



Scientific Excellence • Resource Protection & Conservation • Benefits for Canadians
Excellence scientifique • Protection et conservation des ressources • Bénéfices aux Canadiens

Computer Programs for Investigating the Effects of Environmental Events on a Time Series of Recruitment

J.M. Hoenig, M.H. Prager,
and N.B. Payton

Science Branch
Department of Fisheries and Oceans
P.O. Box 5667
St. John's, Newfoundland A1C 5X1

November 1989

**Canadian Technical Report of
Fisheries and Aquatic Sciences
No. 1713**



Fisheries
and Oceans

Pêches
et Océans

Canada

Canadian Technical Report of Fisheries and Aquatic Sciences

Technical reports contain scientific and technical information that contributes to existing knowledge but which is not normally appropriate for primary literature. Technical reports are directed primarily toward a worldwide audience and have an international distribution. No restriction is placed on subject matter and the series reflects the broad interests and policies of the Department of Fisheries and Oceans, namely, fisheries and aquatic sciences.

Technical reports may be cited as full publications. The correct citation appears above the abstract of each report. Each report is abstracted in *Aquatic Sciences and Fisheries Abstracts* and indexed in the Department's annual index to scientific and technical publications.

Numbers 1-456 in this series were issued as Technical Reports of the Fisheries Research Board of Canada. Numbers 457-714 were issued as Department of the Environment, Fisheries and Marine Service, Research and Development Directorate Technical Reports. Numbers 715-924 were issued as Department of Fisheries and the Environment, Fisheries and Marine Service Technical Reports. The current series name was changed with report number 925.

Technical reports are produced regionally but are numbered nationally. Requests for individual reports will be filled by the issuing establishment listed on the front cover and title page. Out-of-stock reports will be supplied for a fee by commercial agents.

Rapport technique canadien des sciences halieutiques et aquatiques

Les rapports techniques contiennent des renseignements scientifiques et techniques qui constituent une contribution aux connaissances actuelles, mais qui ne sont pas normalement appropriés pour la publication dans un journal scientifique. Les rapports techniques sont destinés essentiellement à un public international et ils sont distribués à cet échelon. Il n'y a aucune restriction quant au sujet; de fait, la série reflète la vaste gamme des intérêts et des politiques du ministère des Pêches et des Océans, c'est-à-dire les sciences halieutiques et aquatiques.

Les rapports techniques peuvent être cités comme des publications complètes. Le titre exact paraît au-dessus du résumé de chaque rapport. Les rapports techniques sont résumés dans la revue *Résumés des sciences aquatiques et halieutiques*, et ils sont classés dans l'index annuel des publications scientifiques et techniques du Ministère.

Les numéros 1 à 456 de cette série ont été publiés à titre de rapports techniques de l'Office des recherches sur les pêcheries du Canada. Les numéros 457 à 714 sont parus à titre de rapports techniques de la Direction générale de la recherche et du développement, Service des pêches et de la mer, ministère de l'Environnement. Les numéros 715 à 924 ont été publiés à titre de rapports techniques du Service des pêches et de la mer, ministère des Pêches et de l'Environnement. Le nom actuel de la série a été établi lors de la parution du numéro 925.

Les rapports techniques sont produits à l'échelon régional, mais numérotés à l'échelon national. Les demandes de rapports seront satisfaites par l'établissement auteur dont le nom figure sur la couverture et la page du titre. Les rapports épuisés seront fournis contre rétribution par des agents commerciaux.

Canadian Technical Report of
Fisheries and Aquatic Sciences 1713

November 1989

COMPUTER PROGRAMS FOR INVESTIGATING THE EFFECTS OF ENVIRONMENTAL
EVENTS ON A TIME SERIES OF RECRUITMENT

by

J. M. Hoenig, M. H. Prager¹, and N. B. Payton

Science Branch
Department of Fisheries and Oceans
P.O. Box 5667
St. John's, Newfoundland A1C 5X1

¹Department of Oceanography, Old Dominion University, Norfolk, VA 23529, U.S.A.

©Minister of Supply and Services Canada 1989

Cat. No. Fs 97-6/1713E

ISSN 0706-6457

Correct citation for this publication:

Hoenig, J. M., M. H. Prager, and N. B. Payton. 1989. Computer programs for investigating the effects of environmental events on a time series of recruitment. Can. Tech. Rep. Fish. Aquat. Sci. 1713: v + 43 p.

CONTENTS

Abstract/Résumé	iv
Preface	v
Introduction	1
Program EPOCH	2
Description	2
Program preparation	2
Data input	3
Program output	3
Program EPOWER	3
Description	3
Program preparation	4
Data input	4
Program output	4
Acknowledgments	5
References	5
Appendix I -- Program EPOCH sample input file	6
Appendix II -- Program EPOCH listing	7
Appendix III -- Program EPOWER sample input file	20
Appendix IV -- Program EPOWER listing	21

ABSTRACT

Hoenig, J. M., M. H. Prager, and N. B. Payton. 1989. Computer programs for investigating the effects of environmental events on a time series of recruitment. Can. Tech. Rep. Fish. Aquat. Sci. 1713: v + 43 p.

Prager and Hoenig (in press) used a randomization procedure to test the significance of discrete environmental events, such as el Niño-Southern Oscillation events, on a time series of recruitment. Their procedure implicitly accounted for autocorrelation and other problems frequently encountered when trying to use a parametric testing procedure. In a follow up paper, Prager and Hoenig (in prep.) investigated the power of several randomization tests and compared them to parametric alternatives. This report presents the two Fortran computer programs used in these studies.

RÉSUMÉ

Hoenig, J. M., M. H. Prager, and N. B. Payton. 1989. Computer programs for investigating the effects of environmental events on a time series of recruitment. Can. Tech. Rep. Fish. Aquat. Sci. 1713: v + 43 p.

Prager et Hoenig (sous presse) ont employé une méthode de randomization afin de tester le degré de signification d'événements environnementaux discrets, tels que les événements El Niño, sur une série chronologique de recrutement. Leur méthode tenait compte de façon implicite de l'autocorrélation ainsi que d'autres problèmes rencontrés fréquemment lors de l'emploi de méthodes de test paramétriques. Dans un autre article, Prager et Hoenig (en préparation) ont étudié la puissance de plusieurs tests de randomization et les ont comparés à d'autres tests paramétriques. Dans ce rapport on présente les deux programmes Fortran qui ont servi dans ces études.

PREFACE

This report provides descriptions, listings, and user's instructions for two Fortran computer programs used by Prager and Hoenig (in press; in review) to study environmental influences on recruitment. The purpose of the report is to document the computational procedures used and to ensure that the programs remain available to future investigators. We have tested these programs carefully. However, as with any large program, it is not possible to state categorically that the programs are error-free. Users are warned to use the programs at their own risk. The programs are available through the American Fisheries Society's Computer Users' Section.

INTRODUCTION

In recent years, a great deal of attention has been focused on the problem of determining to what extent environmental conditions influence recruitment. A large number of studies have involved the use of parametric statistical methods (e.g. regression, correlation, t-tests) to describe and test the relationship between environmental factors and recruitment in available time series. These approaches generally require the following assumptions: independence of observations (i.e., from year to year), normality, and homogeneity and/or homoscedasticity of variances. All of these assumptions may be suspect when dealing with time series data. In addition, the parametric methods assume random sampling (of time) which is obviously not possible. Few studies have considered the effects of failure to meet the necessary assumptions.

One alternative to the above-mentioned parametric procedures is to use classical time series methods as described, for example, by Box and Jenkins (1970). Prager and Hoenig (in press) described another approach based on randomization tests. We concentrate on the latter approach in this report.

The randomization approach of Prager and Hoenig (in press) is based on comparing the recruitment in key event years, i.e. years in which a specified type of environmental event occurred, with recruitment in background years, i.e. the years immediately surrounding the key event years. For example, key event years might be taken to be years in which an el Niño-Southern Oscillation event occurred or years in which offshore transport in May exceeded a pre-specified threshold. Background years might be defined as the two years before and the two years after a key event year. Alternatively, one might define background years as the two years before a key event year and ignore the years immediately after the key events. (This would be appropriate if one had reason to believe that the effect of the key event carried over into the years immediately after the event year.) In the parlance of Haurwitz and Brier (1981), a key event year and its associated background year together comprise an epoch. Superposing (i.e. averaging) a set of epochs provides a visual method for assessing the "average" trend over the course of epochs.

The goal, then, is to test whether recruitment in key event years tends to be different from that in background years. The usual (parametric) approach to comparing two means is to compute some sort of t-statistic and refer to a table of Student's t-distribution to determine the significance level. In the case considered here, we could certainly compute a t-statistic. However, we would be ill-advised to assume this statistic follows a Student's-t under the null hypothesis because of failures of the assumptions. However, we can determine the null distribution, and hence the significance level, by casting the problem as a randomization test (see below). In this way we implicitly account for the failures of assumptions associated with the parametric approach.

For a randomization test, we assume that the available observations comprise the entire population of interest, i.e. that we are not dealing with a sample from some larger population.

We then test whether the association between two variables could have arisen by the chance "assignment" of the values of each variable to the units in the population. For example, the units in the population could be years; the values of one variable, recruitment, could have been assigned to the units according to some arbitrary process. Then the question becomes: could the label "key event" (versus "not-key event") have been assigned to the units (years) independently of the recruitment variable? Or, is there evidence that the key events have some association with the recruitment values, e.g. tending to occur when recruitment is high?

Suppose we wish to measure the strength of the association between recruitment and key event years by computing a t-statistic (exactly as one would for a Student's t-test). We need to determine the significance of the computed statistic with respect to the null hypothesis (that recruitment and key event status [yes or no] are assigned independently to years). Under the null hypothesis, the label "key event year" has no significance with respect to recruitment. Hence, key events could occur anywhere with respect to recruitment. Consequently, the exact null distribution of the test statistic could be determined by computing the value of the test statistic for every possible allocation of key events to years. For example, if 4 key events occurred in a 30 year period, there would be 30 choose 4 = 27,405 possible allocations of key events to years. Since the number of allocations can be large, an attractive alternative, which we used, is to determine the approximate null distribution by a Monte Carlo method. In this approximation version, a given number of "trials" is generated. For each trial, one of the possible allocations (of key events to year) is randomly selected. (The recruitment time series remains as originally observed.) The resulting histogram of computed values of the test statistic approximates the null distribution. If the observed value of the test statistic occurs in the extreme tail of the null distribution, the null hypothesis is rejected.

The interested reader is referred to Edgington (1987) for further information on randomization tests. Details of our use of randomization tests for testing hypotheses about recruitment are presented in Prager and Hoenig (in press).

Program EPOCH can be used to test for an association between discrete environmental events and a time series of recruitment (or relative recruitment or productivity). Since there are various ways to describe the degree of association (i.e. various possible test statistics), the program can be used to determine the significance level for three test statistics.

Program EPOWER was written to investigate the power of the statistical test under various assumed conditions (see Prager and Hoenig in prep.). Power is defined here as the probability of rejecting the null hypothesis under specified conditions. The program can be used to answer two kinds of questions:

- 1) which test statistics are likely to be the most powerful for a given type of recruitment pattern; and

- 2) what kind of power might be expected in practice if recruitment follows a given process and key events affect recruitment in a given way.

It is difficult to determine analytically which test statistics are the most powerful and what power levels can be achieved. Program EPOWER is written in a modular style so that additional test statistics and additional patterns of recruitment can be added to the program as needed.

