such as exponential smoothing and moving average techniques have been used in the past to achieve technical prediction.

Whether one is viewing the stock market, sales volumes, or inventory levels, smoothing and averaging techniques have no basis in fact. They are easy-to-use mathematical crutches that give fairly logical answers, which relieve the user of the responsibility of prediction. Nevertheless, the human mind can usually produce a far more accurate prediction in less time and at a lower cost than these tools.

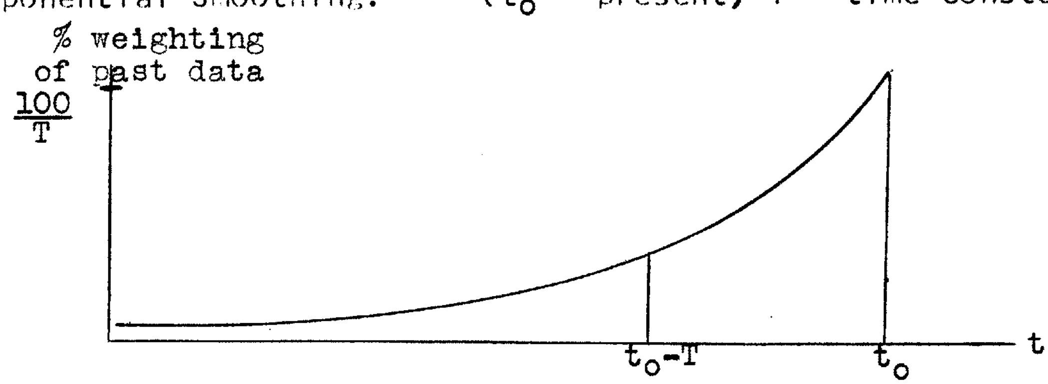
Another conceptual view of technical prediction is the weighting of past data. The above techniques can be visualized as follows:

Moving Average: (t_o = present time)



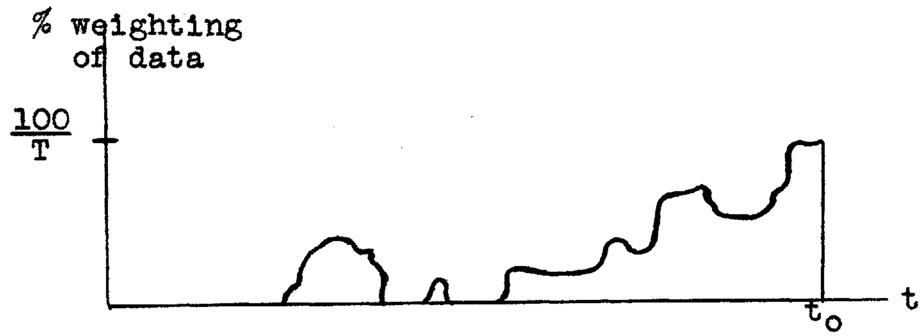
Moving averages equally weight past data over a metaphysically determined and mystically significant period. This is a bold statement but is usually correct. Sometimes the period is theoretically determined, as a twelve month period is correct for smoothing seasonal effects.

Exponential Smoothing: (t_o = present, T = time constant)



Exponential smoothing weights past data exponentially with an empirically-determined time constant.

There is really no reason to believe that the value of past data in predicting future data should be such a clear and easy thing. In fact, one would intuitively expect that an optimal weighting of past data would be quite messy and different in every case. Example:



One thing that is clear from these graphs is that all the time series that they attempt to technically predict should be stationary; that is, time independent. Thus no pure technical method will be able to predict a pure linear, quadratic, etc. trend, since a trend by its nature is a function of time. The reason for this is that the total area under any of the weighting curves is one or one hundred per cent, depending on semantic terms used. Even if all the weight is put on the last sample, that is the largest number that may be predicted. Thus, in general:

$$x_{n+1} \leq \{x\} \quad (35)$$

DATA PREPROCESSING:

This leads us to the subject of preprocessing of the data. Many types of preprocessing were tried. It was attempted to predict price, the first difference of the price, the second difference of the price, and the relative change in the price. Of these, the relative change in the price was found to be the best predictor of the market, but this too was unable to predict a pure trend. This is clear since this type of preprocessing will always leave some time-dependent action. It was finally decided to further preprocess the relative change in the price by mathematically eliminating both linear and quadratic functions of time. This brought about satisfactory results and became a permanent preprocessing scheme.

We can go further: if we would intuitively expect an "optimal" technical scheme to have a messy weighting function of past data, why would we not expect it to also have a messy weighting function of all combinations of past data? Thus it is so, and from this start the problem was attacked.

In View of the New York Stock Exchange:

UTOPIA may be a program in the appendix, but is is only in jest that it can be suggested that utopia has arrived. Originally, this program had following it a simple profit-maximizing decision rule, but this is beyond the sophistication of the program. The following is therefore a review of what was accomplished in the light of the chosen application.

Prediction of Volume:

I had originally hoped to build my principal prediction upon price and volume, but this method produced lower profits than dealing with price alone. When I was confronted with these results, I set about to test my original ideas of how the market works. It was believed that high volume and increasing price signified a strong market, and so forth. In other words, I believed in the concepts of accumulation and distribution. SIGTST, a significance testing program, was developed to correlate a future high positive rate of change of price to a present high rate of change of price and high volume.

In terms of SIGTST:

X = % change in price for one period a in advance

Y = % change in price for one period at the current rate

Z = present volume level

The multiple correlation coefficients of the above ranged from -.1 to +.5 for various predictive periods (a) for both strong and weak markets, even when only the first and fourth quartiles were used. But while there was correlation, it was not enough to yield a profit. None of these predictive ranges were significant at the .1 level in an F-test.

Burst Error Effects:

I have used the term "burst error" to describe the phenomenon whereby the predictive tool tends to make a large number of errors at one time. The following is a graphical sketch of this effect:

The assumption at the outset of this investigation was that lack of improvement in these burst error effects would be taken as an indication of a nonstationary time series. While quadratic prediction lowered the mean error of prediction by approximately 30% of the residual error, there was NO reduction of the error bursts. It may be concluded that no form of prediction would reduce such errors as they are the product of earnings announcements, President Kennedy's death, and the like. Furthermore, these burst errors account for almost 50% of the residual error after quadratic prediction, indicating additional time spent in technical prediction would be to little avail.

Significance of Results:

Father Time plagued this thesis as well as most others. Only one run of fifty predictions could be made for each of the combinations of prediction period and sampling rate of Table 1. Some of these runs included only 25 predictions. Two runs were made on the one day prediction period. The mean error is a monotonically increasing function of the prediction period ranging from one to ten per cent over the range studied. The error was automatically tabulated by the prediction program as follows:

ERROR = ERROR $+\sum(\hat{x}(k) - x(k))/x(k)$

ERROR = ERROR/NUMBER OF PERIODS PREDICTED

where k was indexed as the prediction proceeded in other words:

$$\text{ERROR} = \frac{\sum \frac{\hat{x}(k)-x(k)}{x(k)}}{\#\ \text{periods predicted}}. \qquad (36)$$

After these runs had been made, an improved technique allowed inclusion of many times as much data and generated even better results. This was accomplished by not dropping any samples when going to a bi- or tri-hourly sampling rate. The calculation of the correlates is then done by shifting two or three units instead of one. While giving less error, these results seem incorrect from a pure mathematical viewpoint. Holbrook Working[11] has shown that averaging in calculation of correlates gives artificial results.

Pragmatically, this technique reduced error in the six month prediction case by fifty per cent. Later, Professor Cootner pointed out that in fact what was done was correct in that averaging was carried out after calculation of the unnormalized correlates. An example of program modification which accomplishes the above in <u>Nonlin</u> at line 580:

```
THROUGH ENDD, FOR D = 0,2,D.G.2*M   (bihourly)
THROUGH ENDD, FOR D = 0,3,D.G.3*M   (trihourly)
```

<u>Summary:</u>

Optimal prediction has been derived in theory and has been implemented. Theoretically, this is the best that can be done, but the theory seems almost mystical in application except through the concepts of electrical engineering. The best explanation I can give of what was done is that Wiener's work for a linear predictive filter has been modified to solve for a nonlinear predictive filter with sampled-data inputs.

This predictive method is technical in nature but is different from the traditional technical approach to stock analysis. It is true that little "feeling" is developed for the nature of the market, but in trade for this loss of feeling, much of the human adjustment of data is lost. One of the greatest weaknesses of the human mind in stock analysis is that it is subject to affect completely unrelated to the market. Of course, the great advantage is its relative low cost.

When the computer makes a decision about the market in one case, you can bet if those conditions ever exist again, it will give the same answer. This fact is a major advantage of this method of technical analysis, because the output of the analysis then has an error history which may be fundamentally correlated to other events that effect the market. The residual error after a technician's analysis may be as small as the computer error, but the technician's error is more likely to be a function of his rose-colored glasses than the market itself.

If the reader does not understand the theory of Chapter 1, perhaps the following words of explanation will give some insight to the technique. The stock market analyst expects market history to repeat itself in two ways, technically and fundamentally. He is as lost as the computer if this does not happen. For instance, neither the computer not the technician could have predicted President Kennedy's death, and both would have had large errors that day. The theory of Chapter 1 claims not perfect prediction, but a minimum of error. This method of prediction works by extending <u>all the statistics</u> of the past market and price fluctuations, expecting history to repeat itself. The statistics which are extended are both technical and fundamental. Loss of either would be a great loss of information.

