Woods Hole
Oceanographic
Institution

WHOI-80-9

Copy 1

CALCULATION OF MEAN SEASONAL SEA
SURFACE ELEVATION FIELDS OVER THE NORTH
ATLANTIC CONTINENTAL SHELF AND SLOPE
USING HISTORICAL HYDROSTATION DATA

by

James H. Churchill

February 1980

TECHNICAL REPORT

*Prepared for the Department of Energy under
Grant DE-AC02-79-EV10005.*

WOODS HOLE, MASSACHUSETTS 02543

WHOI-80-9

CALCULATION OF MEAN SEASONAL SEA
SURFACE ELEVATION FIELDS OVER THE NORTH
ATLANTIC CONTINENTAL SHELF AND SLOPE
USING HISTORICAL HYDROSTATION DATA

by

James H. Churchill

WOODS HOLE OCEANOGRAPHIC INSTITUTION
Woods Hole, Massachusetts 02543

February 1980

TECHNICAL REPORT

Prepared for the Department of Energy under Grant DE-AC02-79-10005.

Reproduction in whole or in part is permitted for any purpose of the United States Government. This report should be cited as: Woods Hole Oceanographic Institution Technical Report WHOI-80-9.

Approved for Distribution

Valentine Worthington, Chairman
Department of Physical Oceanography

CONTENTS

	<u>Page</u>
Abstract	1
1. Introduction	2
2. Theoretical Background	2
3. Data Base	4
4. Selection of Seasonal Periods	5
5. Data Analysis	5
a. Initial Data Reduction	6
b. Averaging Sigma-t's on a Geographic Grid	7
c. Calculation of the Surface Elevation Field	9
6. Results	10
7. Validity of Specifying ζ_O and ζ_H as Functions of Bottom Depth	13
Acknowledgements	15
References	16
Table 1	17
Figures	18-48
Appendix I Revision of NODC Data Processing Program OCCOMP	49
Appendix II Software	53
OCCOMP	53
PICKOUT	73
GRID	75
STD	78
INTG	80
TYGRID	82
GRPRS	85
TYGRS	88

Abstract

This report details the calculation and results of a simple coastal elevation field model which was presented in a recent paper (Csanady, 1979). Seasonal mean sea surface elevation fields were calculated over the shelf-slope region off the east coast of North America north of Cape Hatteras by applying the model to a grid of averaged sigma-t's at standard depths. The sigma-t grid was calculated utilizing a data base of approximately 30,000 hydrographic stations from NODC archives. Difficulties involved with the numerical implementation of the model and economizing adaptations to the WHOI NODC data processing program, OCCOMP, are discussed.

1. Introduction

In a recent paper (Csanady, 1979, hereafter referred to as C79) a simple theoretical model was given for the approximate calculation of the density related pressure field over a shelf-slope region. The model was applied to the region off the east coast of North America north of Cape Hatteras. The details of how the calculation was actually carried out were not given in the paper and are described in this report.

2. Theoretical Background

Csanady bases his derivation in C79 on the depth integrated equations of motion and continuity written in the following form:

$$\begin{aligned} -\frac{fV}{gH} &= -\frac{\partial \zeta}{\partial x} - \int_{-H}^0 (1+z/H) \frac{\partial \epsilon}{\partial x} dz + \frac{\tau_{xs} - \tau_{xb}}{\rho_0 g H} \\ \frac{fU}{gH} &= -\frac{\partial \zeta}{\partial y} - \int_{-H}^0 (1+z/H) \frac{\partial \epsilon}{\partial y} dz + \frac{\tau_{ys} - \tau_{yb}}{\rho_0 g H} \end{aligned} \quad (1)$$

$$\frac{\partial U}{\partial x} + \frac{\partial V}{\partial y} = 0$$

where:

- U and V are depth integrated velocity components
- f is the Coriolis parameter
- g is the gravitational acceleration
- H is the bottom depth
- $\tau_{ys}, \tau_{xs}, \tau_{xb}, \tau_{yb}$ are the surface and bottom stress components respectively

ζ is the vertical displacement from the equilibrium position of the sea surface ($z = 0$)
 ϵ is the density excess.