PROGRAM EPOCH

DESCRIPTION

This Fortran 77 program reads in a time series of recruitment and a list of key event years. It also reads in the (user-specified) definition of background years (either the k years before each key event or the k years before and the k years after each key event), and the number of Monte Carlo trials desired. Three test statistics can be computed (D , T , or W - see below). The program then estimates the significance level for the chosen test statistic by a Monte Carlo randomization test. The significance level is estimated by $(x + 1)/(v + 1)$ where v is the number of Monte Carlo trials and x is the number of trials in which the computed test statistic exceeds the value of the test statistic for the actual data.

The simplest test statistic, D , is the difference between the mean recruitment in key event years (\bar{E}) and the mean value in background years (\bar{B}):

$$D = \bar{E} - \bar{B}.$$

The second statistic, T , was modified from Haurwitz and Brier (1981). It is computed as one would compute an ordinary Student's t -statistic:

$$T = (\bar{E} - \bar{B}) / S,$$

where \bar{E} and \bar{B} are as defined for the D statistic. The pooled standard deviation S is computed as for a t -test:

$$S = \left[\frac{(SS_E + SS_B) (N_E + N_B)}{N_E N_B (N_E + N_B - 2)} \right]^{1/2},$$

where SS_E is the sum of squared deviations of the N_E values of recruitment in key event years from their mean:

$$SS_E = \sum_{i=1}^{N_E} (E_i - \bar{E})^2,$$

where E_i is the recruitment in key event year i . Similarly, SS_B is the sum of the squared

deviations of the N_B recruitment values in background years from their grand mean.

The W statistic is analogous to Student's t -statistic computed for paired data. Each key event year is compared with its own background years:

$$W = \bar{d} \sqrt{N_B} / S_W,$$

where S_W is a measure of dispersion (defined below), N_B is as defined for the T statistic, and \bar{d} is the mean of all paired differences between the recruitment value in key event year i and the recruitment value in each corresponding background year. More precisely,

$$\bar{d} = \frac{1}{N_B} \sum_{i=1}^{N_E} \sum_{j=1}^{n_i} (E_i - B_{ij}),$$

where n_i is the number of background values available on recruitment for key event i . Generally n_i is a constant (e.g. 2 years before + 2 years after = 4), but it has fewer observations when the key event occurs near a missing value of the response or near either end of the time series. In other words, missing values of recruitment are accommodated by eliminating them from computation of the test statistic, and adjusting n_i and N_B accordingly. The measure of dispersion S_W is computed as for a paired t -test:

$$S_W = \left[\frac{1}{N_B - 1} \sum_{i=1}^{N_E} \sum_{j=1}^{n_i} (E_i - B_{ij} - \bar{d})^2 \right]^{1/2}.$$

Output from the program consists of two files. One contains information for plotting estimated significance level as a function of the number of Monte Carlo trials. In this way, one can examine whether the estimated significance level (P -value) has stabilized. The other output file documents the data and test options used, and presents the test results.

Program EPOCH uses a slightly modified version of subroutine RAN3 (* 1986 by Numerical Recipes Software. Reproduced by permission, from the book Numerical Recipes: The Art of Scientific Computing, published by Cambridge University Press).

PROGRAM PREPARATION

The program is written in standard Fortran 77 and could easily be run on a microcomputer. The program operates in batch mode - all input is read from a single data file called EPOCH.DAT and output is written to two files. One output file contains information for plotting estimated significance level as a function of the number of Monte Carlo trials (to determine whether the estimated significance level has stabilized). The second output file contains a summary of the data

and parameters used in the test and the test results.

The data arrays are dimensioned to allow up to 100 observations (years of data); 50 key events, and to permit up to 10,000 Monte Carlo trials. If this arrangement is satisfactory the program can be compiled and run without modification.

DATA INPUT

A sample data input file is presented in Appendix I. Parameters and data for analysis are entered into a data file called EPOCH.DAT in free format, as follows:

line 1: number of years (rows) of recruitment data (= NYear)
 line 2: number of Monte Carlo trials
 line 3: number of key events (= NKey)
 next NKey rows: one pair of numbers in each row, each row pertaining to a key event. First number = Year ID for the key event; second number = indicator for expected response (see (1) below)
 next line: minimum spacing for key events (see (2) below)
 next line: indicator for which test statistic should be used (enter 'T', 'D', or 'W' [n single quotes])
 next line: seed for random number generator RAN3 from Numerical Recipes (enter a negative integer)
 next line: epoch definition - enter 1 if only background years preceding the key events are used; enter 2 if epoch is two-sided, i.e. background years on each side of the key events are used.
 next line: number of background years preceding each key event (and also succeeding each key event year, if two-sided epochs are used).
 next NYear lines: time series of recruitment, one year per line. Each line consists of two numbers - the year ID and the recruitment value. Years for which no recruitment data are available are left out of the file.

Notes:

(1) Enter +1 if the recruitment is hypothesized to be higher than background (i.e. a "peak") for the corresponding type of key event (under the alternative hypothesis); enter -1 if recruitment is hypothesized to be lower than background (i.e. a trough). A combination of pluses and minuses is used for reflected event analysis.

(2) Minimum spacing for key events - there may be physical or other reasons for believing that key events cannot occur in adjacent years. In this case, the randomizations (Monte Carlo trials) can be generated subject to the restriction that key events have a minimum spacing. Otherwise enter a 1.

PROGRAM OUTPUT

File EPOCH.SIG contains information on the estimated significance level as a function of the number of Monte Carlo trials. A two line header (label) is followed by two blank lines. The

remaining lines give, for every second Monte Carlo trial, the following information on a line: Monte Carlo trial number i , value of the test statistic for the trial data i , and running significance level $([x+1]/[i+1])$, where x is the number of trial data sets for which the test statistic is greater than that observed for the actual data). The estimated significance level can be plotted against the trial number to see if the estimated significance has stabilized.

File EPOCH.OUT documents the analysis with the following information:

width of epoch
 # of background years preceding the key events
 # of background years following the key events
 the random number generator seed
 # rows of input data
 the year ID for the first row of data
 the year ID for the last row of data
 range of ID values
 # of rows with missing data
 # of key events (NKey)
 the next NKey rows list the year ID of the key event years and the expected direction of the difference between key event years and background years (i.e., positive or negative)
 which test statistic was used
 value of the test statistic for the actual data
 # of Monte Carlo trials
 minimum spacing allowed between the key events in the Monte Carlo trials
 number of Monte Carlo trials with test statistic greater than that observed for the actual data
 estimated significance level

PROGRAM EPOWER

DESCRIPTION

This Fortran 77 program estimates the realized alpha level and the power of the randomization test under various possible conditions. The simulated conditions can be such that a parametric test (e.g. a t-test) is appropriate. That is, the observations in the time series can be generated independently from a normal distribution with observations in key event years being merely shifted by some amount delta. Or, the time series can be generated with a linear trend or with autocorrelation (AR1 model) or both.

In general, recruitment value in year i is generated according to the general model

$$XX_i = \text{slope} \cdot i + AR \cdot XX_{i-1} + \text{delta} \cdot Z_i + e_i$$

where XX_i is the recruitment value in year i , slope is the trend (per year) in recruitment, AR is the autoregressive parameter, delta is the amount by which key event years differ from non-key event years, Z_i is an indicator variable for key event status (=1 if year i is a key event year), and e_i is a standard normal random variable.

Six test statistics/test procedures are included in the program. Two of these are the usual parametric tests: t-test and paired t-test, with significance level determined by referring to

Student's t-distribution. The other test procedures involve determining the significance level by randomization. The program can be used to determine the penalty for using a randomization test when a parametric test is appropriate and vice versa.

The test statistics are as follows:

- 1) $D = \bar{E} - \bar{B}$ = difference of means; see detailed description in writeup for Program EPOCH. Significance is determined by randomization.
- 2) $T = (\bar{E} - \bar{B}) / S$ = parametric t-statistic; see detailed description in writeup for program EPOCH. Significance is determined by referring to a table of Student's t-distribution with $N_E + N_B - 2$ degrees of freedom where N_E and N_B are the number of observations on key event and background years, respectively.
- 3) T_R = same as (2) but significance is determined by randomization.
- 4) $W = d \sqrt{N_B} / S_W$ = paired t-statistic; see detailed description in writeup for program EPOCH. Significance is determined by randomization.
- 5) Q = paired t-statistic defined as follows:

$$Q = \frac{\bar{D} \sqrt{N_B}}{S_Q}$$

where

$$\bar{D} = \frac{\sum_{i=1}^{N_E} (E_i - \bar{B}_i)}{N_E}$$

$$S_Q = \left[\frac{\sum_{i=1}^{N_E} (E_i - \bar{B}_i - \bar{D})^2}{N_E - 1} \right]^{1/2}$$

and N_E is the number of key event years, and \bar{B}_i is the mean of the recruitment values for background years associated with key event i . Significance is determined by referring to a table of Student's t-distribution with $N_E - 1$ degrees of freedom.

- 6) Q_R = same as (5) but significance is determined by randomization.

Program EPOWER uses slightly modified versions of the following routines: RAN3, GASDEV, AVEVAR, TTEST, TPTEST, BETAI, BETACF, GAMMLN. (© 1986 by Numerical Recipes Software. Reproduced by permission, from the book Numerical Recipes: The Art of Scientific Computing, published by Cambridge University Press.)

PROGRAM PREPARATION

The program is written in standard Fortran 77. Although memory requirements are small, so the program could be run on a microcomputer, the execution time is substantial. We used about 100 hours of CPU time on a mainframe supercomputer to generate the series of power curves in Prager and Hoenig (in prep). It took approximately 20 minutes of CPU time to estimate the power for one set of conditions.

The program operates in batch mode - all input is read from a single data file called EPOWER.DAT and output is written to two files. One file, EPOWER.OUT, provides for each test statistic the following information: number of simulated data sets for which a significant result was found, the estimated power, and an approximate 95% confidence interval (based on the normal approximation to a binomial).

The other output file, EPOWER.DET, provides more detailed output (DET for detail). The file contains the value of the test statistic, and the estimated significance level, for every data set generated.

The program's arrays are dimensioned to allow up to 10 key event years to occur in a 50 year time series. If this is satisfactory, the program can be compiled and run without modification.

DATA INPUT

A sample data input file is presented in Appendix III. Parameters of the simulation are read in free format from a data file called EPOWER.DAT, as follows:

- line 1: Number of years in each simulated data set.
- line 2: Number of key event years in each data set.
- line 3: Amount of difference in mean recruitment between background and key event years.
- line 4: Slope of (trend in) recruitment over time (yr^{-1}).
- line 5: Seed for random number generator RAN3 (from Numerical Recipes). Number should be negative.
- line 6: Enter a 1 if only years before each key event are used as background years; enter a 2 if years before and after are used as background years.
- line 7: Number of background years preceding each key event year (and also succeeding each key event year, if two-sided epochs are used).
- line 8: Number of data sets generated (outer trials - to be analyzed by the Monte Carlo randomization test)
- line 9: Number of Monte Carlo randomizations (inner trials - for analyzing a given data set)
- line 10: Autoregressive parameter

PROGRAM OUTPUT

File EPOWER.OUT provides a summary of the simulated conditions and the estimated power

levels with approximate 95% confidence limits.
The information is provided as follows:

```

title ("Power analysis ....")
number of data sets generated randomly
number of years in each data set generated
number of key event years in each data set
definition of an epoch
model type (autoregressive vs. uncorrelated)
delta, the amount added to the recruitment
value to make the year a key event year
slope of trend: the amount by which the
expected recruitment changes per year
autoregressive parameter for the AR1 model
number of Monte Carlo randomizations for a
given generated data set
random number seed for RAN3
title ("Summary of Results")
Statistics
number of significant results (rejections of
H0)
estimated power
lower 95% confidence limit
upper 95% confidence limit

```

The detailed output file, EPOWER.DET, provides the exact same summary of simulated conditions as EPOWER.OUT. It then provides, for every simulated data set, the value of the statistics and their estimated significance levels.