The following is a sketch of the general procedure:

Section One:Preprocessing of Data (PREDCT)

Phase One: Calculation of relative price changes over time

Phase Two: Extraction of any linear trend with time

Phase Three: Extraction of any quadratic trend with time

Section Two: Linear Prediction (PREDLN)

Phase One: Calculation of the Correlates

Phase Two: Construction of Matrix and its inversion

Phase Three: Preprocessing of data for next phase

Section Three: Quadratic Predition (PREDQD)

Phase One: Calculation of correlates

Phase Two: Construction of Matrix and its inversion

Phase Three: Calculation of Prediction

Section Four: Insertion of prediction into running error analysis

Section Five: Fundamental correlation of error with market (EXTRAP)

Section Six: Insertion of prediction into running error analysis

What was done is not related to correlation techniques, harmonic analysis, orthonomal functionals alone; it encompasses all these techniques. To quell a few misconceptions, linear correlation is in no way related to linear prediction. Linear correlation considers only the first and second moments of any distribution. Linear prediction considers the first one hundred as programmed, all in theory. Linear prediction is similar to harmonic analysis, however, and will give the same answer when given the same data to assimilate. Nonlinear prediction builds up knowledge of past market history in orthonomal functionals and is the complete representation of a stock's history in a single formula.

With final reference to Professor Cootner's work, all that can be predicted from past history is stationary time signals--that is, that things will happen in the future as they did in the past. Using any error criterion, the technique of nonlinear prediction is the only complete mathematical or functional representation of history. I contend that Professor Cootner is right; the stock market is almost a random process, but not quite. On the floor of the exchange the brokers will only conduct about one million transactions per hour before they will shut down the exchange. Thus, high frequency effects have been eliminated. In effect, we are trying to predict band-limited noise or a band-limited random process. The technique of prediction contained herein is the optimal prediction of band-limited noise. I refer the interested reader to Y. W. Lee and C. A. Stutt,[12] "Statistical Prediction of Noise," M.I.T. R.L.E. Report #102, for a discussion of this in electrical engineering terms.

APPENDIX

```
R     NONLIN - FOREGROUND NONLINEAR PREDICTION PROGRAM
       NORMAL MODE IS INTEGER
       FLOATING POINT X,X1,X2,XA1,XA2,XB1,XB2,Z,LINMAT,CONVRT,
      1 A0,A01,A0A,A0Q,AVPRD1,AVPRD2,ANSWER,ARRAY,
      2 VECTOR,LIN1,LIN2,A02
       DIMENSION X(100),X1(100),X2(100),XA1(100),XA2(100),Z(100)
       DIMENSION VECTOR(100),XB1(100),XB2(100)
       DIMENSION LINMAT(784,L1DIM),CONVRT(784,L1DIM)
       VECTOR VALUES L1DIM = 2,1,28
       VECTOR VALUES L2DIM = 2,1,1
       DIMENSION LIN1(28,L2DIM), LIN2(28,L1DIM)
       DIMENSION ARRAY(9261,ARY1)
       VECTOR VALUES ARY1 = 3,1,21,21
       PROGRAM COMMON ARRAY
       VECTOR VALUES STRING = $ 81H THE OUTCOME OF THE PREDICTION IS
      1 DOUBTFUL. CAN YOU SUPPLY MORE DATA OR REDUCE M.*$
HERE   EXECUTE SETBRK.(HERE)
START  PRINT COMMENT $ NONLINEAR PREDICTION - STUART A. ROONEY $
       OFF = 0
       I = 1
       THROUGH ENDA, FOR A = 0,1,A.G.100
       X(A) = 0.
       X1(A) = 0.
       X2(A) = 0.
       XA1(A) = 0.
       XA2(A) = 0.
ENDA   Z(A) = 0.
       PRINT COMMENT $ INPUT DATA $
       READ DATA
       WHENEVER N.G.P-1
       PRINT COMMENT $ TUFF LUCK,HUCK,IT CANNOT BE DONE. TRY AGAIN.$
       TRANSFER TO START
       OR WHENEVER N.L.4*M/3 .OR. N-M.L.4
GOOF   PRINT FORMAT STRING
       READ FORMAT $A3*$, TYPEIN
       WHENEVER TYPEIN.E.$YES$
       TRANSFER TO START
       OR WHENEVER TYPEIN.E.$NO$
       PRINT COMMENT $ GOOD LUCK, HUCK. AWAY WE GO. $
       TRANSFER TO GOGOGO
       OTHERWISE
       TRANSFER TO GOOF
       END OF CONDITIONAL
       OTHERWISE
       TRANSFER TO GOGOGO
       END OF CONDITIONAL
GOGOGO S = 0
       A0A = 0.
       A01 = (X(P) - X(P-N))/N
       THROUGH ENDB, FOR B = 1,1,B.G.N
ENDB   A0A = A0A + X(P-N+B)
       A0 = A0A/N
       PRINT COMMENT $4ALPHA ZERO $
       PRINT RESULTS A0, A01
       THROUGH ENDC, FOR C = 1,1,C.G.N
```

1

```
          X2(C) = X(P-N+C) -A0
ENDC      XA1(C) = X2(C)
RETURN    THROUGH ENDD, FOR D = 0,1,D.G.M
          THROUGH ENDE, FOR E = 0,1,E.G.100
ENDE      X1(E) = 0.
          THROUGH ENDF, FOR F = 1,1,F.G.N-D
ENDF      X1(F) = X2(F) * X2(F+D)
          Z(D) = 0.
          THROUGH ENDG, FOR G = 1,1,G.G.N-D
ENDG      Z(D) = Z(D) + X1(G)
ENDD      Z(D) = Z(D)/(N-D)
          WHENEVER S.E.0
          TRANSFER TO RAW
          OR WHENEVER S.E.1
          TRANSFER TO ONE
          OR WHENEVER S.E.2
          TRANSFER TO TWO
          END OF CONDITIONAL
RAW       PRINT COMMENT $4AUTOCORRELATION OF THE DATA $
          PRINT RESULTS Z(0)...Z(M)
          THROUGH ENDKK,FOR KK = 0,1,KK.G.M
ENDKK     XB1(KK) = Z(KK)
          S = 1
          THROUGH ENDH, FOR H = 1,1,H.G.N
          XA2(H) = X(P-N+H) - X(P-N+H-1) - A01
ENDH      X2(H) = XA2(H)
          TRANSFER TO RETURN
ONE       PRINT COMMENT $4AUTOCORRELATION OF THE FIRST DIFFERENCES $
          PRINT RESULTS Z(0)...Z(M)
          THROUGH ENDLL, FOR LL = 0,1,LL.G.M
ENDLL     XB2(LL) = Z(LL)
          S = 2
          THROUGH ENDR, FOR R = 1,1,R.G.N
ENDR      X2(R) = X(P-N+R) - 2*X(P-N+R-1) + X(P-N+R-2)
          TRANSFER TO RETURN
TWO       PRINT COMMENT $4AUTOCORRELATION OF THE SECOND DIFFERENCES $
          SWITCH = 0
SECOND    THROUGH ENDT, FOR T = 0,1,T.G.100
          X(T) = 0.
          X1(T) = 0.
          X2(T) = 0.
ENDT      Z(T) = 0.
          WHENEVER SWITCH.E.0
          A0Q = A0
          THROUGH ENDU,FOR U = 1,1,U.G.N
ENDU      Z(U) = XA1(U)
          THROUGH ENDV, FOR V = 0,1,V.G.M
ENDV      X(V) = XB1(V)
          OR WHENEVER SWITCH/2 + SWITCH/2.NE. SWITCH
          A0Q = A01
          THROUGH ENDW, FOR W = 1,1,W.G.N
ENDW      Z(W) = XA2(W)
          THROUGH ENDAA, FOR AA = 0,1,AA.G.M
```

```
ENDAA      X(AA) = XB2(AA)
           THROUGH ENDBB, FOR BB = 0,1,BB.G.M-1
ENDBB      XA1(BB) = XA1(BB+1)
           XA1(M) = AVPRD1 + AVPRD2
           OR WHENEVER SWITCH/2 + SWITCH/2 .E. SWITCH
           A0Q = A0
           THROUGH ENDMM, FOR MM = 1,1,MM.G.N
ENDMM      Z(MM) = XA1(MM)
           THROUGH ENDNN, FOR NN = 0,1,NN.G.M
ENDNN      X(NN) = XB1(NN)
           THROUGH ENDPP, FOR PP = 0,1,PP.G.M-1
ENDPP      XA2(PP) = XA2(PP+1)
           XA2(M) = AVPRD1 + AVPRD2
           END OF CONDITIONAL
           SWITCH = SWITCH + 1
           THROUGH ENDCC, FOR CC = 1,1,CC.G.M
           LIN1(CC,1) = 1.0
ENDCC      LIN2(1,CC) = X(CC - 1)
           THROUGH ENDDEF, FOR DD = 1,1,DD.G.M
           THROUGH ENDDEF, FOR EE = 1,1,EE.G.M
           CONVRT(DD,EE) = 0.
           THROUGH ENDDEF, FOR FF = 1,1,FF.G.1
ENDDEF     CONVRT(DD,EE)=CONVRT(DD,EE)+LIN1(DD,FF)*LIN2(FF,EE)
           THROUGH ENDGH, FOR GG = 1,1,GG.G.M
           THROUGH ENDGH, FOR HH = 1,1,HH.G.M
           WHENEVER GG.LE.HH
           LINMAT(GG,HH) = CONVRT(GG,HH - GG +1)
           OTHERWISE
           LINMAT(GG,HH) = CONVRT(GG,GG - HH +1)
           END OF CONDITIONAL
ENDGH      CONTINUE
           THROUGH ENDII, FOR II = 1,1,II.G.M
ENDII      CONVRT(II,1) = X(II)
           SCALE = 1.0
           TEST1 = XSMEQ.(28,M,1,LINMAT,CONVRT,SCALE,X1)
           PRINT RESULTS TEST1
           AVPRD1 = 0.
           AVPRD2 = 0.
           A02 = 0.
           THROUGH ENDJJ, FOR JJ = 1,1,JJ.G.M
           X(JJ) = LINMAT(M+1-JJ,1) * Z(N-M+JJ)
           PRINT RESULTS LINMAT(M+1-JJ,1),Z(N-M+JJ),X(JJ)
ENDJJ      AVPRD1 = AVPRD1 + X(JJ)
           PRINT RESULTS AVPRD1
           ANSWER = A0Q + AVPRD1
           PRINT RESULTS ANSWER
           THROUGH ENDQQ, FOR QQ = 1,1,QQ.G.M
           X(QQ) = Z(N-M+QQ) - X(QQ)
ENDQQ      A02 = A02 + X(QQ)
           THROUGH ENDT2, FOR T2 = 1,1,T2.G.M
ENDT2      X(T2) = X(T2) - A02/M
           PRINT RESULTS X(1)...X(M)
           COUNTR = L*(L+1)/2 + 1
```

```
          THROUGH ENDAB1, FOR A1 = 1,1,A1.G.L
          THROUGH ENDAB1, FOR B1 = 1,1,B1.G.A1
          LINMAT(A1,B1) = 0.
          THROUGH ENDC1, FOR C1 = 1,1,C1.G.M-A1
ENDC1     LINMAT(A1,B1) = LINMAT(A1,B1) + X(C1)*X(C1+A1)*X(C1+A1-B1)
          COUNTR = COUNTR - 1
ENDAB1    CONVRT(COUNTR,1) = LINMAT(A1,B1)/((M-A1)*(M-A1))
          INT = L*(L+1)/2
          THROUGH ENDAB2, FOR A2 = 1,1,A2.G.L
          THROUGH ENDAB2, FOR B2 = 1,1,B2.G.A2
          THROUGH ENDAB2, FOR AB = 1,1,AB.G.B2
          ARRAY(A2,B2,AB) = 0.
          WHENEVER A2.GE.B2 .AND. A2.GE.AB
          AC = M - A2 + 1
          OR WHENEVER B2.G.A2 .AND. B2.G.AB
          AC = M - B2 + 1
          OR WHENEVER AB.G.A2 .AND. AB.G.B2
          AC = M - AB + 1
          END OF CONDITIONAL
          THROUGH ENDC2, FOR C2 = 1,1,C2.G.AC
ENDC2     ARRAY(A2,B2,AB) = ARRAY(A2,B2,AB) +
         1 X(C2)*X(C2+AB-1)*X(C2+A2-1)*X(C2+B2-1)
          ARRAY(A2,B2,AB) = ARRAY(A2,B2,AB)/(AC*AC*AC)
          WHENEVER A2.GE.B2 .AND. B2.GE.AB
          ARRAY(A2,AB,B2) = ARRAY(A2,B2,AB)
          ARRAY(B2,AB,A2) = ARRAY(A2,B2,AB)
          ARRAY(B2,A2,AB) = ARRAY(A2,B2,AB)
          ARRAY(AB,A2,B2) = ARRAY(A2,B2,AB)
          ARRAY(AB,B2,A2) = ARRAY(A2,B2,AB)
          END OF CONDITIONAL
ENDAB2    CONTINUE
          SS = 1
          SSS = -1
          THROUGH ENDFED, FOR D2 = 1,1,D2.G.INT
          WHENEVER SSS.E.L-SS
          SSS = -1
          SS = SS + 1
          END OF CONDITIONAL
          SSS = SSS + 1
          SSSS = 0
          THROUGH ENDFED, FOR E2 = 1,1,E2.G.L
          THROUGH ENDFED, FOR F2 = 1,1,F2.G.L
          WHENEVER E2.E.F2
          SSSS = SSSS + 1
          LINMAT(D2,SSSS) = ARRAY(SSS+1,.ABS.(E2-SSS-SS)+1,1)
          OR WHENEVER E2.L.F2
          SSSS = SSSS + 1
          LINMAT(D2,SSSS) = ARRAY(SSS+1,.ABS.(E2-SSS-SS)+1,F2-E2+1)
         1 + ARRAY(SSS+1,.ABS.(F2-SSS-SS)+1,F2-E2+1)
          END OF CONDITIONAL
ENDFED    CONTINUE
          SCALE = 1.0
          TEST2 = XSMEQ.(28,INT,1,LINMAT,CONVRT,SCALE,X1)
```