The density at any level is given by

$$\rho = \rho_0 (1 + \epsilon(x, y, z)) \quad (2)$$

where ρ_0 is the density of the deepest water, the abyssal density.

For this application surface stress is not considered and the bottom stress is parameterized according to two schemes. In scheme 1 the stress is considered proportional to the bottom geostrophic velocity. In scheme 2 the stress is proportional to the depth averaged velocity.

A local coordinate system is used with y orientated parallel to the isobaths, x perpendicular to the isobaths, and z positive downward. For this system the x component of bottom stress is considered zero. The idealization of zero long-isobath density variation is adopted, and the boundary conditions of zero transport at the coast and zero elevation field at the slope boundary ($\zeta = 0$ at the outer edge) are applied.

For stress scheme 1 the following equation describing the coastal elevation field results:

$$\zeta = \zeta_d + \zeta_b \quad (3)$$

where:

$$\zeta_d = - \int_{-H}^0 \epsilon dz$$

$$\zeta_b = - \int_H^{H_m} \epsilon_b dH$$

ϵ_b is the bottom density excess, a function of depth alone since the long-isobath density variation is considered zero.

H_m is the maximum depth of the region considered.

For stress scheme 2 the resulting equation is

$$\zeta = \zeta_d - \zeta_g + \zeta_H \quad (4)$$

where

$$\zeta_g = \int_{-H}^0 \epsilon \frac{z}{H} dz$$

$$\zeta_H = \int_H^{H_m} \zeta_g / H dH$$

The method and approximations required for computing the above elevations numerically will be discussed.

3. Data Base

The data base consists of temperature, salinity and depth station data (binary format) from NODC archives for the area outlined in Figure 1. This area encompasses the continental shelf and slope, from the shore to approximately

4,000 m depth. All historical data up to and including 1977, approximately 30,000 stations in all, have been included. The data has strong seasonal and regional variability as indicated by the seasonal station plots of Figures 1-4.

4. Selection of Seasonal Periods

The data density was insufficient to permit a useful calculation of the monthly mean elevation field, necessitating seasonal mean calculations. The surface temperature information presented in Table 1 was used to select the time periods of the seasons. At all locations listed there are two periods of relative temperature stability (Jan.-March and July-Sept.) and two periods of transition (Apr.-June and Oct.-Dec.). These periods were chosen as seasons and will hereafter be referred to as follows:

Winter	Jan.-March
Spring	April-June
Summer	July-Sept.
Fall	Oct.-Dec.

5. Data Analysis

The method for computing the elevation field from the NODC data involves calculating sigma-t's at standard levels and averaging these sigma-t's within elements of a geographic grid. For each grid element, the integrals ζ_d and ζ_g of eqns. (3) and (4) are computed by trapezoid integration. ζ_H and ζ_b are specified as piecewise linear functions of depth whose values are obtained by integration of ϵ_b and ζ_g along a representative transect from the from the deepest water to the shore.

A detailed step by step description of this method follows:

5a. Initial Data Reduction

The first data processing step is to read the NODC tapes, calculate the sigma-t's at standard depths and sort the data into seasonal and regional groups.

W.H.O.I. maintains a program, OCCOMP, which processes NODC data. For this specific application OCCOMP proved too costly and cumbersome to be feasible (the estimated cost of processing all 30,000 stations was over \$10,000). It was therefore modified extensively, as described in Appendix I, to suit this particular project. The basic features of the modified version of OCCOMP are described below.

— The data is read and processed from the input tape station by station. The salinity and temperature at each observation depth are checked for data quality. If a station contains less than 3 good observations it is disregarded. The salinity and temperature are interpolated to standard levels. The standard levels are (in meters): 1,10,20,30,40,50,60,75,100,125,150,200,250,300,400,500,600,700,800,900,1000,1200,1500,1750,2000,2500,3000,4000. The sigma-t and specific volume anomaly are calculated at each standard level.

— For each station two binary records are output to tape or disk. The first record contains general station information. The second has the values of salinity, temperature, sigma-t and specific volume anomaly at the standard depths.

— The processed data was output to 4 tapes, 1 per season. The tapes were divided into files according to Marsden Square.