ACKNOWLEDGMENTS

We thank the Computer Center and the Department of Oceanography at Old Dominion University for providing computer facilities and support for program development. We especially thank William Vetterling and Numerical Recipes Software for permission to reproduce several proprietary software routines. Marianne Eckenswiler and Janice Lannon typed the manuscript.

REFERENCES

- Box, G. E. P. and G. M. Jenkins. 1970. Time series analysis-forecasting and control. Holden-Day, San Francisco, CA.
- Edgington, E. S. 1987. Randomization tests, second edition. Marcel Dekker, New York.
- Haurwitz, M. W. and G. W. Brier. 1981. A critique of the superposed epoch analysis method: its application to solar-weather relations. Monthly Weather Review 109: 2074-2079.
- Prager, M. H. and J. M. Hoenig. In press. Superposed epoch analysis: a randomization test of environmental effects on recruitment, with application to chub mackerel. Trans. Amer. Fish. Soc. 118 (6).
- Prager, M. H. and J. M. Hoenig. In prep. Can we determine the significance of key events on time series of recruitment? - a power analysis. To be submitted to Trans. Amer. Fish. Soc.
- Press, W. H., B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. 1986. Numerical recipes - the art of scientific computing. Cambridge Univ. Press, New York.

Appendix I - Program EPOCH
sample input file (EPOCH.DAT) with annotations

40	Number of years (rows) of data.
10000	# Monte Carlo trials.
8	# of key events.
1933 -1	Key event year ID's and expected directions of
1941 +1	difference between key event year and background years.
1951 +1	
1955 -1	
1957 +1	
1958 +1	
1959 +1	
1964 -1	
1	Minimum spacing for key events.
'W'	Test statistic used.
-12345	Seed for random # generator (negative integer).
2	Epoch definition: 1 for background years to the left. 2 for left & right.
2	Number of background years preceding (and possibly following) key events.
1929 0.701664	Year ID, recruitment value.
1930 0.738894	
1931 0.447222	
1932 0.229122	
1933 0.127301	
1934 0.077382	
1935 0.124930	
1936 0.363709	
1937 0.388174	
1938 0.671057	
1939 0.530253	
1940 0.736018	
1941 1.416940	
1942 0.655209	
1943 0.400316	
1944 0.338645	
1945 0.160253	
1946 0.205481	
1947 2.403677	
1948 2.454197	
1949 0.154710	
1950 0.040119	
1951 0.052797	
1952 1.682868	
1953 10.02919	
1954 1.189460	
1955 1.132808	
1956 0.211062	
1957 0.362257	
1958 2.482832	
1959 2.089239	
1960 2.041735	
1961 1.052638	
1962 0.136613	
1963 0.056806	
1964 0.023837	
1965 0.095131	
1966 0.0326	
1967 0.2775	
1968 1.5857	

Appendix II - Program EPOCH Listing

```

C *****
C PROGRAM EPOCH
C *****
C Prager/Hoenig/Payton November, 1989
C
C Use superposed epoch analysis to test hypotheses.
C This program uses an epoch of specified half-width on one side,
C or centered around, the "key event." It also expects to see
C the resulting effect in the same year as the key event.
C Reference: Prager, M. H. and J. M. Hoenig. (in press).
C Superposed epoch analysis: a randomization test of
C enviromental effects on recruitment, with application
C to chub mackerel. Trans. Amer. Fish. Soc. 118(6)
C -----
C Input Data File Structure (Free Format):
C -----
C Line 1: NYear Number of years (rows) in data matrix.
C Line 2: NSim Number of Monte Carlo trials.
C Line 3: NKey Number of key events.
C Lines 4 to
C NKey+3: Pairs of numbers, one pair for each key event.
C Key() Year ID's of key events.
C ISign(): +1 for key event(peak), -1 for reflected
C key event (trough).
C Next Line: MinDif Minimum spacing of key events.
C Next Line: AStat Flag for statistic to be used.
C 'T': T-statistic.
C 'D': D-statistic.
C 'W': W-statistic.
C Next Line: MSeed Negative seed for RAN3
C Next Line: NSide Sides to be used for MWidth.
C '1': One sided. Uses left side.
C '2': Two sided.
C Next Line: MWidth Half-width of epoch on each side of
C key event if two sided, full-width if one
C sided.
C Next NYear Lines:
C pairs of IYear() and XX() values.
C IYear(): Year ID's of each row of data matrix
C XX(): Dependent variable of each row
C -----
C *** Units & ddnames of files ***
C
C Unit DDname Action Contents
C -----
C Unit 12 epoch.dat read Data
C Unit 13 epoch.out write Main Output File
C Unit 14 epoch.sig write Series of P-values for plotting
C -----
C *** Major variables ***
C
C IYear() Year ID's (col. 1 of input data matrix)
C XX() Dependent variable (col. 2 of input data matrix)
C NYear Number of years (rows) in the input data matrix
C NKey Number of key events
C Key() Year ID's of key events (input)
C Key2() Year ID's of key events (Monte Carlo)
C NSim Number of Monte Carlo trials
C ILowYr Value of IYear(1)
C IHiYr Value of IYear(NYear)

```

```

C   MinDif           Minimum spacing of key events.  For example, if          0061
C                   MinDif=2, then key events in both 1980 and 1981          0062
C                   would not be allowed.  This is used in the              0063
C                   Monte Carlo part of the technique, rather than to       0064
C                   check the real data.                                   0065
C   AStat            Statistic to use -- see above                          0066
C   ISign             Sign of key event: +1 for expected peak,              0067
C                   -1 for expected trough                                0068
C *****                                                    0069
C   *** Declarations ***                                                    0070
C                                                                 0071
C   integer I, ILowYr, Int, IHiYr, J, K, NAbove, MinDif                    0072
C   integer ISign(50), ABS, MSeed, Mod                                    0073
C   integer Key(50), Key2(50), LWidth, RWidth                             0074
C   double precision P(10000), TrialStat(10000), XX(100)                  0075
C   double precision DataStat, Stat                                       0076
C   double precision DFloat, Ran3                                         0077
C   external Ran3, ReadData, CompT, CompD, CompW, WrtOut                  0078
C   character*1 AStat                                                    0079
C                                                                 0080
C   integer IYear(100), NYear, NKey, NSim, ISeed                          0081
C                                                                 0082
C   common/bigblock/XX,IYear,NYear,NKey,NSim,ISeed,P,TrialStat,          0083
C   & ILowYr,IHiYr,LWidth,RWidth                                         0084
C -----                                                    0085
C                                                                 0086
C   *** Read the data file                                                0087
C                                                                 0088
C   Call ReadData(Key,ISign,AStat,MinDif,MSeed)                          0089
C                                                                 0090
C   *** Find the Test Statistic (T, D, or W) on the True Data            0091
C                                                                 0092
C   IF (AStat .EQ. 'T') then                                              0093
C       call CompT(ISign,Key,Stat)                                         0094
C   ELSEIF (AStat .EQ. 'D') then                                          0095
C       call CompD(ISign,Key,Stat)                                         0096
C   ELSEIF (AStat .EQ. 'W') THEN                                          0097
C       call CompW(ISign,Key,Stat)                                         0098
C   ELSE                                                                    0099
C       Print *, 'ERROR: Bad type of statistic: ',AStat                  0100
C       stop                                                                0101
C   ENDIF                                                                    0102
C   DataStat = Stat                                                         0103
C                                                                 0104
C                                                                 0105
C   *** Do NSim Monte Carlo Trials & Compute                             0106
C       Test Statistic for Each                                           0107
C                                                                 0108
C   print *, ' Beginning simulations....'                                0109
C   First initialize the counter of larger P-values                      0110
C   NAbove = 1                                                             0111
C   Do 100 I=1, NSim                                                       0112
C       if (mod(I,1000) .EQ. 0) print *, ' Simulation:',I                 0113
C       Pick out NKey years at random                                     0114
C       Do 110 J=1, NKey                                                    0115
105         Key2(J) = ILowYr +                                             0116
C       & int( (IHiYr - ILowYr) * Ran3(iseed) + 0.5d0)                   0117
C       If a year has missing data, choose another one                   0118
C       IF (XX(Key2(J)-ILowYr+1) .LT. -9000.0d0) then                     0119
C           print *, 'Year with missing data replaced:', Key2(J)         0120

```



```

        goto 105
    ENDIF
C      If a year is too close to another year, choose another one
    do 107, k=1,J-1
    IF (ABS(Key2(J)-Key2(K)) .LT. MinDif) THEN
C      print *, 'Years too close:', (Key2(L),L=1,J)
        goto 105
    ENDIF
107    continue
110    continue
C
    IF (AStat .EQ. 'T') then
        Call CompT(ISign,Key2,Stat)
    ELSEIF (AStat .EQ. 'D') then
        call CompD(ISign,Key2,Stat)
    ELSEIF (AStat .EQ. 'W') then
        call CompW(ISign,Key2,Stat)
    ENDIF
C
    TrialStat(I) = Stat
    if (Stat .GT. DataStat) NAbove = NAbove + 1
C    Compute running P(i) for plotting
    P(I) = dfloat(NAbove) / dfloat(I+1)
100    continue
C
C    *** Write Output & End
C
    Call WrtOut(DataStat,NAbove,Key,ISign,AStat,MinDif,MSeed)
C
    end
C
C    *** End of main program (EPOCH)
C *****
C *****
C
    SUBROUTINE ReadData(Key,ISign,AStat,MinDif,MSeed)
C
    integer I, ILowYr, IHiYr, Abs, MinDif
    integer IYear(100), ISign(50), Key(50), LWidth, RWidth
    integer NYear, NKey, NSim, ISeed, MSeed, NSide, MWidth
    double precision TrialStat(10000), P(10000), XX(100)
    character*1 AStat
C
    common/bigblock/XX,IYear,NYear,NKey,NSim,ISeed,P,TrialStat,
& ILowYr,IHiYr,LWidth,RWidth
C
    open (unit=12, file='epoch.dat', status='OLD')
    read (12,*) NYear, NSim, NKey
    do 110 i=1, NKey
        read (12,*) Key(i), ISign(i)
        IF (ABS(ISign(i)) .NE. 1) then
            print *, 'ERROR: Bad sign flag for key event #',i
            print *, 'Value must be 1 or -1'
            STOP
        ENDIF
110    continue
    read (12,*) MinDif
    read (12,*) AStat
    read (12,*) MSeed
    read (12,*) NSide