4

```
          PRINT RESULTS TEST2
          THROUGH ENDQ1, FOR Q1 = 1,1,Q1.G.L
          LIN1(Q1,1) = X(M-L+Q1)
ENDQ1     LIN2(1,Q1) = X(M-L+Q1)
          THROUGH ENDRTU,FOR RR = 1,1,RR.G.L
          THROUGH ENDRTU, FOR TT = 1,1,TT.G.L
          CONVRT(RR,TT) = 0.
          THROUGH ENDRTU, FOR UU = 1,1,UU.G.1
ENDRTU    CONVRT(RR,TT) = CONVRT(RR,TT)+LIN1(RR,UU)*LIN2(UU,TT)
          COUNTR = 0
          THROUGH ENDSTU, FOR STU = 1,1,STU.G.L
          THROUGH ENDSTU, FOR ROO = 1,1,ROO.G.L
          WHENEVER STU.L.ROO
          COUNTR = COUNTR +1
          VECTOR(COUNTR) = 2*CONVRT(STU,ROO)
          OR WHENEVER STU.E.ROO
          COUNTR = COUNTR + 1
          VECTOR(COUNTR) = CONVRT(STU,ROO)
          END OF CONDITIONAL
ENDSTU    CONTINUE
          THROUGH ENDEND, FOR END = 1,1,END.G.INT
          PRINT RESULTS VECTOR(END),LINMAT(END,1)
ENDEND    AVPRD2 = AVPRD2 + VECTOR(END)*LINMAT(END,1)
          PRINT RESULTS AVPRD2
          ANSWER = AOQ + AVPRD1 + AVPRD2
          PRINT RESULTS ANSWER
          WHENEVER SWITCH/2 + SWITCH/2.NE.SWITCH
          PRINT COMMENT $1PREDICTION OF THE NEXT FIRST DIFFERENCE $
          TRANSFER TO SECOND
          OR WHENEVER SWITCH/2.NE.I
          TRANSFER TO SECOND
          OR WHENEVER SWITCH/2.E.I.AND. OFF.E.0
          TRANSFER TO START
          OR WHENEVER SWITCH/2.E.I.AND.OFF.E.1
          EXECUTE EXIT.
          END OF CONDITIONAL
          END OF PROGRAM
                                                                    251

                                                          TOTAL     251
```

```
R     UTOPIA - STOCK MARKET DECISION RULE PROGRAM
       NORMAL MODE IS INTEGER
       FLOATING POINT X,X1,X2,Z,LINMAT,CONVRT,A01,A02,AVPRD1,AVPRD2,
      1 ANSWER,ARRAY,VECTOR,LIN1,LIN2,INPUT,OUTPUT
       DIMENSION X(100),X1(100),X2(100),Z(100),VECTOR(100)
       DIMENSION LINMAT(784,L1DIM),CONVRT(784,L1DIM)
       VECTOR VALUES L1DIM = 2,1,28
       VECTOR VALUES L2DIM = 2,1,1
       DIMENSION LIN1(28,L2DIM), LIN2(28,L1DIM)
       DIMENSION INPUT(185,INDIM),OUTPUT(50,OUTDIM)
       VECTOR VALUES INDIM = 2,1,37
       VECTOR VALUES OUTDIM = 2,1,10
       DIMENSION ARRAY(9261,ARY1)
       VECTOR VALUES ARY1 = 3,1,21,21
       PROGRAM COMMON ARRAY
HERE   EXECUTE SETBRK.(HERE)
       PRINT COMMENT $1 STOCK MARKET DECISION RULE PROGRAM$
       PRINT COMMENT $ STUART A. ROONEY$
       L = 6
       M = 13
       N = 27
       P = 28
       NSTOCK = 5
       PERIOD = 10
       PRINT COMMENT $ INPUT DATA $
       READ DATA
       THROUGH FINISH, FOR S = 1,1,S.G.NSTOCK
       TRIGER = 1
       THROUGH BITTER, FOR I = 1,1,I.G.PERIOD
       THROUGH ENDA, FOR A = 0,1,A.G.100
       X1(A) = 0.
       X2(A) = 0.
ENDA   Z(A) = 0.
       WHENEVER TRIGER.E.1
       TRIGER = 0
       THROUGH ENDB, FOR B = 1,1,B.G.P
ENDB   X(B) = INPUT(S,B)
       TRANSFER TO AGAIN
       OTHERWISE
       THROUGH ENDC, FOR C = 1,1,C.G.P-1
ENDC   X(C) = X(C+1)
       X(P) = INPUT(S,P+I-1)
       END OF CONDITIONAL
AGAIN  A01 = (X(P) - X(P-N))/N
       PRINT COMMENT $4ALPHA ZERO $
       PRINT RESULTS A01
       THROUGH ENDH, FOR H = 1,1,H.G.N
ENDH   X2(H) = X(P-N+H) - X(P-N+H-1) -A01
       THROUGH ENDD, FOR D = 0,1,D.G.M
       THROUGH ENDE, FOR E = 0,1,E.G.100
ENDE   X1(E) = 0.
       THROUGH ENDF, FOR F = 1,1,F.G.N-D
ENDF   X1(F) = X2(F) * X2(F+D)
       Z(D) = 0.
       THROUGH ENDG, FOR G = 1,1,G.G.N-D
```