— The sigma-t's were extracted from these tapes and written onto disk files by program PICKOUT. This program rejects sigma-t's which have associated temperature of greater than 40°C or less than 2°C, or which have associated salinity of greater than 39‰ or less than 26‰. The sigma-t's which pass are written to a binary disk file. A printout is generated which lists, for each station, the number of sigma-t's recorded and the number rejected. Typically less than 0.02% of the sigma-t's were rejected.

5b. Averaging Sigma-t's on a Geographic Grid

For each season the binary sigma-t files were combined into two regional files whose geographic bounds are shown in Figure 1. Region 1 extends from Cape Hatteras to Nova Scotia, encompassing Georges Bank; Region 2 stretches north to Newfoundland, including the Grand Banks. In each region the sigma-t's were averaged within elements of a geographic grid by program GRID.

The grid of each region is specified by a card-imaged file containing the center locations and dimensions of each grid element together with the bottom depth, generally taken as the depth at the element's center. Program GRID reads the regional sigma-t file station by station and calculates the average sigma-t and standard deviation at each standard level of each grid element. The data file is then reread and the sigma-t's are filtered based on the deviations from the mean.

The filtering procedure is similar to one employed by Levitus and Oort (1977). For each sigma-t the affected mean and standard deviation are calculated with that sigma-t absent. For convenience let $\langle \sigma_t \rangle$, STD and $\langle \sigma_t \rangle'$, STD' represent the mean and standard deviation with and without the sigma-t. If the sigma-t is greater than 5 STD' from

$\langle \sigma_t \rangle'$, then STD' and $\langle \sigma_t \rangle'$ replace $\langle \sigma_t \rangle$ and STD . Filtering is not performed when the standard deviation is less than .02 or when the number of observations is less than 4.

Density inversions were occasionally encountered in the averaged sigma-t field. Various techniques, including smoothing methods were experimented with to make the sigma-t's monotonically increase with depth. In the end it was decided that the technique which had the least effect on the uninverted portion of the field should be used. This is simply to raise the value of the inverted sigma-t to that of the standard depth immediately above. All sigma-t's changed are flagged as negative numbers.

Program GRID optionally calculates ζ_d , ζ_g and ϵ_b . The values obtained are not the final values used as will be discussed in the next section.

For each grid element a printout is generated of the average sigma-t, standard deviation, and number of observations at each level. The magnitude of any density inversion is also printed.

A binary file is created which contains the grid's dimensions and depths, the average sigma-t's, standard deviations and numbers of observations of each element, as well as ζ_d , ζ_g and ϵ_b if calculated.

Program TYGRID reads this file and prints out sigma-t's, standard deviations, or number of observations at any standard depth desired in the form of the geographic grid.

5c. Calculation of the Surface Elevation Field

For some grid elements the contributions to the surface elevation fields, ζ_d and ζ_g of (3) and (4), cannot be calculated from the averaged sigma-t field because the bottom depth exceeds the depth of the deepest available data. In such cases the missing data is filled in with values representative of the particular region for that season. For deep water this is a justifiable procedure because the horizontal density variation is small, as will be shown.

The integrals ζ_d and ζ_g are calculated by trapezoid integration with the value of abyssal density, ρ_0 , always 1.028 (other integration schemes were tried but all gave insignificantly differing results).

ζ_H and ζ_b are specified as piecewise linear functions of depth, different functions for each region and season. These functions are obtained using a transect of grid elements which cuts perpendicular to the isobaths at approximately the center of the particular region. Starting with the grid element of greatest bottom depth ζ_b and ζ_H are computed by trapezoid integration as follows:

$$\zeta_{bn} = \sum_{i=2}^n (H_i - H_{i-1}) (\epsilon_{b_i} + \epsilon_{b_{i-1}}) / 2 \quad (5)$$

$$\zeta_{Hn} = \sum_{i=2}^n (H_i - H_{i-1}) (\zeta_{g_i} + \zeta_{g_{i-1}}) / 2 \quad (6)$$

where:

ζ_{bn} and ζ_{Hn} are the values of ζ_b and ζ_H at bottom

depth H_n ; H_i , $i = 1, 2, 3, \dots$ are the bottom depths in decreasing order, $H_1 = \text{maximum bottom depth} = H_m$; ϵ_{bi} and ζ_g are the associated values of bottom density excess and ζ_g .