```



```

close (unit=12)                                0241
return                                          0242
end                                              0243
C *****                                          0244
C                                          0245
C      Subroutine CompT(ISign,Key,TStat)          0246
C                                          0247
C      Arguments:                                0248
C      Key -- (Input) An array of Key Events.   0249
C              Key(i) is a value                 0250
C              equal to the IYear value of the   0251
C              ith key event.                    0252
C      ILowYear -- (Input) The first key year.   0253
C      TStat -- (Output) The returned value of  0254
C              the T-Statistic.                 0255
C                                          0256
C      Integer Key(50), I, I3, NB, JJ(50), I2,  0257
C      Integer IYear(100), ISign(50)            0258
C      Double precision B, BMean, EMean, ESSD,  0259
C      Double precision SD, dfloat, sqrt        0260
C      double precision P(10000), TrialStat(10000), XX(100)
C      integer LWidth, RWidth, NYear, NKey, NSim, ISeed, K
C                                          0261
C      common/bigblock/XX,IYear,NYear,NKey,NSim,ISeed,P,TrialStat,
C      & ILowYr,IHiYr,LWidth,RWidth             0262
C                                          0263
C      *** Put into the JJ array indexes to the XX array for the
C      NKey key-event years. The indexes are used below. 0264
C      *** While doing it, check for minus signs in input. 0265
C      They are not appropriate for the T-statistic.      0266
C                                          0267
C      DO 105 I = 1, NKey                          0268
C          IF (ISign(i) .LT. 0) THEN                0269
C              print *, 'ERROR: "-1" flag input for key year',Key(i)
C              print *, 'Only "+1" flags are allowed for T-statistic.'
C              STOP
C          ENDIF
C          JJ(I) = Key(I) - ILowYr + 1
C      CONTINUE
C                                          0270
C      *** Compute E-Mean (Mean of Event Values)          0271
C                                          0272
C      EMean = 0.0d0
C      DO 110 I = 1, NKey
C          EMean = EMean + XX(JJ(I))
C      CONTINUE
C      EMean = EMean / dfloat(NKey)
C                                          0273
C      *** Compute ESSD (Sum of squared deviations for Event Col.)
C                                          0274
C      ESSD = 0.0d0
C      DO 120 I = 1, NKey
C          ESSD = ESSD + (XX(JJ(I)) - EMean) ** 2
C      CONTINUE
C                                          0275
C      *** Compute B-mean (Background)
C                                          0276
C      BMean = 0.0d0
C      NB = 0
C      The first loop goes across the row
C      DO 140 I = LWidth, RWidth
C          if (I .EQ. 0) goto 140
C      The inner loop goes down a column

```

0301
0302
0303
0304
0305
0306
0307
0308
0309
0310
0311
0312
0313
0314
0315
0316
0317
0318
0319
0320
0321
0322
0323
0324
0325
0326
0327
0328
0329
0330
0331
0332
0333
0334
0335
0336
0337
0338
0339
0340
0341
0342
0343
0344
0345
0346
0347
0348
0349
0350
0351
0352
0353
0354
0355
0356
0357
0358
0359
0360

```

C ***** 0361
C 0362
C      Subroutine CompD(ISign,Key,DStat) 0363
C 0364
C      Compute D-statistic. D is the difference between the mean of 0365
C      values in event years and in background years. 0366
C 0367
C      Arguments: 0368
C      Key      (Input) An array of Key Events. Key(i) is a value 0369
C               equal to the IYear value of the ith key event. 0370
C      ILowYr   (Input) The first key year. 0371
C      DStat    (Output) The returned value of the D-Statistic. 0372
C 0373
C      Integer Key(50), I, NB, JJ(50), I2, I3, ILowYr, IHiYr 0374
C      Double precision B, BMean, EMean, DStat, DFloat 0375
C      double precision P(10000), TrialStat(10000), XX(100) 0376
C      integer IYear(100), ISign(50) 0377
C      integer LWidth, RWidth, NYear, NKey, NSim, ISeed, K 0378
C 0379
C      common/bigblock/XX,IYear,NYear,NKey,NSim,ISeed,P,TrialStat, 0380
C      & ILowYr,IHiYr,LWidth,RWidth 0381
C 0382
C      *** Put into the JJ array indexes to the XX array for the 0383
C      NKey key-event years. The indexes are used below. 0384
C      *** While doing it, check for minus signs in input. 0385
C      They are not appropriate for the D-statistic. 0386
C 0387
C      DO 105 I = 1, NKey 0388
C          IF (ISign(i) .LT. 0) THEN 0389
C              print *, 'ERROR: "-1" flag found at key year',Key(i) 0390
C              print *, 'Only "+1" flags are allowed for D-statistic.' 0391
C              STOP 0392
C          ENDIF 0393
C          JJ(I) = Key(I) - ILowYr + 1 0394
105  CONTINUE 0395
C 0396
C      *** Compute E-Mean (Mean of Event Values) 0397
C 0398
C      EMean = 0.0d0 0399
C      DO 110 I = 1, NKey 0400
C          EMean = EMean + XX(JJ(I)) 0401
110  CONTINUE 0402
C      EMean = EMean / dfloat(NKey) 0403
C 0404
C      *** Compute B-mean (Background) 0405
C 0406
C      BMean = 0.0d0 0407
C      NB = 0 0408
C      The first loop goes across the row 0409
C      DO 140 I = LWidth, RWidth 0410
C          if (I .EQ. 0) goto 140 0411
C      The inner loop goes down a column 0412
C      DO 150 I2 = 1, NKey 0413
C          K indexes the XX array (1 ... NYear) 0414
C          K = JJ(I2) + I 0415
C          Check for out-of-range background years 0416
C          IF ((K .LT. 1) .OR. (K .GT. NYear)) THEN 0417
C              B = -9999.0d0 0418
C          print *, 'Out of range year & row:',Key(I2)+I,K 0419
C          ELSE 0420

```

```

        B = XX(K)                                0421
C      Check for a background year that's on the list of key      0422
C      years.  If it is, then make the data value missing.      0423
        DO 120 I3 = 1, NKey                        0424
            IF (Key(I3) .EQ. IYear(K)) THEN          0425
                B = -9999.0d0                        0426
C            print *, 'Background is also key', Key(I3), IYear(K) 0427
            ENDIF                                    0428
120      continue                                    0429
        ENDIF                                        0430
C      Check for missing values ( = -9999)              0431
        IF (B .GT. -9000.0d0) THEN                  0432
            BMean = BMean + B                      0433
            NB = NB + 1                             0434
        ENDIF                                        0435
150      CONTINUE                                    0436
140      CONTINUE                                    0437
        BMean = BMean / dfloat(NB)                 0438
C                                                    0439
C      *** Compute D-Statistic                          0440
C                                                    0441
        DStat = EMean - BMean                      0442
C                                                    0443
C      *** Return                                       0444
C                                                    0445
        RETURN                                     0446
        END                                         0447
C                                                    0448
C *****                                              0449
C                                                    0450
        Subroutine CompW(ISign,Key,W)               0451
C                                                    0452
C      Compute W-statistic.  W is analogous to a paired      0453
C      Student's t statistic.                            0454
C                                                    0455
C      Arguments:                                       0456
C      Key      (Input) An array of Key Events.  Key(i) is a value 0457
C               equal to the IYear value of the ith key event.    0458
C      ILowYr   (Input) The first key year.              0459
C      W       (Output) The returned value of the W-Statistic.    0460
C                                                    0461
        Integer Key(50), I, NDif, JJ(50), I2, I3, ILowYr, IHiYr 0462
        Double precision B, E, DifMean, DifVar, W        0463
        Double precision dfloat, sqrt                  0464
        Double precision Dif(100)                      0465
        integer IYear(100), ISign(50)                  0466
        integer LWidth, RWidth, NYear, NKey, NSim, ISeed, K 0467
        double precision P(10000), TrialStat(10000), XX(100) 0468
C                                                    0469
        common/bigblock/XX,IYear,NYear,NKey,NSim,ISeed,P,TrialStat, 0470
        & ILowYr,IHiYr,LWidth,RWidth                  0471
C                                                    0472
C      *** Put into the JJ array indexes to the XX array for the 0473
C      NKey key-event years.  The indexes are used below.      0474
C                                                    0475
        DO 105 I = 1, NKey                            0476
            JJ(I) = Key(I) - ILowYr + 1                0477
105      CONTINUE                                    0478
C                                                    0479
C      *** Compute differences between E and B and accumulate mean 0480

```

```

C                                     0481
DifMean = 0.0d0                      0482
NDif = 0                             0483
C The first loop goes across the row of the epoch 0484
DO 140 I = LWidth, RWidth            0485
  if (I .EQ. 0) goto 140             0486
C The inner loop goes down a column of the epoch 0487
DO 150 I2 = 1, NKey                  0488
  C Get the "E" value for this row      0489
  E = XX(JJ(I2))                     0490
  C Get the "B" value for this comparison 0491
  C Store the XX index for this B in K   0492
  K = I + JJ(I2)                     0493
  C Check for out-of-range background years 0494
  IF ((K .LT. 1) .OR. (K .GT. NYear)) THEN 0495
    B = -9999.0d0                     0496
  C print *, 'Out of range year & row:', Key(I2)+I, K 0497
  ELSE                                0498
    B = XX(K)                         0499
  C Check for a background year that's on the list of key 0500
  C years. If it is, then make the data value missing. 0501
  DO 120 I3 = 1, NKey                0502
    IF (Key(I3) .EQ. IYear(K)) THEN 0503
      B = -9999.0d0                   0504
    C print *, 'Background is also key', Key(I3), IYear(K) 0505
    ENENDIF                           0506
  120 continue                        0507
  ENENDIF                             0508
  C Check for missing values ( = -9999) 0509
  IF (B .GT. -9000.0d0) THEN          0510
    NDif = NDif + 1                   0511
    Dif(NDif) = Dfloat(ISign(I2)) * (E - B) 0512
    DifMean = DifMean + Dif(NDif)     0513
  ENENDIF                             0514
150 CONTINUE                          0515
140 CONTINUE                          0516
DifMean = DifMean / DFloat(NDif)      0517
C                                     0518
C *** Compute DifVar (Variance of differences) 0519
C                                     0520
DifVar = 0.0d0                        0521
DO 160 I = 1, NDif                    0522
  DifVar = DifVar + (Dif(I) - DifMean) ** 2 0523
160 CONTINUE                          0524
DifVar = DifVar / DFloat(NDif - 1)    0525
C                                     0526
C *** Compute W-Statistic by dividing mean difference by 0527
C the std. error of the mean difference. See Snedecor 0528
C and Cochran p. 85.                  0529
C                                     0530
W = DifMean / SQRT(DifVar/dfloat(NDif)) 0531
C                                     0532
C *** Return                           0533
C                                     0534
RETURN                                0535
END                                   0536
C                                     0537
C *****                                0538
C                                     0539
SUBROUTINE WrtOut(DataStat, NAbove, Key, ISign, Astat, MinDif, MSeed) 0540