1

```
ENDG        Z(D) = Z(D) + X1(G)
ENDD        Z(D) = Z(D)/(N-D)
            PRINT COMMENT $4AUTOCORRELATION OF THE FIRST DIFFERENCES $
            PRINT RESULTS Z(0)...Z(M)
            THROUGH ENDT, FOR T = 0,1,T.G.100
ENDT        X1(T) = 0.
            THROUGH ENDCC, FOR CC = 1,1,CC.G.M
            LIN1(CC,1) = 1.0
ENDCC       LIN2(1,CC) = Z(CC - 1)
            THROUGH ENDDEF, FOR DD = 1,1,DD.G.M
            THROUGH ENDDEF, FOR EE = 1,1,EE.G.M
            CONVRT(DD,EE) = 0.
            THROUGH ENDDEF, FOR FF = 1,1,FF.G.1
ENDDEF      CONVRT(DD,EE)=CONVRT(DD,EE)+LIN1(DD,FF)*LIN2(FF,EE)
            THROUGH ENDGH, FOR GG = 1,1,GG.G.M
            THROUGH ENDGH, FOR HH = 1,1,HH.G.M
            WHENEVER GG.LE.HH
            LINMAT(GG,HH) = CONVRT(GG,HH - GG +1)
            OTHERWISE
            LINMAT(GG,HH) = CONVRT(GG,GG - HH +1)
            END OF CONDITIONAL
ENDGH       CONTINUE
            THROUGH ENDII, FOR II = 1,1,II.G.M
ENDII       CONVRT(II,1) = Z(II)
            SCALE = 1.0
            TEST1 = XSMEQ.(28,M,1,LINMAT,CONVRT,SCALE,X1)
            PRINT RESULTS TEST1
            AVPRD1 = 0.
            AVPRD2 = 0.
            A02 = 0.
            THROUGH ENDJJ, FOR JJ = 1,1,JJ.G.M
            X1(JJ) = LINMAT(M+1-JJ,1) * X2(N-M+JJ)
            PRINT RESULTS LINMAT(M+1-JJ,1),X2(N-M+JJ),X1(JJ)
ENDJJ       AVPRD1 = AVPRD1 + X1(JJ)
            PRINT RESULTS AVPRD1
            ANSWER = A01 + AVPRD1
            PRINT RESULTS ANSWER
            THROUGH ENDQQ, FOR QQ = 1,1,QQ.G.M
            X1(QQ) = X2(N-M+QQ) - X1(QQ)
ENDQQ       A02 = A02 +X1(QQ)
            THROUGH ENDT2, FOR T2 = 1,1,T2.G.M
ENDT2       X1(T2) = X1(T2) - A02/M
            PRINT RESULTS X1(1)...X1(M)
            COUNTR = L*(L+1)/2 + 1
            THROUGH ENDAB1, FOR A1 = 1,1,A1.G.L
            THROUGH ENDAB1, FOR B1 = 1,1,B1.G.A1
            LINMAT(A1,B1) = 0.
            THROUGH ENDC1, FOR C1 = 1,1,C1.G.M-A1
ENDC1       LINMAT(A1,B1) = LINMAT(A1,B1) + X1(C1)*X1(C1+A1)*X1(C1+A1-B1)
            COUNTR = COUNTR - 1
ENDAB1      CONVRT(COUNTR,1) = LINMAT(A1,B1)/((M-A1)*(M-A1))
            INT = L*(L+1)/2
            THROUGH ENDAB2, FOR A2 = 1,1,A2.G.L
```

```
            THROUGH ENDAB2, FOR B2 = 1,1,B2.G.A2
            THROUGH ENDAB2, FOR AB = 1,1,AB.G.B2
            ARRAY(A2,B2,AB) = 0.
            AC = M - A2 + 1
            THROUGH ENDC2, FOR C2 = 1,1,C2.G.AC
ENDC2       ARRAY(A2,B2,AB) = ARRAY(A2,B2,AB) +
           1 X1(C2)*X1(C2+AB-1)*X1(C2+A2-1)*X1(C2+B2-1)
            ARRAY(A2,B2,AB) = ARRAY(A2,B2,AB)/(AC*AC*AC)
            WHENEVER A2.GE.B2 .AND. B2.GE.AB
            ARRAY(A2,AB,B2) = ARRAY(A2,B2,AB)
            ARRAY(B2,AB,A2) = ARRAY(A2,B2,AB)
            ARRAY(B2,A2,AB) = ARRAY(A2,B2,AB)
            ARRAY(AB,A2,B2) = ARRAY(A2,B2,AB)
            ARRAY(AB,B2,A2) = ARRAY(A2,B2,AB)
            END OF CONDITIONAL
ENDAB2      CONTINUE
            SS = 1
            SSS = -1
            THROUGH ENDFED, FOR D2 = 1,1,D2.G.INT
            WHENEVER SSS.E.L-SS
            SSS = -1
            SS = SS + 1
            END OF CONDITIONAL
            SSS = SSS + 1
            SSSS = 0
            THROUGH ENDFED, FOR E2 = 1,1,E2.G.L
            THROUGH ENDFED, FOR F2 = 1,1,F2.G.L
            WHENEVER E2.E.F2
            SSSS = SSSS + 1
            LINMAT(D2,SSSS) = ARRAY(SSS+1,.ABS.(E2-SSS-SS)+1,1)
            OR WHENEVER E2.L.F2
            SSSS = SSSS + 1
            LINMAT(D2,SSSS) = ARRAY(SSS+1,.ABS.(E2-SSS-SS)+1,F2-E2+1)
           1 + ARRAY(SSS+1,.ABS.(F2-SSS-SS)+1,F2-E2+1)
            END OF CONDITIONAL
ENDFED      CONTINUE
            SCALE = 1.0
            TEST2 = XSMEQ.(28,INT,1,LINMAT,CONVRT,SCALE,VECTOR)
            PRINT RESULTS TEST2
            THROUGH ENDQ1, FOR Q1 = 1,1,Q1.G.L
            LIN1(Q1,1) =X1(M-L+Q1)
ENDQ1       LIN2(1,Q1) =X1(M-L+Q1)
            THROUGH ENDRTU,FOR RR = 1,1,RR.G.L
            THROUGH ENDRTU, FOR TT = 1,1,TT.G.L
            CONVRT(RR,TT) = 0.
            THROUGH ENDRTU, FOR UU = 1,1,UU.G.1
ENDRTU      CONVRT(RR,TT) = CONVRT(RR,TT)+LIN1(RR,UU)*LIN2(UU,TT)
            COUNTR = 0
            THROUGH ENDSTU, FOR STU = 1,1,STU.G.L
            THROUGH ENDSTU, FOR ROO = 1,1,ROO.G.L
            WHENEVER STU.L.ROO
            COUNTR = COUNTR +1
            VECTOR(COUNTR) = 2*CONVRT(STU,ROO)
```

3

```
          OR WHENEVER STU.E.ROO
          COUNTR = COUNTR + 1
          VECTOR(COUNTR) = CONVRT(STU,ROO)
          END OF CONDITIONAL
ENDSTU    CONTINUE
          THROUGH ENDEND, FOR END = 1,1,END.G.INT
          PRINT RESULTS VECTOR(END),LINMAT(END,1)
ENDEND    AVPRD2 = AVPRD2 + VECTOR(END)*LINMAT(END,1)
          PRINT RESULTS AVPRD2
          OUTPUT(S,I) = A01 + AVPRD1 + AVPRD2
          PRINT RESULTS OUTPUT(S,I)
BITTER    CONTINUE
FINISH    CONTINUE
          PRINT COMMENT $1 RESULTS$
          THROUGH ENDPNT, FOR PNT = 1,1,PNT.G.NSTOCK
ENDPNT    PRINT RESULTS OUTPUT(PNT,1)...OUTPUT(PNT,PERIOD)
          PRINT COMMENT $1 THAT IS ALL.$
          EXECUTE EXIT.
          END OF PROGRAM
                                                                  180

                                                        TOTAL     180
```

```
          R    PREDLN - LINEAR PREDICTION PROGRAM
           NORMAL MODE IS INTEGER
           FLOATING POINT X,X1,X2,XA1,XA2,XB1,XB2,Z,LINMAT,CONVRT,
          1 A0,A01,A0A,A0Q,AVPRD1,ANSWER,LIN1,LIN2
           DIMENSION X(500),X1(500),X2(500),XA1(500),XA2(500),Z(500)
           DIMENSION XB1(500),XB2(500)
           DIMENSION LIN1(100,L2DIM),LIN2(100,L1DIM)
           DIMENSION LINMAT(10000,L1DIM),CONVRT(10000,L1DIM)
           VECTOR VALUES L1DIM = 2,1,100
           VECTOR VALUES L2DIM = 2,1,1
           VECTOR VALUES OUTPUT = $5HX(1)=,25(4(E15.8,2H, )/),
          1 7HANSWER=,E15.8*$
START      PRINT COMMENT $1 LINEAR PREDICTION - STUART A. ROONEY$
           OFF = 0
           I = 1
           M = 100
           THROUGH ENDA, FOR A = 0,1,A.G.500
           X(A) = 0.
           X1(A) = 0.
           X2(A) = 0.
           XA1(A) = 0.
           XA2(A) = 0.
ENDA       Z(A) = 0.
           PRINT COMMENT $ INPUT DATA $
           READ AND PRINT DATA
           S = 0
           A0A = 0.
           A01 = (X(P) - X(P-N))/N
           THROUGH ENDB, FOR B = 1,1,B.G.N
ENDB       A0A = A0A + X(P-N+B)
           A0 = A0A/N
           PRINT COMMENT $4ALPHA ZERO $
           PRINT RESULTS A0, A01
           THROUGH ENDC, FOR C = 1,1,C.G.N
           X2(C) = X(P-N+C) -A0
ENDC       XA1(C) = X2(C)
RETURN     THROUGH ENDD, FOR D = 0,1,D.G.M
           THROUGH ENDE, FOR E = 0,1,E.G.500
ENDE       X1(E) = 0.
           THROUGH ENDF, FOR F = 1,1,F.G.N-D
ENDF       X1(F) = X2(F) * X2(F+D)
           Z(D) = 0.
           THROUGH ENDG, FOR G = 1,1,G.G.N-D
ENDG       Z(D) = Z(D) + X1(G)
ENDD       Z(D) = Z(D)/(N-D)
           WHENEVER S.E.0
           TRANSFER TO RAW
           OR WHENEVER S.E.1
           TRANSFER TO ONE
           OR WHENEVER S.E.2
           TRANSFER TO TWO
           END OF CONDITIONAL
RAW        PRINT COMMENT $4AUTOCORRELATION OF THE DATA $
           PRINT RESULTS Z(0)...Z(M)
           THROUGH ENDKK,FOR KK = 0,1,KK.G.M
```