In practice a number of elements close to the transect are added to ensure a sufficient number of values for meaningful integration. The need for and the validity of using this method to calculate ζ_b and ζ_H will be discussed in a later section.

Program GRPRS calculates the seasonal surface elevation fields. This program reads the binary file of averaged sigma-t's created by program GRID. The library of deepwater densities and the values of ζ_b and ζ_H at specified depths are input from cards. The program prints out the surface elevations according to eqns. (3) and (4) together with ζ_d , ζ_b , ζ_g and ζ_H for each grid element. The above values are also printed in the form of the geographic grid.

6. Results

The grid networks used to represent Regions 1 and 2 both consist of overlapping 1 degree squares separated by 1/2 degree. A 1/2 degree square grid was tried but the results were excessively erratic.

Grid depths of Region 1 are shown in Figure 5. The elements enclosed in circles are those used for calculating ζ_b and ζ_H of equations (5) and (6).

The sigma-t field at 10 m during summer is presented in Figure 6. The sigma-t's are written as the differences from 20, times 100. 450, for example, represents a sigma-t of 24.50. The standard deviations times 100 and the number of observations per element are displayed in Figures 7 and 8.

Note that the standard deviations are approximately 1 in the area between Cape Hatteras and Long Island and about $1/2$ over the rest of the region. This pattern is characteristic of all seasons.

Contoured sigma-t fields, drawn from the digital printouts, of all seasons at 10 m are shown in Figures 9-12. These seasonal patterns are similar; all show lower sigma-t's in the region of Cape Hatteras and the northern Nova Scotian shelf and higher sigma-t's in the area of Georges Bank and the Gulf of Maine.

A printout of the sigma-t field at 3000 m during summer is shown in Figure 13. Note that the horizontal density variation at this depth is very slight. This is characteristic of all seasons.

The sea surface elevation field of summer according to bottom stress parameterization scheme 1 (equation 3) is shown in Figure 14 (See also C79, Fig.7). A few questionable points (due to scarcity of reliable observations) have been deleted. The main feature is a level change of about 10 cm approximately coincident with the 1000 m isobath. As indicated in C79 this would produce a geostrophic surface current to the southwest of magnitude 20 cm/sec, flowing parallel to the isobaths and of width in the order of 100 km. There is also a definite high associated with the Gulf of Maine. The scattered high values at the eastern edge of the region are presumably due to Gulf Stream influence.

Figure 15 show the sea surface elevation field according to bottom stress parameterization scheme 2. The predominant features are the same as in Figure 13 except that the level differences are diminished in magnitude by about $1/2$. Since these two schemes yield such similar results further discussion will be limited to results using scheme 1, where the signal to noise ratio is better.

Sea surface elevations for the remaining seasons are displayed in Figures 16-18. All seasonal fields possess the same basic pattern. The level change at the shelf break appears slightly higher in the winter and spring than in the autumn.

The bottom depths of grid elements modeling Region 2 are shown in Figure 19. The circled elements are again those used in determining ζ_b and ζ_H . The standard deviations times 100 and the number of observations at 10 m during spring are shown in Figures 20 and 21.

Contoured sigma-t fields at 10 m for spring and summer are displayed in Figures 22-23. (Autumn and winter were not processed due to very low density of observations). For both seasons there is a density minimum associated with the St. Lawrence outflow and the density steadily increases in the eastward direction.

Figures 24-25 present the sea surface elevation fields for spring and summer. Both seasonal fields display a high in the area of the St. Lawrence outflow with the elevation steadily decreasing in the eastward direction over the Grand Banks region. As in the seasonal fields of Region 1 there is a well defined level change nearly coincident with the 1000 m isobath.

The implications of these elevation fields are discussed in detail in C79. The basic conclusion is that the mean pressure field is partly caused by fresh water influx (most noticeably from the St. Lawrence), but the major contribution is from the deep ocean.