```

```

C
integer I, J, NAbove, Key(50), ILowYr, IHiYr, MOD
integer LWidth, RWidth, MSeed, MinDif
integer IYear(100), ISign(50), NYear, NKey, NSim, ISeed
double precision XX(100), P(10000), DataStat, TrialStat(10000)
character*1 AStat

C
common/bigblock/XX,IYear,NYear,NKey,NSim,ISeed,P,TrialStat,
& ILowYr,IHiYr,LWidth,RWidth

C
open (unit=13, file='epoch.out', status='UNKNOWN')

C
*** Write Header
C
write (13,*) ' Results of Superposed Epoch Analysis'
write (13,*) ' -----'

C
*** Write Information on data
C
I = RWidth - LWidth + 1
write (13,501) I
501 format (' Width of epoch: ',I8)
write (13,502) -LWidth
502 format (' Background Years :',/,8x,'The ',i1,' years on the Left')
write (13,503) RWidth
503 format (8x,'The ',i1,' years on the Right')
write (13,504) MSeed
504 format (' Random number seed: ',I8)
write (13,505) NYear
505 format (' Number of rows in input data ..... ',I5)
write (13,610) ILowYr
610 format (' ID of first row ..... ',I5)
write (13,620) IHiYr
620 format (' ID of last row ..... ',I5)
write (13,590) IHiYr-ILowYr+1
590 format (' Range of ID values ..... ',I5)
C
*** Report on Missing Values
C
J = 0
do 250 I = 1, NYear
  if (XX(I) .LT. -9000) then
    write (13,630) IYear(I)
    J = J + 1
  endif
250 continue
630 format (' Missing data at ID # ..... ',I5)
write (13,640) J
640 format (' Number of rows with missing data ..... ',I5)
C
*** Write Information on key years
C
write (13,507) NKey
507 format (' Number of key periods: ',I4)
write (13,510) (I,Key(I),ISign(I), I=1,NKey)
510 format (' Key period',1X,'#',I2,':',2X,I5,3X,'Sign:',SP,I4,S)
C
*** Write the statistics
C
write (13,512) AStat

```


	DO 13 K=1,4	0661
	DO 12 I=1,55	0662
	MA(I)=MA(I)-MA(1+MOD(I+30,55))	0663
	IF(MA(I).LT.MZ)MA(I)=MA(I)+MBIG	0664
12	CONTINUE	0665
13	CONTINUE	0666
	INEXT=0	0667
	INEXTP=31	0668
	IDUM=1	0669
	ENDIF	0670
	INEXT=INEXT+1	0671
	IF(INEXT.EQ.56) INEXT=1	0672
	INEXTP=INEXTP+1	0673
	IF(INEXTP.EQ.56) INEXTP=1	0674
	MJ=MA(INEXT)-MA(INEXTP)	0675
	IF(MJ.LT.MZ) MJ=MJ+MBIG	0676
	MA(INEXT)=MJ	0677
	RAN3=MJ*FAC	0678
	RETURN	0679
	END	0680
C		0681
C	*****	0682
C	End of EPOCH.FOR	0683
C	*****	0684

Appendix III - Program EPOWER
sample input file (EPOWER.DAT) with annotations

35	Number of years in each simulated data set.
3	Number of key event years in each data set.
1.5d0	Amount of difference in mean recruitment between background and key event years.
0.0d0	Time slope of recruitment.
-72365	Negative seed for RAN3.
2	Epower definition: 1 for background years to the left. 2 for left & right.
2	Number of background years preceding (and possibly following) key events.
10	Number of data sets generated (outer trials).
10000	Number of Monte Carlo trials per data set (inner trials).
0.2d0	Autoregressive parameter.

Appendix IV - Program EPOWER Listing

```

C ***** 0001
C EPOWER.FOR 0002
C Examine the power of superposed epoch analysis. 0003
C ***** 0004
C PROGRAM EPOWER 0005
C Prager/Hoenig/Payton November, 1989 0006
C ----- 0007
C Input Data File Structure (Free Format): 0008
C ----- 0009
C Line 1: NYear Number of years in each simulated data set. 0010
C Line 2: NKey Number of key event years in each data set. 0011
C Line 3: Delta Amount of difference in mean recruitment 0012
C between background and key event years. 0013
C Line 4: Slope Time slope of recruitment. 0014
C Line 5: MSeed Negative seed for RAN3. 0015
C Line 6: NSide Sides to be used for MWidth. 0016
C '1': One sided. Uses left side. 0017
C '2': Two sided. 0018
C Line 7: MWidth Half-width of epoch on each side of 0019
C key event if two sided, full-width if one 0020
C sided. 0021
C Line 8: NData Number of data sets generated (outer trials). 0022
C Line 9: NMonte Number of Monte Carlo trials (inner trials). 0023
C Line 10: ARparm Autoregressive parameter. 0024
C ===== 0025
C File Unit DDname Action Contents 0026
C ----- 0027
C Unit 12 EPOWER.DAT read Input parameters 0028
C Unit 13 EPOWER.OUT write Main output file 0029
C Unit 14 EPOWER.DET write Detailed output file 0030
C ===== 0031
C *** Major variables *** 0032
C 0033
C XX(i) Recruitment in year i. 0034
C NYear Number of years in each simulated data set. 0035
C NKey Number of key events in each simulated data set. 0036
C NData Number of phony data sets to be evaluated. 0037
C Key(j) Year (index) of key event j (j = 1 ... NKey) 0038
C Key2() Same, in Monte Carlo simulation. 0039
C NMonte Number of Monte Carlo trials for randomization. 0040
C ARparm Autoregressive parameter. 0041
C ARFlag Switch used to determine which way to generate time 0042
C series 0043
C 0044
C StatD Difference statistic. 0045
C PD Prob > D, determined by randomization 0046
C 0047
C StatT Student's t statistic (= Prager/Hoenig "T" statistic) 0048
C PRT Prob > t, determined by randomization 0049
C PTT Prob > t, determined from tables 0050
C 0051
C StatW Quasi-paired t, as used in first paper by Hoenig & 0052
C Prager 0053
C PW Prob > W, determined by randomization 0054
C 0055
C StatQ Student's paired t statistic (classic) 0056
C PRQ Prob > Q, determined by randomization. 0057
C PTQ Prob > Q, determined from tables. 0058
C 0059
C NTSigT (e.g.; others similar) # of instances with PTT < alpha 0060

```

```

C PTSigT          NTSigT / NData                                0061
C MSeed           Original random number seed                  0062
C ISeed           Running random number seed                   0063
C *****                                                0064
C   *** Declarations ***                                       0065
C                                                           0066
C   integer Key(10), IData, MSeed                             0067
C   integer NSigD, NRSigT, NTSigT, NSigW, NTSigQ, NRSigQ       0068
C   logical xmonte                                           0069
C   double precision alpha, aninv,xi                          0070
C   double precision StatD, StatT, StatW, StatQ               0071
C   double precision PD, PRT, PTT, PW, PRQ, PTQ               0072
C   double precision PSigD, PRSigT, PTSigT, PSigW, PRSigQ, PTSigQ 0073
C                                                           0074
C *****                                                0075
C   Common declarations and statement for EPOWER.FOR          0076
C *****                                                0077
C                                                           0078
C   integer ndata, nmonte, nyear, nkey, iseed,                0079
C   +   mindif, arflag, LWidth, RWidth, MWidth                0080
C   double precision xx(50), delta, slope, arparm             0081
C                                                           0082
C   common/bigblock/xx, delta, slope, mindif, lwidth, rwidth, 0083
C   +   ndata, nmonte, nyear, nkey, iseed, arflag, arparm, mwidth 0084
C                                                           0085
C *****                                                0086
C                                                           0087
C   common/flags/xmonte                                       0088
C   MinDif = 1                                                 0089
C   Call ReadData(MSeed)                                       0090
C   alpha = 0.05d0                                             0091
C   nsigd = 0                                                  0092
C   ntsigt = 0                                                  0093
C   nrsigt = 0                                                  0094
C   nsigw = 0                                                  0095
C   ntsigq = 0                                                  0096
C   nrsigq = 0                                                  0097
C   xmonte = .TRUE.                                           0098
C   print *, 'Power study beginning....'                      0099
C                                                           0100
C   *** Run the analysis for each of NDATA fake data sets    0101
C                                                           0102
C   Do 1001 IData = 1, NData                                    0103
C     xi = dfloat(IData)/10.0d0                                0104
C     if (int(xi) .eq. xi) print *, 'IData = ', IData         0105
C                                                           0106
C     *** Generate a data set:                                  0107
C                                                           0108
C     if (ARFlag .eq. 0) then                                   0109
C       call MDataL(Key)                                        0110
C     else if (ARFlag .eq. 1) then                               0111
C       call MDataAR(Key)                                       0112
C     else                                                       0113
C       print *, 'Bad value for ARFlag.'                       0114
C       stop                                                    0115
C     endif                                                      0116
C                                                           0117
C     *** Compute the statistics: D, T, W, and Paired t (Pt): 0118
C                                                           0119
C     Call CompD(Key, StatD)                                    0120

```

```

      Call CompT(Key,StatT,PTT)
      Call CompW(Key,StatW)
      Call CompQ(Key,StatQ,PTQ)
C
C      *** Compute significance probabilities of D, T, W, Q by
C      randomization
C
      xmonte = .FALSE.
      Call ranprob(statd,statt,statw,statq,PD,PRT,PW,PRQ)
      xmonte = .TRUE.
C
C      *** Increment counters of significant statistics
C
      if (pd .lt. alpha) nsigd = nsigd + 1
      if (prt .lt. alpha) nrsigt = nrsigt + 1
      if (ptt .lt. alpha) ntsigt = ntsigt + 1
      if (pw .lt. alpha) nsigw = nsigw + 1
      if (prq .lt. alpha) nrsigq = nrsigq + 1
      if (ptq .lt. alpha) ntsigq = ntsigq + 1
C
      call wrtout(key,idata,statd,statt,statw,statq,
+      pd,prt,ptt,pw,prq,ptq,mseed)
C
1001 continue
C
C      *** Compute proportion of cases found significant
C
      ANINV = 1.0d0 / dfloat(NData)
      PSigD = dfloat(NSigD) * ANINV
      PRSigT = dfloat(NRSigT) * ANINV
      PTSigT = dfloat(NTSigT) * ANINV
      PSigW = dfloat(NSigW) * ANINV
      PRSigQ = dfloat(NRSigQ) * ANINV
      PTSigQ = dfloat(NTSigQ) * ANINV
C
C      *** Call the summary output routine to print a summary of results
C
      Call WrtSum(NSigD, NRSigT, NTSigT, NSigW, NRSigQ, NTSigQ,
+      PSigD, PRSigT, PTSigT, PSigW, PRSigQ, PTSigQ)
C
C      *** End of main program (EPOWER)
      end
C
C      *****
C      *****
C      SUBROUTINE ReadData(MSeed)
C
      integer MSeed, NSide
C
C      *****
C      Common declarations and statement for EPOWER.FOR
C      *****
C
      integer ndata, nmonte, nyear, nkey, iseed,
+      mindif, arflag, LWidth, RWidth, MWidth
      double precision xx(50), delta, slope, arparm