1

```
ENDKK       XB1(KK) = Z(KK)
            S = 1
            THROUGH ENDH, FOR H = 1,1,H.G.N
            XA2(H) = X(P-N+H) - X(P-N+H-1) - A01
ENDH        X2(H) = XA2(H)
            TRANSFER TO RETURN
ONE         PRINT COMMENT $4AUTOCORRELATION OF THE FIRST DIFFERENCES $
            PRINT RESULTS Z(0)...Z(M)
            THROUGH ENDLL, FOR LL = 0,1,LL.G.M
ENDLL       XB2(LL) = Z(LL)
            S = 2
            THROUGH ENDR, FOR R = 1,1,R.G.N
ENDR        X2(R) = X(P-N+R) - 2*X(P-N+R-1) + X(P-N+R-2)
            TRANSFER TO RETURN
TWO         PRINT COMMENT $4AUTOCORRELATION OF THE SECOND DIFFERENCES $
            SWITCH = 0
SECOND      THROUGH ENDT, FOR T = 0,1,T.G.500
            X(T) = 0.
            X1(T) = 0.
            X2(T) = 0.
ENDT        Z(T) = 0.
            WHENEVER SWITCH.E.0
            A0Q = A0
            THROUGH ENDU,FOR U = 1,1,U.G.N
ENDU        Z(U) = XA1(U)
            THROUGH ENDV, FOR V = 0,1,V.G.M
ENDV        X(V) = XB1(V)
            OR WHENEVER SWITCH/2 + SWITCH/2.NE. SWITCH
            A0Q = A01
            THROUGH ENDW, FOR W = 1,1,W.G.N
ENDW        Z(W) = XA2(W)
            THROUGH ENDAA, FOR AA = 0,1,AA.G.M
ENDAA       X(AA) = XB2(AA)
            THROUGH ENDBB, FOR BB = 0,1,BB.G.M-1
ENDBB       XA1(BB) = XA1(BB+1)
            XA1(M) = AVPRD1 + AVPRD2
            OR WHENEVER SWITCH/2 + SWITCH/2 .E. SWITCH
            A0Q = A0
            THROUGH ENDMM, FOR MM = 1,1,MM.G.N
ENDMM       Z(MM) = XA1(MM)
            THROUGH ENDNN, FOR NN = 0,1,NN.G.M
ENDNN       X(NN) = XB1(NN)
            THROUGH ENDPP, FOR PP = 0,1,PP.G.M-1
ENDPP       XA2(PP) = XA2(PP+1)
            XA2(M) = AVPRD1 + AVPRD2
            END OF CONDITIONAL
            SWITCH = SWITCH + 1
            THROUGH ENDCC, FOR CC = 1,1,CC.G.M
            LIN1(CC,1) = 1.0
ENDCC       LIN2(1,CC) = X(CC - 1)
            THROUGH ENDDEF, FOR DD = 1,1,DD.G.M
            THROUGH ENDDEF, FOR EE = 1,1,EE.G.M
            CONVRT(DD,EE) = 0.
```

```
         THROUGH ENDDEF, FOR FF = 1,1,FF.G.1
ENDDEF   CONVRT(DD,EE)=CONVRT(DD,EE)+LIN1(DD,FF)*LIN2(FF,EE)
         THROUGH ENDGH, FOR GG = 1,1,GG.G.M
         THROUGH ENDGH, FOR HH = 1,1,HH.G.M
         WHENEVER GG.LE.HH
         LINMAT(GG,HH) = CONVRT(GG,HH - GG +1)
         OTHERWISE
         LINMAT(GG,HH) = CONVRT(GG,GG - HH +1)
         END OF CONDITIONAL
ENDGH    CONTINUE
         THROUGH ENDII, FOR II = 1,1,II.G.M
ENDII    CONVRT(II,1) = X(II)
         SCALE = 1.0
         TEST1 = XSMEQ.(100,M,1,LINMAT,CONVRT,SCALE,X1)
         PRINT RESULTS TEST1
         AVPRD1 = 0.
         AVPRD2 = 0.
         A02 = 0.
         THROUGH ENDJJ, FOR JJ = 1,1,JJ.G.M
         X(JJ) = LINMAT(M+1-JJ,1) * Z(N-M+JJ)
         PRINT RESULTS LINMAT(M+1-JJ,1),Z(N-M+JJ),X(JJ)
ENDJJ    AVPRD1 = AVPRD1 + X(JJ)
         PRINT RESULTS AVPRD1
         ANSWER = A0Q + AVPRD1
         PRINT RESULTS ANSWER
         THROUGH ENDQQ, FOR QQ = 1,1,QQ.G.M
         X(QQ) = Z(N-M+QQ) - X(QQ)
ENDQQ    A02 = A02 + X(QQ)
         THROUGH ENDT2, FOR T2 = 1,1,T2.G.M
ENDT2    X(T2) = X(T2) - A02/M
         PRINT RESULTS X(1)...X(M)
         PUNCH FORMAT OUTPUT, X(1)...X(M),ANSWER
         WHENEVER SWITCH/2 + SWITCH/2.NE.SWITCH
         PRINT COMMENT $1PREDICTION OF THE NEXT FIRST DIFFERENCE $
         TRANSFER TO SECOND
         OR WHENEVER SWITCH/2.NE.I
         TRANSFER TO SECOND
         OR WHENEVER SWITCH/2.E.I.AND. OFF.E.0
         TRANSFER TO START
         OR WHENEVER SWITCH/2.E.I.AND.OFF.E.1
         EXECUTE EXIT.
         END OF CONDITIONAL
         END OF PROGRAM
       COUNT   5
       REM     PROGRAM TO DISABLE FOREGROUND COMMUNICATION
       ENTRY   WRFLX
       ENTRY   WRFLXA
WRFLXA TSX     $EXIT,4
WRFLX  EQU     WRFLXA
       END
```

158

TOTAL 158

```
R   PREDQD - QUADRATIC PREDICTION PROGRAM (FAST)
           NORMAL MODE IS INTEGER
           FLOATING POINT X,LINMAT,CONVRT,AVPRD2,ANSWER,ARRAY,VECTOR,
          1 LIN1,LIN2
           DIMENSION X(100),VECTOR(100),ARRAY(21200)
           DIMENSION LINMAT(2500,L1DIM),CONVRT(2500,L1DIM)
           DIMENSION LIN1(50,L2DIM),LIN2(50,L1DIM)
           VECTOR VALUES L1DIM = 2,1,50
           VECTOR VALUES L2DIM = 2,1,1
START      PRINT COMMENT $1 QUADRATIC PREDICTION PROGRAM$
           READ AND PRINT DATA
           COUNTR = L*(L+1)/2 + 1
           THROUGH ENDAB1, FOR A1 = 1,1,A1.G.L
           THROUGH ENDAB1, FOR B1 = 1,1,B1.G.A1
           LINMAT(A1,B1) = 0.
           THROUGH ENDC1, FOR C1 = 1,1,C1.G.M-A1
ENDC1      LINMAT(A1,B1) = LINMAT(A1,B1) + X(C1)*X(C1+A1)*X(C1+A1-B1)
           COUNTR = COUNTR - 1
ENDAB1     CONVRT(COUNTR,1) = LINMAT(A1,B1)/((M-A1)*(M-A1))
           INT = L*(L+1)/2
           THROUGH ENDAB2, FOR A2 = 1,1,A2.G.L
           THROUGH ENDAB2, FOR B2 = 1,1,B2.G.A2
           THROUGH ENDAB2, FOR AB = 1,1,AB.G.B2
           CC = 1 + 2500*(AB-1) + 50*(B2-1) + (A2-1)
           ARRAY(CC) = 0.
           WHENEVER A2.GE.B2 .AND. A2.GE.AB
           AC = M - A2 + 1
           OR WHENEVER B2.G.A2 .AND. B2.G.AB
           AC = M - B2 + 1
           OR WHENEVER AB.G.A2 .AND. AB.G.B2
           AC = M - AB + 1
           END OF CONDITIONAL
           THROUGH ENDC2, FOR C2 = 1,1,C2.G.AC
ENDC2      ARRAY(CC)=ARRAY(CC)+X(C2)*X(C2+AB-1)*X(C2+A2-1)*X(C2+B2-1)
ENDAB2     ARRAY(CC) = ARRAY(CC)/(AC*AC*AC)
           SS = 1
           SSS = -1
           THROUGH ENDFED, FOR D2 = 1,1,D2.G.INT
           WHENEVER SSS.E.L-SS
           SSS = -1
           SS = SS + 1
           END OF CONDITIONAL
           SSS = SSS + 1
           SSSS = 0
           THROUGH ENDFED, FOR E2 = 1,1,E2.G.L
           THROUGH ENDFED, FOR F2 = 1,1,F2.G.L
           WHENEVER E2.E.F2
           SSSS = SSSS + 1
           LINMAT(D2,SSSS)=ARRAY(WHICHR.(SSS+1,.ABS.(E2-SSS-SS)+1,1,
          1 RET))
           OR WHENEVER E2.L.F2
           SSSS = SSSS + 1
           LINMAT(D2,SSSS)=ARRAY(WHICHR.(SSS+1,.ABS.(E2-SSS-SS)+1,F2-E2+
          11,RET))+ARRAY(WHICHR.(SSS+1,.ABS.(F2-SSS-SS)+1,F2-E2+1,RET))
           END OF CONDITIONAL
```