```

0121
0122
0123
0124
0125
0126
0127
0128
0129
0130
0131
0132
0133
0134
0135
0136
0137
0138
0139
0140
0141
0142
0143
0144
0145
0146
0147
0148
0149
0150
0151
0152
0153
0154
0155
0156
0157
0158
0159
0160
0161
0162
0163
0164
0165
0166
0167
0168
0169
0170
0171
0172
0173
0174
0175
0176
0177
0178
0179
0180

	common/bigblock/xx, delta, slope, mindif, lwidth, rwidth,	0181
	+ ndata, nmonte, nyear, nkey, iseed, arflag, arparm, mwidth	0182
C		0183
C	*****	0184
C		0185
	open (unit=12, file='EPOWER.DAT', status='old')	0186
	read (12,*) NYear	0187
	read (12,*) NKey	0188
	read (12,*) Delta	0189
	read (12,*) Slope	0190
	read (12,*) MSeed	0191
	read (12,*) NSide	0192
	read (12,*) MWidth	0193
	read (12,*) NData	0194
	read (12,*) NMonte	0195
	read (12,*) ARparm	0196
C		0197
C	*** Seed random number generator with a negative number.	0198
C	Use -ABS(SEED) from input file if it's not 0. Use a	0199
C	fixed number (-85625) if given seed is 0.	0200
C		0201
	IF (MSeed .LT. 0) THEN	0202
	ISeed = MSeed	0203
	ELSE IF (MSeed .GT. 0) THEN	0204
	ISeed = -1 * MSeed	0205
	ELSE IF (MSeed .EQ. 0) THEN	0206
	MSeed = -815625	0207
	ISeed = MSeed	0208
	END IF	0209
C		0210
C	*** Determine the left and right widths on each side of the	0211
C	key events.	0212
C		0213
	if (NSide .eq. 2) then	0214
	LWidth = -MWidth	0215
	RWidth = MWidth	0216
	else if (NSide .eq. 1) then	0217
	LWidth = -MWidth	0218
	RWidth = 0	0219
	else	0220
	print *, 'ERROR: NSide equals ', NSide	0221
	print *, 'NSide must equal 1 or 2 '	0222
	stop	0223
	endif	0224
C		0225
C	*** Set ARFlag (Flag for autoregression)	0226
C		0227
	if (abs(ARparm) .lt. 1.0d-10) then	0228
	arflag = 0	0229
	else	0230
	arflag = 1	0231
	end if	0232
C		0233
	close (unit=12)	0234
	return	0235
	end	0236
C		0237
C	*****	0238
C	*****	0239
C	*****	0240


```

C                                     0241
SUBROUTINE MDataL(Key)                                     0242
C                                     0243
C      Generate a random dataset for the epoch power test. 0244
C      This generates data for a linear (non-autoregressive) model. 0245
C                                     0246
      double precision GasDev, Ran3                         0247
      integer JPick(50),I,J,Key(10)                        0248
C                                     0249
C                                     0250
C *****                                     0251
C      Common declarations and statement for EPOWER.FOR    0252
C *****                                     0253
C                                     0254
      integer ndata, nmonte, nyear, nkey, iseed,           0255
+      mindif, arflag, LWidth, RWidth, MWidth             0256
      double precision xx(50), delta, slope, arparm        0257
C                                     0258
      common/bigblock/xx, delta, slope, mindif, lwidth, rwidth, 0259
+      ndata, nmonte, nyear, nkey, iseed, arflag, arparm, mwidth 0260
C                                     0261
C *****                                     0262
C                                     0263
C                                     0264
      JPick = 0 if a background, 1 if a key event year.    0265
      Delta = amount by which key years differ from background. 0266
      NKey = number of key event years                     0267
      XX = recruitment                                     0268
C                                     0269
      do 100 i = 1, NYear                                   0270
C      * Generate normal random recruitments for each year 0271
      XX(i) = gasdev(ISeed) + Slope * i                    0272
      JPick(i) = 0                                         0273
100  continue                                             0274
C                                     0275
C      *** Pick NKey key event years, and make sure they are unique 0276
C                                     0277
      do 150 j = 1, NKey                                   0278
120  Key(j) = 1 + int(dfloat(nyear) * RAN3(iseed))        0279
      IF (JPick(Key(j)) .eq. 1) THEN                      0280
        goto 120                                          0281
      ELSE                                                0282
        JPick(Key(j)) = 1                                0283
        XX(Key(j)) = XX(Key(j)) + Delta                  0284
      ENDIF                                              0285
150  continue                                             0286
      return                                              0287
      end                                                0288
C                                     0289
C *****                                     0290
C *****                                     0291
C                                     0292
      Subroutine MDataAR(Key)                             0293
C                                     0294
C      This subroutine generates the time series of observations based on 0295
C       $Y(t) = ARparm * Y(T-1) + Delta * Z(T) + E(T)$  0296
C                                     0297
C      WHERE      Y(T) = OBSERVATION IN YEAR T.           0298
C                Y(T-1) = OBSERVATION IN YEAR T-1.       0299
C                ARparm = CONSTANT                       0300

```

```

160  continue                                0361
C                                          0362
C  Now generate an autoregressive time series based on the formula. 0363
C                                          0364
      xx(1) = arparm * x1 + delta * dfloat(jpick(1)) + e(1) 0365
      if (jpick(1) .eq. 1) xx(1) = xx(1) + delta            0366
      do 170 i = 2, nyear                                0367
          xx(i) = arparm * xx(i-1) + delta * dfloat(jpick(i)) + e(i) 0368
170  continue                                            0369
C                                          0370
      return                                             0371
      end                                               0372
C                                          0373
C *****                                              0374
C *****                                              0375
C                                          0376
      SUBROUTINE ranprob(d,t,w,q,pd,prt,pw,prq)           0377
C                                          0378
C  Find significance of statistics by randomization.      0379
C                                          0380
      integer nd, nt, nw, nq, key2(10), i, j, k           0381
      double precision pd, prt, pw, prq, dummy, anml      0382
      double precision d, t, w, q, dd, tt, ww, qq         0383
      double precision dx, tx, wx, qx                     0384
      double precision abs, ran3                          0385
C                                          0386
C                                          0387
C                                          0388
C *****                                              0389
C  Common declarations and statement for EPOWER.FOR      0390
C *****                                              0391
C                                          0392
      integer ndata, nmonte, nyear, nkey, iseed,          0393
+      mindif, arflag, LWidth, RWidth, MWidth           0394
      double precision xx(50), delta, slope, arparm      0395
C                                          0396
      common/bigblock/xx, delta, slope, mindif, lwidth, rwidth, 0397
+      ndata, nmonte, nyear, nkey, iseed, arflag, arparm, mwidth 0398
C                                          0399
C *****                                              0400
C                                          0401
      * Initialize the counters for the number of trials with 0402
      larger values of each statistic                    0403
C                                          0404
      ND = 1                                              0405
      NT = 1                                              0406
      NW = 1                                              0407
      NQ = 1                                              0408
C                                          0409
      dd = d                                              0410
      tt = t                                              0411
      ww = w                                              0412
      qq = q                                              0413
C                                          0414
C  *** Run NMonte randomization trials                    0415
C                                          0416
      Do 100 I=1, NMonte                                0417
C                                          0418
C  *** Pick out NKey years at random                      0419
C                                          0420

```

```

do 110 j = 1, nkey                                0421
105   key2(j) = 1 + int(dfloating(nyear) * ran3(iseed)) 0422
C                                                     0423
C   *** If too close to another key year, choose another one 0424
C                                                     0425
C   do 107, k=1,J-1                                0426
C       if (abs(Key2(J)-Key2(K)) .LT. MinDif) goto 105 0427
107   continue                                      0428
110   continue                                      0429
C                                                     0430
C   Call CompD(Key2,DX)                             0431
C   Call CompT(Key2,TX,dummy)                       0432
C   Call CompW(Key2,WX)                             0433
C   Call CompQ(Key2,QX,dummy)                       0434
C   if (DX .GT. DD) ND = ND + 1                     0435
C   if (TX .GT. TT) NT = NT + 1                     0436
C   if (WX .GT. WW) NW = NW + 1                     0437
C   if (QX .GT. QQ) NQ = NQ + 1                     0438
100   continue                                      0439
C                                                     0440
C   *** Compute significance probabilities for all statistics 0441
C                                                     0442
C   ANM1 = 1.0d0 / dfloating(NMonte + 1)             0443
C                                                     0444
C   PD = dfloating(ND) * ANM1                       0445
C   PRT = dfloating(NT) * ANM1                      0446
C   PW = dfloating(NW) * ANM1                       0447
C   PRQ = dfloating(NQ) * ANM1                      0448
C                                                     0449
C   return                                           0450
C   end                                              0451
C                                                     0452
C *****                                           0453
C *****                                           0454
C *****                                           0455
C   Subroutine CompD(Key,DStat)                     0456
C                                                     0457
C   Compute D-statistic. D is the difference between the mean of 0458
C   values in event years and in background years 0459
C                                                     0460
C   Key(i)(Input) The indexes of the nkey key years. 0461
C   DStat (Output) The returned value of the D-Statistic. 0462
C                                                     0463
C   Integer I, J, K, NB, Key(10), bflag(50), kflag(50) 0464
C   Double precision bbar, ebar, DStat               0465
C                                                     0466
C                                                     0467
C                                                     0468
C *****                                           0469
C   Common declarations and statement for EPOWER.FOR 0470
C *****                                           0471
C                                                     0472
C   integer ndata, nmonte, nyear, nkey, iseed,        0473
C   + mindif, arflag, LWidth, RWidth, MWidth         0474
C   double precision xx(50), delta, slope, arparm     0475
C                                                     0476
C   common/bigblock/xx, delta, slope, mindif, lwidth, rwidth, 0477
C   + ndata, nmonte, nyear, nkey, iseed, arflag, arparm, mwidth 0478
C                                                     0479
C *****                                           0480

```

```

C
C      *** Initialize flags to zero
C
C      do 100 i = 1, nyear
C          kflag(i) = 0
C          bflag(i) = 0
100  continue
C
C      *** Compute ebar (mean of key event recruitments)
C
C      ebar = 0.0d0
C      do 110 i = 1, nkey
C          ebar = ebar + xx(key(i))
C          kflag(key(i)) = 1
110  continue
C      ebar = ebar / dfloat(nkey)
C
C      *** Flag background years.  Since epochs may overlap, they are
C      accumulated into a sum only after all are flagged.
C
C      do 130 j = 1, nkey
C          do 220 i = lwidth, rwidth
C              if (i .eq. 0) goto 220
C              * store the index for this year temporarily in k
C              k = key(j) + i
C              if ((k .ge. 1) .and. (k .lt. nyear)) then
C                  if (kflag(k) .eq. 0) bflag(k) = 1
C              endif
220  continue
130  continue
C
C      *** Sum the background years in bbar and add up NB.
C
C      nb = 0
C      bbar = 0.0d0
C      do 140 i = 1, nyear
C          if (bflag(i) .eq. 1) then
C              nb = nb + 1
C              bbar = bbar + xx(i)
C          end if
140  continue
C      bbar = bbar / dfloat(nb)
C
C      dstat = ebar - bbar
C
C      return
C      end
C
C      *****
C      *****
C
C      SUBROUTINE COMPQ(Key,stat,Prob)
C
C      Compute a paired Student's t-statistic and its significance
C      probability.  This is done by preparing two data sets (DATA1
C      and DATA2) of sample size N (= NKey), and then calling
C      subroutine TPTEST from Numerical Recipes.
C
C      stat  (Output) The returned value of the paired t-statistic.
C      Prob  (Output) The returned probability of a greater paired t.