1

```
ENDFED      CONTINUE
            SCALE = 1.0
            TEST2 = XSMEQ.(100,INT,1,LINMAT,CONVRT,SCALE,VECTOR)
            PRINT RESULTS TEST2
            THROUGH ENDQ1, FOR Q1 = 1,1,Q1.G.L
            LIN1(Q1,1) = X(M-L+Q1)
ENDQ1       LIN2(1,Q1) = X(M-L+Q1)
            THROUGH ENDRTU,FOR RR = 1,1,RR.G.L
            THROUGH ENDRTU, FOR TT = 1,1,TT.G.L
            CONVRT(RR,TT) = 0.
            THROUGH ENDRTU, FOR UU = 1,1,UU.G.1
ENDRTU      CONVRT(RR,TT) = CONVRT(RR,TT)+LIN1(RR,UU)*LIN2(UU,TT)
            COUNTR = 0
            THROUGH ENDSTU, FOR STU = 1,1,STU.G.L
            THROUGH ENDSTU, FOR ROO = 1,1,ROO.G.L
            WHENEVER STU.L.ROO
            COUNTR = COUNTR +1
            VECTOR(COUNTR) = 2*CONVRT(STU,ROO)
            OR WHENEVER STU.E.ROO
            COUNTR = COUNTR + 1
            VECTOR(COUNTR) = CONVRT(STU,ROO)
            END OF CONDITIONAL
ENDSTU      CONTINUE
            THROUGH ENDEND, FOR END = 1,1,END.G.INT
            PRINT RESULTS VECTOR(END),LINMAT(END,1)
ENDEND      AVPRD2 = AVPRD2 + VECTOR(END)*LINMAT(END,1)
            PRINT RESULTS AVPRD2
            ANSWER = ANSWER + AVPRD2
            PRINT RESULTS ANSWER
RET         TRANSFER TO START
            END OF PROGRAM
*           MAD
            EXTERNAL FUNCTION (X,Y,Z)
            NORMAL MODE IS INTEGER
            ENTRY TO WHICHR.
            A = X
            B = Y
            C = Z
            WHENEVER A.LE.B .AND. A.LE.C
            D = A
            WHENEVER B.LE.C
            E = B
            F = C
            OTHERWISE
            E = C
            F = B
            END OF CONDITIONAL
            OR WHENEVER B.LE.A .AND. B.LE.C
            D = B
            WHENEVER A.LE.C
            E = A
            F = C
            OTHERWISE
```

2

```
          E = C
          F = A
          END OF CONDITIONAL
          OTHERWISE
          D = C
          WHENEVER A.LE.B
          E = A
          F = B
          OTHERWISE
          E = B
          F = A
          END OF CONDITIONAL
          END OF CONDITIONAL
          K = 1 + 2500*(D-1) + 50*(E-1) + (F-1)
          WHENEVER K.G.0 .AND. K.L.21200
          FUNCTION RETURN
          OTHERWISE
          ERROR RETURN
          END OF CONDITIONAL
          END OF FUNCTION
       COUNT   5
       REM     PROGRAM TO DISABLE FOREGROUND COMMUNICATION
       ENTRY   WRFLX
       ENTRY   WRFLXA
WRFLXA TSX     $EXIT,4
WRFLX  EQU     WRFLXA
       END
                                                                 135

                                                       TOTAL     135
```

```
         R   PREDQD - QUADRATIC PREDICTION PROGRAM (SLOW)
          NORMAL MODE IS INTEGER
          FLOATING POINT X,LINMAT,CONVRT,AVPRD2,ANSWER,VECTOR,LIN1,LIN2
          DIMENSION X(100),VECTOR(100)
          DIMENSION LINMAT(2500,L1DIM),CONVRT(2500,L1DIM)
          DIMENSION LIN1(50,L2DIM),LIN2(50,L1DIM)
          VECTOR VALUES L1DIM = 2,1,50
          VECTOR VALUES L2DIM = 2,1,1
          PROGRAM COMMON X, M
START     PRINT COMMENT $1 QUADRATIC PREDICTION PROGRAM$
          READ AND PRINT DATA
          COUNTR = L*(L+1)/2 + 1
          THROUGH ENDAB1, FOR A1 = 1,1,A1.G.L
          THROUGH ENDAB1, FOR B1 = 1,1,B1.G.A1
          LINMAT(A1,B1) = 0.
          THROUGH ENDC1, FOR C1 = 1,1,C1.G.M-A1
ENDC1     LINMAT(A1,B1) = LINMAT(A1,B1) + X(C1)*X(C1+A1)*X(C1+A1-B1)
          COUNTR = COUNTR - 1
ENDAB1    CONVRT(COUNTR,1) = LINMAT(A1,B1)/((M-A1)*(M-A1))
          INT = L*(L+1)/2
          SS = 1
          SSS = -1
          THROUGH ENDFED, FOR D2 = 1,1,D2.G.INT
          WHENEVER SSS.E.L-SS
          SSS = -1
          SS = SS + 1
          END OF CONDITIONAL
          SSS = SSS + 1
          SSSS = 0
          THROUGH ENDFED, FOR E2 = 1,1,E2.G.L
          THROUGH ENDFED, FOR F2 = 1,1,F2.G.L
          WHENEVER E2.E.F2
          SSSS = SSSS + 1
          LINMAT(D2,SSSS)=ARRAY.(SSS+1,.ABS.(E2-SSS-SS)+1,1)
          OR WHENEVER E2.L.F2
          SSSS = SSSS + 1
          LINMAT(D2,SSSS)=ARRAY.(SSS+1,.ABS.(E2-SSS-SS)+1,F2-E2+1)
         1 + ARRAY.(SSS+1,.ABS.(F2-SSS-SS)+1,F2-E2+1)
          END OF CONDITIONAL
ENDFED    CONTINUE
          SCALE = 1.0
          TEST2 = XSMEQ.( 50,INT,1,LINMAT,CONVRT,SCALE,VECTOR)
          PRINT RESULTS TEST2
          THROUGH ENDQ1, FOR Q1 = 1,1,Q1.G.L
          LIN1(Q1,1) = X(M-L+Q1)
ENDQ1     LIN2(1,Q1) = X(M-L+Q1)
          THROUGH ENDRTU,FOR RR = 1,1,RR.G.L
          THROUGH ENDRTU, FOR TT = 1,1,TT.G.L
          CONVRT(RR,TT) = 0.
          THROUGH ENDRTU, FOR UU = 1,1,UU.G.1
ENDRTU    CONVRT(RR,TT) = CONVRT(RR,TT)+LIN1(RR,UU)*LIN2(UU,TT)
          COUNTR = 0
          THROUGH ENDSTU, FOR STU = 1,1,STU.G.L
          THROUGH ENDSTU, FOR ROO = 1,1,ROO.G.L
          WHENEVER STU.L.ROO
```

1

```
           COUNTR = COUNTR +1
           VECTOR(COUNTR) = 2*CONVRT(STU,ROO)
           OR WHENEVER STU.E.ROO
           COUNTR = COUNTR + 1
           VECTOR(COUNTR) = CONVRT(STU,ROO)
           END OF CONDITIONAL
ENDSTU     CONTINUE
           THROUGH ENDEND, FOR END = 1,1,END.G.INT
           PRINT RESULTS VECTOR(END),LINMAT(END,1)
ENDEND     AVPRD2 = AVPRD2 + VECTOR(END)*LINMAT(END,1)
           PRINT RESULTS AVPRD2
           ANSWER = ANSWER + AVPRD2
           PRINT RESULTS ANSWER
           TRANSFER TO START
           END OF PROGRAM
*          MAD
           EXTERNAL FUNCTION (A,B,C)
           NORMAL MODE IS INTEGER
           FLOATING POINT X, Y
           PROGRAM COMMON X, M
           DIMENSION X(100)
           ENTRY TO ARRAY.
           D = A
           E = B
           F = C
           WHENEVER D - E .L. 0
           H = E
           OTHERWISE
           H = D
           END OF CONDITIONAL
           WHENEVER H - F .L. 0, H = F
           J = M - H + 1
           THROUGH END, FOR K = 1,1,K.G.J
END        Y = Y + X(K)*X(K+D-1)*X(K+E-1)*X(K+F-1)
           Y = Y/(J*J*J)
           FUNCTION RETURN Y
           END OF FUNCTION
       COUNT   5
       REM     PROGRAM TO DISABLE FOREGROUND COMMUNICATION
       ENTRY   WRFLX
       ENTRY   WRFLXA
WRFLXA TSX     $EXIT,4
WRFLX  EQU     WRFLXA
       END
                                                                     99

                                                            TOTAL    99
```

```
R     EXTRAP - NONLINEAR MULTIPLE CORRELATION PROGRAM
        NORMAL MODE IS INTEGER
        DIMENSION M(11),ARRAY(6600,ARYDIM),X(150),Z(50)
        DIMENSION MATRIX(2500,MATDIM), COLMAT(2500,MATDIM)
        FLOATING POINT ARRAY,MATRIX,COLMAT,SCALE,X,Z
        VECTOR VALUES ARYDIM = 3,1,4,11
        VECTOR VALUES MATDIM = 2,1,50
        VECTOR VALUES INPUT = $A2,5X,I3,5X,I2,3X,11(I1,2X)*$
        VECTOR VALUES OUTPUT = $7H-ALPHA(,I2,1H,,I1,4H) = ,E13.6*$
        PROGRAM COMMON ARRAY
        DATANO = 0
        M(0) = 1
START   GO = $NO$
        READ FORMAT INPUT, GO, P, N, M(1)...M(11)
        WHENEVER GO.NE.$GO$
        PRINT COMMENT $1 THAT IS ALL.$
        EXECUTE EXIT.
        END OF CONDITIONAL
        DATANO = DATANO + 1
        PRINT COMMENT $1 NONLINEAR MULTIPLE CORRELATION PROGRAM$
        PRINT RESULTS DATANO
        THROUGH ENDA, FOR A = 1,1,A.G.P
ENDA    READ FORMAT $12F6*$,X(A),ARRAY(A,1,1)... ARRAY(A,1,11)
        THROUGH ENDB, FOR B = 1,1,B.G.P
        THROUGH ENDB, FOR D = 1,1,D.G.N
        THROUGH ENDB, FOR C = 2,1,C.G.M(D)
ENDB    ARRAY(B,C,D) = ARRAY(B,C-1,D)*ARRAY(B,1,D)
        MAT = 0
        THROUGH ENDE, FOR E = 0,1,E.G.N
ENDE    MAT = MAT + M(E)
        CNT5 = 0
        CNT6 = 0
        THROUGH ENDI, FOR I = 1,1,I.G.MAT
        CNT6 = CNT6 + 1
        WHENEVER CNT6.G.M(CNT5)
        CNT6 = 1
        CNT5 = CNT5 + 1
        END OF CONDITIONAL
        Z(I) = 0.
        COLMAT(I,1) = 0.
        WHENEVER CNT5.E.0
        Z(I) = P
        THROUGH ENDL, FOR L = 1,1,L.G.P
ENDL    COLMAT(1,1) = COLMAT(1,1) + X(L)
        TRANSFER TO ENDI
        END OF CONDITIONAL
        THROUGH ENDJ, FOR J = 1,1,J.G.P
        Z(I) = Z(I) + ARRAY(J,CNT6,CNT5)
ENDJ    COLMAT(I,1) = COLMAT(I,1) + X(J)*ARRAY(J,CNT6,CNT5)
ENDI    CONTINUE
        CNT1 = 0
        CNT2 = 0
        THROUGH ENDF, FOR F = 1,1,F.G.MAT
        CNT2 = CNT2 + 1
        WHENEVER CNT2.G.M(CNT1)
```

1

```
          CNT2 = 1
          CNT1 = CNT1 + 1
          END OF CONDITIONAL
          CNT3 = 0
          CNT4 = 0
          THROUGH ENDF, FOR G = 1,1,G.G.MAT
          CNT4 = CNT4 + 1
          WHENEVER CNT4.G.M(CNT3)
          CNT4 = 1
          CNT3 = CNT3 + 1
          END OF CONDITIONAL
          WHENEVER F.E.1
          MATRIX(F,G) = Z(G)
          TRANSFER TO ENDF
          OR WHENEVER G.E.1
          MATRIX(F,G) = Z(F)
          TRANSFER TO ENDF
          END OF CONDITIONAL
          THROUGH ENDH, FOR H = 1,1,H.G.P
ENDH      MATRIX(F,G) = MATRIX(F,G) +
         1 ARRAY(H,CNT4,CNT3)*ARRAY(H,CNT2,CNT1)
ENDF      CONTINUE
          SCALE = 1.0
          T = XSMEQ.(50,MAT,1,MATRIX,COLMAT,SCALE,Z)
          WHENEVER T.NE.1
          PRINT COMMENT $2 YOU LOSE.$
          WHENEVER T.E.2
          PRINT COMMENT $ MULTIPLICATION OVERFLOW IN INVERSION.$
          OR WHENEVER T.E.3
          PRINT COMMENT $ THE MATRIX IS SINGULAR.$
          END OF CONDITIONAL
          TRANSFER TO START
          END OF CONDITIONAL
          CNT7 = 0
          CNT8 = 0
          THROUGH ENDK, FOR K = 1,1,K.G.MAT
          CNT8 = CNT8 + 1
          WHENEVER CNT8.G.M(CNT7)
          CNT8 = 1
          CNT7 = CNT7 + 1
          END OF CONDITIONAL
ENDK      PRINT FORMAT OUTPUT, CNT7, CNT8, MATRIX(K,1)
          TRANSFER TO START
          END OF PROGRAM
       COUNT   5
       REM     PROGRAM TO DISABLE FOREGROUND COMMUNICATION
       ENTRY   WRFLX
       ENTRY   WRFLXA
WRFLXA TSX     $EXIT,4
WRFLX  EQU     WRFLXA
       END
```

106

TOTAL 106

```
        R   SIGTST - SIGNIFICANCE TESTING PROGRAM
            INTEGER A,B,N,DATANO,TEST,XSMEQ.,GO
            DIMENSION X(1000),Y(1000),Z(1000)
            DIMENSION MATRIX(9,DIM),COLMAT(9,DIM)
            VECTOR VALUES DIM=2,1,3
            VECTOR VALUES INPUT = $3(F5,5X)*$
            DATANO=0
START       DATANO=DATANO+1
            GO = 1
            PRINT COMMENT $1MULTIPLE CORRELATION PROGRAM$
            PRINT RESULTS DATANO
            READ FORMAT $I1,I4*$,GO,N
            WHENEVER GO.NE.0, EXECUTE EXIT.
            THROUGH ENDA, FOR A=1,1,A.G.N
            READ FORMAT INPUT, X(A), Y(A), Z(A)
ENDA        PRINT RESULTS X(A), Y(A), Z(A)
            SUM1=0.
            SUM2=0.
            SUM3=0.
            SUM11=0.
            SUM12=0.
            SUM13=0.
            SUM22=0.
            SUM23=0.
            SUM33=0.
            A123 = 0.
            B123 = 0.
            B132 = 0.
            THROUGH ENDB, FOR B=1,1,B.G.N
            SUM1=SUM1+X(B)
            SUM2=SUM2+Y(B)
            SUM3=SUM3+Z(B)
            SUM11=SUM11+X(B)*X(B)
            SUM12=SUM12+X(B)*Y(B)
            SUM13=SUM13+X(B)*Z(B)
            SUM22=SUM22+Y(B)*Y(B)
            SUM23=SUM23+Y(B)*Z(B)
ENDB        SUM33=SUM33+Z(B)*Z(B)
            PRINT RESULTS SUM1,SUM2,SUM3
            PRINT RESULTS SUM11,SUM12,SUM13
            PRINT RESULTS SUM22,SUM23,SUM33
            MATRIX(1,1)=N
            MATRIX(1,2)=SUM2
            MATRIX(2,1)=SUM2
            MATRIX(2,2)=SUM22
            COLMAT(1,1)=SUM1
            COLMAT(2,1)=SUM12
            SCALE=1.0
            TEST=XSMEQ.(3,2,1,MATRIX, COLMAT, SCALE, X)
            PRINT RESULTS TEST
            WHENEVER TEST.NE.1
            PRINT COMMENT $1 YOU LOSE$
            TRANSFER TO START
            END OF CONDITIONAL
            A12=MATRIX(1,1)
```

1

```
              B12=MATRIX(2,1)
              MATRIX(1,1)=N
              MATRIX(1,2)=SUM3
              MATRIX(2,1)=SUM3
              MATRIX(2,2)=SUM33
              COLMAT(1,1)=SUM1
              COLMAT(2,1)=SUM13
              SCALE=1.0
              TEST=XSMEQ.(3,2,1, MATRIX, COLMAT, SCALE, X)
              PRINT RESULTS TEST
              WHENEVER TEST.NE.1
              PRINT COMMENT $1 YOU LOSE$
              TRANSFER TO START
              END OF CONDITIONAL
              A13=MATRIX(1,1)
              B13=MATRIX(2,1)
              MATRIX(1,1)=N
              MATRIX(1,2)=SUM2
              MATRIX(1,3)=SUM3
              MATRIX(2,1)=SUM2
              MATRIX(2,2)=SUM22
              MATRIX(2,3)=SUM23
              MATRIX(3,1)=SUM3
              MATRIX(3,2)=SUM23
              MATRIX(3,3)=SUM33
              COLMAT(1,1)=SUM1
              COLMAT(2,1)=SUM12
              COLMAT(3,1)=SUM13
              SCALE=1.0
              TEST=XSMEQ.(3,3,1,MATRIX,COLMAT,SCALE,X)
              PRINT RESULTS TEST
              WHENEVER TEST.NE.1
              PRINT COMMENT $1 YOU LOSE$
              TRANSFER TO GOOF
              END OF CONDITIONAL
              A123=MATRIX(1,1)
              B123=MATRIX(2,1)
              B132=MATRIX(3,1)
GOOF               R21 =  (A12*SUM1+B12*SUM12-SUM1*SUM1/N)/
             1 (SUM11-SUM1*SUM1/N)
                   R31 =  (A13*SUM1+B13*SUM13-SUM1*SUM1/N)/
             1 (SUM11-SUM1*SUM1/N)
                  R321 = (A123*SUM1+B123*SUM12+B132*SUM13-SUM1*SUM1/N)/
             1 (SUM11-SUM1*SUM1/N)
              R12 = SQRT.(R21)
              R13 = SQRT.(R31)
              R123 = SQRT.(R321)
              S12=SQRT.((SUM11-A12*SUM1-B12*SUM12)/N)
              S13=SQRT.((SUM11-A13*SUM1-B13*SUM13)/N)
              S123=SQRT.((SUM11-A123*SUM1-B123*SUM12-B132*SUM13)/N)
              F12 = R21/((1-R21)/(N-2))
              F13 = R31/((1-R31)/(N-2))
              F123 = (R321/2)/((1-R321)/(N-3))
```

```
            PRINT COMMENT $4ANSWER$
            PRINT RESULTS R12,R13,A12,A13,B12,B13
            PRINT RESULTS R123, A123, B123, B132
            PRINT RESULTS S12,S13,S123
            PRINT RESULTS F12,F13,F123
            TRANSFER TO START
            END OF PROGRAM
        *EOF
INPUT,M4008,3519,PREDCT,MAD,1
            NORMAL MODE IS INTEGER
            FLOATING POINT X,X1,X2,Y,Z,LINMAT,CONVRT,LIN1,LIN2,AVPRD1
            FLOATING POINT ERROR,SUM,W
            DIMENSION X( 700), X1(600), X2(600),Y( 60),Z(600)
            DIMENSION LIN1(100,L2DIM), LIN2(100,L1DIM)
            DIMENSION LINMAT(10000,L1DIM),CONVRT(10000,L1DIM)
            VECTOR VALUES L1DIM = 2,1,100
            VECTOR VALUES L2DIM = 2,1,1
            I = 1
            M = 100
            SUM = 0.
            ERROR = 0.
            READ DATA
            THROUGH ENDC, FOR C = 1,1,C.G.P-1
ENDC        X(I) = X(I+1)/X(I) - 1.0
            THROUGH END, FOR B = 0,1,B.G.P-N-1
            THROUGH ENDA, FOR A = 0,1,A.G.600
            X1(A) = 0.
            X2(A) = 0.
ENDA        Z(A) = 0.
            THROUGH ENDH, FOR H = 1,1,H.G.N
ENDH        X2(H) = X(H+B)
            THROUGH ENDD, FOR D = 0,1,D.G.M+I-1
            THROUGH ENDE, FOR E = 0,1,E.G.600
ENDE        X1(E) = 0.
            THROUGH ENDF, FOR F = 1,1,F.G.N-D
ENDF        X1(F) = X2(F) * X2(F+D)
            Z(D) = 0.
            THROUGH ENDG, FOR G = 1,1,G.G.N-D
ENDG        Z(D) = Z(D) + X1(G)
ENDD        Z(D) = Z(D)/(N-D)
            PRINT COMMENT $4 AUTOCORRELATES$
            PRINT RESULTS Z(0)...Z(M+I-1)
            THROUGH ENDT, FOR T = 0,1,T.G.600
ENDT        X1(T) = 0.
            THROUGH ENDCC, FOR CC = 1,1,CC.G.M
            LIN1(CC,1) = 1.0
ENDCC       LIN2(1,CC) = Z(CC - 1)
            THROUGH ENDDEF, FOR DD = 1,1,DD.G.M
            THROUGH ENDDEF, FOR EE = 1,1,EE.G.M
            CONVRT(DD,EE) = 0.
            THROUGH ENDDEF, FOR FF = 1,1,FF.G.1
ENDDEF      CONVRT(DD,EE)=CONVRT(DD,EE)+LIN1(DD,FF)*LIN2(FF,EE)
            THROUGH ENDGH, FOR GG = 1,1,GG.G.M
```