```

```

C
integer i, j, k, n, nb, key(10), kflag(50), jLWidth, jRWidth
double precision data1(50), data2(50), stat, prob
C
C
C
C *****
C Common declarations and statement for EPOWER.FOR
C *****
C
integer ndata, nmonte, nyear, nkey, iseed,
+ mindif, arflag, LWidth, RWidth, MWidth
double precision xx(50), delta, slope, arparm
C
common/bigblock/xx, delta, slope, mindif, lwidth, rwidth,
+ ndata, nmonte, nyear, nkey, iseed, arflag, arparm, mwidth
C
C *****
C
C Initialize sample size and key year status flags.
C
n = nkey
do 100 i = 1, nyear
kflag(i) = 0
100 continue
C
C Copy key year recruitment values into the DATA1 array
C and set all key year status flags
C
do 120 i = 1, nkey
k = key(i)
data1(i) = xx(k)
kflag(k) = 1
120 continue
C
C Find the mean value of background years for each key event.
C
do 200 j = 1, nkey
jLWidth = LWidth
jRWidth = RWidth
data2(j) = 0.0d0
nb = 0
140 do 220 i = jLWidth, jRWidth
C * Don't use the key event year itself.
if (i .eq. 0) goto 220
C * Store the xx index for this year in k
k = key(j) + i
C * If the year is off the end of the series, don't use it.
if ((k .lt. 1) .or. (k .gt. nyear)) goto 220
C * If the year isn't a(nother) key year, add it in.
if (kflag(k) .eq. 0) then
data2(j) = data2(j) + xx(k)
nb = nb + 1
endif
220 continue
C
C * Make sure there are some background years for the test.
C
if (nb .eq. 0) then
C print *, 'No background years for epoch #',j

```

```

C      print *, 'Half-epoch widened by 1.'
C      if (RWidth .ne. 0) then
C          jRWidth = RWidth + 1
C          jLWidth = LWidth - 1
C      else
C          jLWidth = LWidth - 1
C      endif
C      goto 140
C  endif
C
C      * Compute the mean background recruitment for this key event
C
C      data2(j) = data2(j) / DFLOAT(nb)
200 continue
C
C      *** Compute the test statistic and significance probability
C
C      call tptest(data1,data2,n,stat,prob)
C
C      return
C      end
C
C *****
C *****
C
C      SUBROUTINE COMPT(Key,stat,Prob)
C
C      Compute Student's t-statistic and its significance probability.
C      This is done by preparing two data sets (DATA1 and DATA2) and
C      then calling subroutine TTEST from Numerical Recipes.
C
C      stat (Output) The returned value of the t-Statistic.
C      Prob (Output) The returned probability of a greater t.
C
C      Integer I, J, k, N1, N2, Key(10), KFlag(50), BFlag(50)
C      Double precision DATA1(50), DATA2(50), stat, Prob
C
C
C
C *****
C      Common declarations and statement for EPOWER.FOR
C *****
C
C      integer ndata, nmonte, nyear, nkey, iseed,
C      + mindif, arflag, LWidth, RWidth, MWidth
C      double precision xx(50), delta, slope, arparm
C
C      common/bigblock/xx, delta, slope, mindif, lwidth, rwidth,
C      + ndata, nmonte, nyear, nkey, iseed, arflag, arparm, mwidth
C
C *****
C
C      Initialize flags and counters to zero.
C
C      N1 = NKey
C      do 100 i = 1, NYear
C          kflag(i) = 0
C          bflag(i) = 0
100 continue
C

```

```

C      Copy key year recruitment values into the DATA1 array                                0661
C                                                                                               0662
C      do 110 i = 1, NKey                               0663
C          DATA1(i) = xx(key(i))                        0664
C          kflag(key(i)) = 1                             0665
110  continue                                           0666
C                                                                                               0667
C      Flag background years. Since epochs may overlap, they are 0668
C      transferred to DATA2 only after all are flagged.    0669
C                                                                                               0670
C      do 130 j = 1, nkey                                0671
C          do 220 i = lwidth, rwidth                      0672
C              if (i .eq. 0) goto 220                     0673
C              Store the index for this year in k         0674
C              k = key(j) + i                             0675
C              if ((k .ge. 1) .and. (k .lt. nyear)) then 0676
C                  if (kflag(k) .eq. 0) bflag(k) = 1     0677
C              endif                                       0678
220  continue                                           0679
130  continue                                           0680
C                                                                                               0681
C      Transfer the background years to DATA2 and accumulate N2. 0682
C                                                                                               0683
C      N2 = 0                                             0684
C      do 140 i = 1, nyear                               0685
C          if (bflag(i) .eq. 1) then                     0686
C              n2 = n2 + 1                                0687
C              data2(n2) = xx(i)                          0688
C          end if                                         0689
140  continue                                           0690
C                                                                                               0691
C      call ttest(data1,data2,stat,prob,n1,n2)           0692
C                                                                                               0693
C      RETURN                                             0694
C      END                                                0695
C                                                                                               0696
C      *****                                           0697
C      *****                                           0698
C      *****                                           0699
C      Subroutine CompW(Key,W)                            0700
C                                                                                               0701
C      Compute W-statistic, analogous to a paired Student's t-statistic. 0702
C                                                                                               0703
C      W (Output): The returned value of the W-Statistic. 0704
C                                                                                               0705
C      Integer I, I2, I3, K, NDif, Key(10)              0706
C      Double precision B, E, DifMean, DifVar, W         0707
C      Double precision dfloat, sqrt, Dif(100)          0708
C                                                                                               0709
C                                                                                               0710
C      *****                                           0711
C      Common declarations and statement for EPOWER.FOR 0712
C      *****                                           0713
C                                                                                               0714
C      integer ndata, nmonte, nyear, nkey, iseed,        0715
C      + mindif, arflag, LWidth, RWidth, MWidth          0716
C      double precision xx(50), delta, slope, arparm     0717
C                                                                                               0718
C      common/bigblock/xx, delta, slope, mindif, lwidth, rwidth, 0719
C      + ndata, nmonte, nyear, nkey, iseed, arflag, arparm, mwidth 0720

```



```

C 0721
C ***** 0722
C 0723
C *** Compute differences between E and B and accumulate mean 0724
C 0725
DifMean = 0.0d0 0726
NDif = 0 0727
C The first loop goes across the row of the epoch 0728
DO 140 I = LWidth, RWidth 0729
  if (I .EQ. 0) goto 140 0730
  C The inner loop goes down a column of the epoch 0731
  DO 150 I2 = 1, NKey 0732
    C Get the "E" value for this row 0733
    E = XX(Key(I2)) 0734
    C Get the "B" value for this comparison 0735
    C Store the XX index for this B in K 0736
    K = Key(I2) + I 0737
    C Check for out-of-range background years 0738
    IF ((K .LT. 1) .OR. (K .GT. NYear)) THEN 0739
      B = -9999.0d0 0740
    C print *, 'Out of range year & row:', Key(I2)+I, K 0741
    ELSE 0742
      B = XX(K) 0743
    C Check for a background year that's on the list of key 0744
    C years. If it is, then make the data value missing. 0745
    do 110 I3=1,NKey 0746
      IF (Key(I3) .EQ. K) B = -9999.0d0 0747
    110 continue 0748
    end if 0749
    C * Check for missing values ( = -9999) 0750
    if (B .GT. -9000.0d0) then 0751
      NDif = NDif + 1 0752
      Dif(NDif) = E - B 0753
      DifMean = DifMean + Dif(NDif) 0754
    endif 0755
  150 continue 0756
140 continue 0757
DifMean = DifMean / DFloat(NDif) 0758
C 0759
C *** Compute DifVar (Variance of differences) 0760
C 0761
DifVar = 0.0d0 0762
do 160 I = 1, NDif 0763
  DifVar = DifVar + (Dif(I) - DifMean) **2 0764
160 continue 0765
DifVar = DifVar / DFloat(NDif - 1) 0766
C 0767
C *** Compute W-Statistic by dividing mean difference by 0768
C the std. error of the mean difference. See Snedecor 0769
C and Cochran p. 85. 0770
C 0771
W = DifMean / SQRT(DifVar/dfloat(NDif)) 0772
C 0773
RETURN 0774
END 0775
C 0776
C ***** 0777
C ***** 0778
C 0779
SUBROUTINE WrtOut(Key, IData, StatD, StatT, StatW, StatQ, 0780

```

```

+      PD,PRT,PTT,PW,PRQ,PTQ,MSeed)                                0781
C                                                                    0782
C      Write output for epoch power study                          0783
C                                                                    0784
C      integer I, J, IData, Key(10), ifile, MSeed                  0785
C      character*10 filedd                                          0786
C      character*15 atype                                           0787
C      double precision StatD,StatT,StatW,StatQ                    0788
C      double precision PD,PRT,PTT,PW,PRQ,PTQ                      0789
C                                                                    0790
C                                                                    0791
C *****                                                            0792
C      Common declarations and statement for EPOWER.FOR            0793
C *****                                                            0794
C                                                                    0795
C      integer ndata, nmonte, nyear, nkey, iseed,                  0796
+      mindif, arflag, LWidth, RWidth, MWidth                      0797
C      double precision xx(50), delta, slope, arparm               0798
C                                                                    0799
C      common/bigblock/xx, delta, slope, mindif, lwidth, rwidth,   0800
+      ndata, nmonte, nyear, nkey, iseed, arflag, arparm, mwidth   0801
C                                                                    0802
C *****                                                            0803
C                                                                    0804
C      *** The first time through, open files & write headers     0805
C                                                                    0806
C      If (IData .EQ. 1) THEN                                       0807
C                                                                    0808
C      * Open files & write headers                                  0809
C      do 100 ifile = 13, 14                                         0810
C          if (ifile .eq. 13) filedd = 'EPOWER.OUT'                 0811
C          if (ifile .eq. 14) filedd = 'EPOWER.DET'                 0812
C          open (unit=ifile, file=filedd, status='unknown')         0813
C                                                                    0814
C          write (ifile,400)                                          0815
400      format (' Power Analysis of Superposed Epoch Analysis')    0816
C          write (ifile,410)                                          0817
410      format (' -----')                                         0818
C                                                                    0819
C          Compute variables needed for output only                 0820
C                                                                    0821
C          Full width of epoch                                       0822
C          I = RWidth - LWidth + 1                                    0823
C          if (ARFlag .eq. 1) then                                    0824
C              atype = ' Autoregressive'                             0825
C          else                                                       0826
C              atype = '.. Uncorrelated'                             0827
C          end if                                                    0828
C                                                                    0829
C          * Write input data to output file                        0830
C                                                                    0831
C          write (ifile, 510) NData                                   0832
510      format (' Number of simulated data sets ..... ',I5)       0833
C          write (ifile, 515) NYear                                  0834
515      format (' Number of years in each data set ..... ',I5)    0835
C          write (ifile, 520) NKey                                    0836
520      format (' Number of key event years ..... ',I5)           0837
C          write (ifile, 525) I                                       0838
525      format (' Width of epoch..... ',I5)                      0839
C          write (ifile, 526) -LWidth                                0840

```



```

common/bigblock/xx, delta, slope, mindif, lwidth, rwidth,
+   ndata, nmonte, nyear, nkey, iseed, arflag, arparm, mwidth
C
C *****
C
C   * Write Header
C   write (13,500)
500  format (// ' SUMMARY OF RESULTS:')
C   write (13,510)
510  format (' -----')
C
C   *** Compute upper & lower confidence bounds on powers:
C   AN2 = 2.0d0 / dsqrt(dfloat(Ndata))
C
C   SD = AN2 * dsqrt(PSigD*(1.0d0-PSigD))
C   LD = PSigD - SD
C   UD = PSigD + SD
C
C   STT = AN2 * dsqrt(PTSigT*(1.0d0-PTSigT))
C   LTT = PTSigT - STT
C   UTT = PTSigT + STT
C
C   SRT = AN2 * dsqrt(PRSigT*(1.0d0-PRSigT))
C   LRT = PRSigT - SRT
C   URT = PRSigT + SRT
C
C   SW = AN2 * dsqrt(PSigW*(1.0d0-PSigW))
C   LW = PSigW - SW
C   UW = PSigW + SW
C
C   SRQ = AN2 * dsqrt(PRSigQ*(1.0d0-PRSigQ))
C   LRQ = PRSigQ - SRQ
C   URQ = PRSigQ + SRQ
C
C   STQ = AN2 * dsqrt(PTSigQ*(1.0d0-PTSigQ))
C   LTQ = PTSigQ - STQ
C   UTQ = PTSigQ + STQ
C
C   *** Write the statistics
520  format(/ ' Statistic:',t20,'D',t26,'T(r)',t35,'T(t)',t47,'W',
+   t53,'Q(r)',t62,'Q(t)')
600  format (/ ' N Significant:', t15, 6(I6,3X))
610  format (/ ' Est. Power:', t15, 6(F6.3,3X))
620  format (/ ' Lower 95% CL:', t15, 6(F6.3,3X))
630  format ( ' Upper 95% CL:', t15, 6(F6.3,3X))
C   write (13,520)
C   write (13,600) NSigD, NRSigT, NTSigT, NSigW, NRSigQ, NTSigQ
C   write (13,610) PSigD, PRSigT, PTSigT, PSigW, PRSigQ, PTSigQ
C   write (13,620) LD, LRT, LTT, LW, LRQ, LTQ
C   write (13,630) UD, URT, UTT, UW, URQ, UTQ
C
C   write (13,700)
C   write (13,705)
C   write (13,710)
C   write (13,720)
C   write (13,730)
C   write (13,740)
C   write (13,750)
C   write (13,760)
C