3

```
          THROUGH ENDGH, FOR HH = 1,1,HH.G.M
          WHENEVER GG.LE.HH
          LINMAT(GG,HH) = CONVRT(GG,HH - GG +1)
          OTHERWISE
          LINMAT(GG,HH) = CONVRT(GG,GG - HH +1)
          END OF CONDITIONAL
ENDGH     CONTINUE
          THROUGH ENDII, FOR II = 1,1,II.G.M
ENDII     CONVRT(II,1) = Z(II+I-1)
          SCALE = 1.0
          T = XSMEQ.(100,M,1,LINMAT,CONVRT,SCALE,X1)
          WHENEVER T.NE.1
          PRINT COMMENT $2 YOU LOSE.$
          WHENEVER T.E.2
          PRINT COMMENT $ MULTIPLICATION OVERFLOW IN INVERSION.$
          OR WHENEVER T.E.3
          PRINT COMMENT $ THE MATRIX IS SINGULAR.$
          END OF CONDITIONAL
          EXECUTE EXIT.
          END OF CONDITIONAL
          AVPRD1 = 0.
          THROUGH ENDJJ, FOR JJ = 1,1,JJ.G.M
          X(JJ) = LINMAT(M+1-JJ,1) * Z(N-M+JJ)
          PRINT RESULTS LINMAT(M+1-JJ,1),Z(N-M+JJ),X(JJ)
ENDJJ     AVPRD1 = AVPRD1 + X(JJ)
          Y(B) = AVPRD1*X(N+B) + X(N+B)
          W = X(N+B+1)*X(N+B) + X(N+B)
          SUM = SUM + W*W
END       ERROR = ERROR + (X(N+B+1)-Y(B))*(X(N+B+1)-Y(B))
          ERROR = ERROR/SUM
          PRINT RESULTS ERROR,Y(0)...Y(P-N-1)
          EXECUTE EXIT.
          END OF PROGRAM
                                                              194

                                                   TOTAL      194
```

```
            R   PREDCT - PREDICTION PROGRAM
             NORMAL MODE IS INTEGER
             FLOATING POINT X,X1,X2,Y,Z,LINMAT,CONVRT,LIN1,LIN2,AVPRD1
             FLOATING POINT ERROR,SUM,W
             DIMENSION X( 700), X1(600), X2(600),Y( 60),Z(600)
             DIMENSION LIN1(100,L2DIM), LIN2(100,L1DIM)
             DIMENSION LINMAT(10000,L1DIM),CONVRT(10000,L1DIM)
             VECTOR VALUES L1DIM = 2,1,100
             VECTOR VALUES L2DIM = 2,1,1
             I = 1
             M = 100
             SUM = 0.
             ERROR = 0.
             READ DATA
             THROUGH ENDC, FOR C = 1,1,C.G.P-1
ENDC         X(I) = X(I+1)/X(I) - 1.0
             THROUGH END, FOR B = 0,1,B.G.P-N-1
             THROUGH ENDA, FOR A = 0,1,A.G.600
             X1(A) = 0.
             X2(A) = 0.
ENDA         Z(A) = 0.
             THROUGH ENDH, FOR H = 1,1,H.G.N
ENDH         X2(H) = X(H+B)
             THROUGH ENDD, FOR D = 0,1,D.G.M+I-1
             THROUGH ENDE, FOR E = 0,1,E.G.600
ENDE         X1(E) = 0.
             THROUGH ENDF, FOR F = 1,1,F.G.N-D
ENDF         X1(F) = X2(F) * X2(F+D)
             Z(D) = 0.
             THROUGH ENDG, FOR G = 1,1,G.G.N-D
ENDG         Z(D) = Z(D) + X1(G)
ENDD         Z(D) = Z(D)/(N-D)
             PRINT COMMENT $4 AUTOCORRELATES$
             PRINT RESULTS Z(0)...Z(M+I-1)
             THROUGH ENDT, FOR T = 0,1,T.G.600
ENDT         X1(T) = 0.
             THROUGH ENDCC, FOR CC = 1,1,CC.G.M
             LIN1(CC,1) = 1.0
ENDCC        LIN2(1,CC) = Z(CC - 1)
             THROUGH ENDDEF, FOR DD = 1,1,DD.G.M
             THROUGH ENDDEF, FOR EE = 1,1,EE.G.M
             CONVRT(DD,EE) = 0.
             THROUGH ENDDEF, FOR FF = 1,1,FF.G.1
ENDDEF       CONVRT(DD,EE)=CONVRT(DD,EE)+LIN1(DD,FF)*LIN2(FF,EE)
             THROUGH ENDGH, FOR GG = 1,1,GG.G.M
             THROUGH ENDGH, FOR HH = 1,1,HH.G.M
             WHENEVER GG.LE.HH
             LINMAT(GG,HH) = CONVRT(GG,HH - GG +1)
             OTHERWISE
             LINMAT(GG,HH) = CONVRT(GG,GG - HH +1)
             END OF CONDITIONAL
ENDGH        CONTINUE
             THROUGH ENDII, FOR II = 1,1,II.G.M
ENDII        CONVRT(II,1) = Z(II+I-1)
             SCALE = 1.0
```

1

```
          T = XSMEQ.(100,M,1,LINMAT,CONVRT,SCALE,X1)
          WHENEVER T.NE.1
          PRINT COMMENT $2 YOU LOSE.$
          WHENEVER T.E.2
          PRINT COMMENT $ MULTIPLICATION OVERFLOW IN INVERSION.$
          OR WHENEVER T.E.3
          PRINT COMMENT $ THE MATRIX IS SINGULAR.$
          END OF CONDITIONAL
          EXECUTE EXIT.
          END OF CONDITIONAL
          AVPRD1 = 0.
          THROUGH ENDJJ, FOR JJ = 1,1,JJ.G.M
          X(JJ) = LINMAT(M+1-JJ,1) * Z(N-M+JJ)
          PRINT RESULTS LINMAT(M+1-JJ,1),Z(N-M+JJ),X(JJ)
ENDJJ     AVPRD1 = AVPRD1 + X(JJ)
          Y(B) = AVPRD1*X(N+B) + X(N+B)
          W = X(N+B+1)*X(N+B) + X(N+B)
          SUM = SUM + W*W
END       ERROR = ERROR + (X(N+B+1)-Y(B))*(X(N+B+1)-Y(B))
          ERROR = ERROR/SUM
          PRINT RESULTS ERROR,Y(0)...Y(P-N-1)
          EXECUTE EXIT.
          END OF PROGRAM
       COUNT   5
       REM     PROGRAM TO DISABLE FOREGROUND COMMUNICATION
       ENTRY   WRFLX
       ENTRY   WRFLXA
WRFLXA TSX     $EXIT,4
WRFLX  EQU     WRFLXA
       END
                                                                  85

                                                      TOTAL       85
```

REFERENCES

1. Paul H. Cootner, The Random Character of Stock Prices, Cambridge, Mass., MIT Press, 1964.

2. Norbert Wiener, The Extrapolation, Interpolation, and Smoothing of Stationary Time Series, New York, N.Y.: John Wiley and Sons, Inc., 1949.

3. Murray Rosenblatt, Random Processes, New York, N.Y.: Oxford University Press, 1962, pp. 149-153.

4. Norbert Wiener, Nonlinear Problems in Random Theory, Cambridge, Mass.: MIT Press, 1958.

5. Claud Shannon, "A Mathematical Theory of Communication," Bell System Technical Journal, Vol. 27, pp. 379-423, 623-656, 1948.

6. Vito Volterra, Theory of Functionals and Integral and Integro-differential Equations, Dover Edition, 1959, pp. 14, 21.

7. Robert Parente, The Analysis of Nonlinear Systems, Proposal for MIT Doctor of Science Thesis, 1962, pp. 13-24.

8. Fernando Corbató, et al., The Compatible Time-Sharing System, Cambridge, Mass.: MIT Press, 1963.

9. Francis Russo, Jr., Predicting the Dow Jones Industrial Index, MIT Industrial Management Thesis, 1961.

10. Ponzin and Hellwig, "A Proposal for Definition of Macro-commands," MIT Computation Center Memo No. 238, 1964.

11. Holbrook Working, "Note on the Correlation of First Differences of Averages in a Random Chain," Econometrica, Vol. 28 (No. 4), October 1960, pp. 916-918.

12. C.A. Stutt and Y.W. Lee, "Statistical Prediction of Noise," MIT Research Laboratory for Electronics Report No. 102, 1960.