```

```

700  format (//' Notes on Statistics:')
705  format (' -----')
710  format (' D is the difference statistic.')
720  format (' T(t) and T(r) are Student''s t-statistics, with')
730  format (' prob. from tables / computed by randomization.')
740  format (' W is the paired t-like statistic of Prager & Hoenig.')
750  format (' Q(t) and Q(r) are paired Student''s t-stats, with')
760  format (' prob. from tables / computed by randomization.')
C
      close(unit=13)
      close(unit=14)
C
      return
      end
C *****
      FUNCTION RAN3(IDUM)
C *****
C      Modified Uniform random deviate generator. (C) 1986 by Numerical
C      Recipes Software. Reproduced by permission, from the book
C      Numerical Recipes: The Art of Scientific Computing, published by
C      Cambridge University Press.
C *****
C      DOUBLE PRECISION fac, ran3
C      INTEGER mbig, mseed, mz, ma(55), iff, mj, iabs, mod, mk
C      INTEGER i, ii, k, inext, inextp, idum
C      PARAMETER (MBIG=1000000000,MSEED=161803398,MZ=0,FAC=1.0d-9)
C      DATA IFF /0/
C
      IF(IDUM .LT. 0 .OR. IFF .EQ. 0) THEN
          IFF = 1
          MJ = MSEED - IABS(IDUM)
          MJ = MOD(MJ,MBIG)
          MA(55) = MJ
          MK = 1
          DO 11 I=1,54
              II=MOD(21*I,55)
              MA(II)=MK
              MK=MJ-MK
              IF(MK.LT.MZ) MK=MK+MBIG
              MJ=MA(II)
11          CONTINUE
          DO 13 K=1,4
              DO 12 I=1,55
                  MA(I)=MA(I)-MA(1+MOD(I+30,55))
                  IF(MA(I).LT.MZ) MA(I)=MA(I)+MBIG
12              CONTINUE
13          CONTINUE
          INEXT=0
          INEXTP=31
          IDUM=1
          ENDIF
          INEXT=INEXT+1
          IF(INEXT.EQ.56) INEXT=1
          INEXTP=INEXTP+1
          IF(INEXTP.EQ.56) INEXTP=1
          MJ=MA(INEXT)-MA(INEXTP)
          IF(MJ.LT.MZ) MJ=MJ+MBIG
          MA(INEXT)=MJ

```

```

RAN3=MJ*FAC
RETURN
END
C
C *****
FUNCTION GASDEV(IDUM)
C
C *****
C Modified Gaussian random deviate generator. (C) 1986 by Numerical
C Recipes Software. Reproduced by permission, from the book
C Numerical Recipes: The Art of Scientific Computing, published by
C Cambridge University Press.
C *****
C
integer iset, idum
double precision fac,gset,gasdev,r,ran3,v1,v2
double precision sqrt, log
C
DATA ISET/0/
IF (ISET .EQ. 0) THEN
1 V1 = 2.0d0 * RAN3(IDUM) - 1.0d0
V2 = 2.0d0 * RAN3(IDUM) - 1.0d0
R = V1**2 + V2**2
IF (R .GE. 1.0d0) GO TO 1
FAC = SQRT(-2.0d0 * LOG(R) / R)
GSET = V1 * FAC
GASDEV = V2 * FAC
ISET = 1
ELSE
GASDEV = GSET
ISET = 0
ENDIF
RETURN
END
C
C *****
C *****
C
SUBROUTINE AVEVAR(DATA,N,AVE,VAR)
C
C *****
C Modified averaged variance subroutine. (C) 1986 by Numerical
C Recipes Software. Reproduced by permission, from the book
C Numerical Recipes: The Art of Scientific Computing, published by
C Cambridge University Press. Called by TTEST.
C *****
C
double precision data(50), s, ave, var
integer j, n
C
ave=0.0
var=0.0
do 11 j=1,n
ave=ave+data(j)
11 continue
ave=ave/n
do 12 j=1,n
s=data(j)-ave
var=var+s*s
12 continue

```

```

var=var/dfloat(n-1)
return
end
C
C *****
C *****
C
SUBROUTINE TTEST(DATA1,DATA2,T,PROB,N1,N2)
C
C *****
C Modified Student's t-test subroutine. (C) 1986 by Numerical
C Recipes Software. Reproduced by permission, from the book
C Numerical Recipes: The Art of Scientific Computing, published by
C Cambridge University Press.
C *****
C
double precision data1(50),data2(50)
double precision ave1, ave2, var, var1, var2, t, prob
double precision betai
integer df, n1, n2
logical xmonte
common/flags/xmonte
C
call avevar(data1,n1,ave1,var1)
call avevar(data2,n2,ave2,var2)
df=n1+n2-2
var=((n1-1)*var1+(n2-1)*var2)/df
t=(ave1-ave2)/sqrt(var*(1.0d0/n1+1.0d0/n2))
C Note: prob. adjusted for one-tailed test.
C Compute only if randomizing
C This is a one-tailed test
if (xmonte) then
if (t .gt. 0.0d0) then
prob=0.5d0*betai(0.5d0*df,0.5d0,df/(df+t**2))
else
prob = 1.0d0
endif
endif
return
end
C
C *****
C *****
C
SUBROUTINE TPTEST(DATA1,DATA2,N,T,PROB)
C
C *****
C Modified paired Student's t-test subroutine. (C) 1986 by
C Numerical Recipes Software. Reproduced by permission, from the
C book Numerical Recipes: The Art of Scientific Computing,
C published by Cambridge University Press.
C *****
C
double precision DATA1(50),DATA2(50)
double precision sqrt, betai, sd
double precision ave1, ave2, var1, var2, cov, t, prob
integer n, j, df
logical xmonte
common/flags/xmonte
C

```

```

CALL AVEVAR(DATA1,N,AVE1,VAR1) 1141
CALL AVEVAR(DATA2,N,AVE2,VAR2) 1142
COV = 0.0d0 1143
DO 11 J = 1, N 1144
    COV = COV + (DATA1(J) - AVE1) * (DATA2(J) - AVE2) 1145
11 CONTINUE 1146
DF=N-1 1147
COV=COV/DF 1148
SD = SQRT((VAR1 + VAR2 - 2.0d0 * COV) / N) 1149
T = (AVE1 - AVE2) / SD 1150
if (xmonte) then 1151
    if (t .gt. 0.0d0) then 1152
        prob=0.5d0*betai(0.5d0*df,0.5d0,df/(df+t**2)) 1153
    else 1154
        prob = 1.0d0 1155
    endif 1156
endif 1157
RETURN 1158
END 1159

C 1160
C ***** 1161
C FUNCTION BETAI(A,B,X) 1162
C 1163
C ***** 1164
C Modified incomplete beta function. (C) 1986 by Numerical Recipes 1165
C Software. Reproduced by permission, from the book Numerical 1166
C Recipes: The Art of Scientific Computing, published by 1167
C Cambridge University Press. 1168
C ***** 1169
C 1170
C double precision a, b, x, bt, betai, betacf, dlog, gammln 1171
C 1172
C if(x .lt. 0.0d0 .or. x .gt. 1.0d0) pause 'bad argument x in betai' 1173
C if(x .eq. 0.0d0 .or. x .eq. 1.0d0 ) then 1174
C     bt = 0.0d0 1175
C else 1176
C     bt = exp(gammln(a+b)-gammln(a)-gammln(b) 1177
C & +a*dlog(x)+b*dlog(1.0d0-x)) 1178
C endif 1179
C if(x.lt.(a+1.0d0)/(a+b+2.0d0))then 1180
C     betai=bt*betacf(a,b,x)/a 1181
C     return 1182
C else 1183
C     betai=1.0d0-bt*betacf(b,a,1.0d0-x)/b 1184
C     return 1185
C endif 1186
C end 1187
C 1188
C ***** 1189
C FUNCTION BETACF(A,B,X) 1190
C 1191
C ***** 1192
C Modified continued fraction for incomplete beta function. 1193
C (C) 1986 by Numerical Recipes Software. Reproduced by 1194
C permission, from the book Numerical Recipes: The Art of 1195
C Scientific Computing, published by Cambridge University Press. 1196
C ***** 1197
C 1198
C double precision am, bm, az, qab, qap, gam, bz, ap, bp, app, bpp 1199
C double precision a, b, x, em, tem, betacf, eps, d, aold 1200

```



```

integer m, itmax
parameter (itmax=100,eps=3.0d-7)
C
am=1.0d0
bm=1.0d0
az=1.0d0
qab=a+b
qap=a+1.0d0
gam=a-1.0d0
bz=1.0d0-qab*x/qap
do 11 m=1,itmax
    em=m
    tem=em+em
    d=em*(b-m)*x/((gam+tem)*(a+tem))
    ap=az+d*am
    bp=bz+d*bm
    d=-(a+em)*(qab+em)*x/((a+tem)*(qap+tem))
    app=ap+d*az
    bpp=bp+d*bz
    aold=az
    am=ap/bpp
    bm=bp/bpp
    az=app/bpp
    bz=1.
    if (abs(az - aold) .lt. eps * abs(az)) go to 1
11  continue
    pause 'a or b too big, or itmax too small'
1   betacf=az
    return
end
C
C*****
FUNCTION GAMMLN(XX)
C
C*****
C   Modified log gamma function. (C) 1986 by Numerical Recipes
C   Software. Reproduced by permission, from the book Numerical
C   Recipes: The Art of Scientific Computing, published by
C   Cambridge University Press.
C*****
C   double precision COF(6),STP,HALF,ONE,FPF,X,TMP,SER,xx, gammln
C   integer j
C
C   data cof,stp/76.18009173d0,-86.50532033d0,24.01409822d0,
*   -1.231739516d0,.120858003d-2,-.536382d-5,2.50662827465d0/
C
C   data half,one,fpf/0.5d0,1.0d0,5.5d0/
C
C   x = xx-one
C   tmp = x+fpf
C   tmp=(x+half)*log(tmp)-tmp
C   ser=one
C   do 11 j=1,6
C       x=x+one
C       ser=ser+cof(j)/x
11  continue
C   gammln=tmp+log(stp*ser)
C   return
C   end

```

C*****	1261
C *****	1262
C End of EPOWER.FOR	1263
C *****	1264