7. Validity of Specifying ζ_b and ζ_H as Functions of Bottom Depth

For one of the early attempts at calculating the elevation fields of equations (3) and (4) the integrals ζ_b and ζ_H were computed for each transect of grid elements which formed nearly a right angle with the isobaths. Unfortunately, few transects had sufficient values for meaningful integration. This combined with the scatter of ϵ_b and ζ_g caused the resulting elevation fields to be excessively erratic, prohibiting useful interpretation. The approach of calculating ζ_b and ζ_H from ϵ_b and ζ_g values in the vicinity of one transect, conveniently chosen at the approximate center of the particular region, and specifying these as functions of bottom depth was finally decided upon. To what extent this approach is valid may be discerned by examining the ϵ_b as a function of bottom depth.

ϵ_b vs bottom depth for Region 1 in summer is displayed in Figure 26. Points marked with an X are those used in the ζ_b calculation (eqn.(5)). The scatter is seen more clearly in Figure 27 which presents only the top 400 m. A similar graph for spring is presented in Figure 28. As one can see the scatter is nontrivial.

One can qualitatively determine from the graphs of both seasons that a polynomial curve fit by least squares to the ϵ_b values would generally lie above the points chosen for the integration. These curves would, however, have a positive second derivative in the direction of integration. For such curves trapezoid integration computes a high integral value. Graphs for winter and fall have characteristics similar to the spring and summer graphs.

ϵ_b vs bottom depth for Region 2 during spring is shown in Figures 29 and 30. Values from the two transects used for the ζ_b calculation are indicated by '+'s and 'x's (see Figure 14). The points marked by Δ 's are a continuation of the transect denoted by '+'s. Together the '+'s and Δ 's demonstrate a hysteresis which occurs progressing shoreward along the transect from deep water to shallow water and back into deeper water.

Figure 31 shows a similar pattern during summer.

One can observe that each region has a markedly different dependence of ϵ_b with depth. ϵ_b vs depth graphs of various subregions of Regions 1 and 2 were produced in an attempt to determine if the scatter could be reduced by using smaller regions. Within each region the scatter, and thus the ϵ_b vs depth dependence, appeared to be fairly uniform, indicating that our rather arbitrary choice of region boundaries was good.

Acknowledgements

I would like to express deep appreciation to Dr. G.T. Csanady who developed the concept and theory of this project. My thanks also go to Mr. George Heimerdinger who was very helpful in obtaining NODC data.

References

Csanady, G.T., The pressure field along the western margin of the North Atlantic, J. Geophys. Res., 84, 4905-4915, 1979.

Levitus, Sydney and Abraham Oort, Global Analysis of Oceanographic Data, Bulletin of the American Meteorological Society, 84, 1270-1284, 1977.

Oceanographic Atlas of the North Atlantic Ocean,
Section II: Physical Properties, Pub.#700, U.S.
Naval Oceanographic Office, 1967.

Table 1

Monthly Estimates of Sea Surface Temperatures
on the North Atlantic Continental Shelf

Month	Surface Temperature - °F				
	CH	CM	MV	NS	AP
Jan	60	42	40	34	32
Feb	60	20	40	32	30
Mar	62	42	40	32	30
Apr	66	46	42	34	33
May	72	55	48	42	36
Jun	76	65	56	48	42
Jul	80	72	64	56	50
Aug	80	74	66	60	54
Sep	78	70	66	60	54
Oct	74	63	57	52	48
Nov	68	55	50	45	42
Dec	66	47	44	38	35

Abbreviations of Nearest Landmarks:

CH - Cape Hatteras

NS - Nova Scotia

CM - Cape May

AP - Avalon Peninsula (off
Newfoundland)

MV - Martha's Vineyard

Above figures estimate from profile maps II 1 — II 12 in
Oceanographic Atlas of the North Atlantic Ocean, Section II:
Physical Properties, Pub.#700, U.S. Naval Oceanographic Office,
1967.

Figure 1. NODC hydrostation locations during Jan.-March.

Figure 2. NODC hydrostation locations during April-June.

Figure 3. NODC hydrostation locations during July-Sept.

Figure 4. NODC hydrostation locations during Oct.-Dec.

Figure 5. (Bottom depth)/10 of the grid of Region 1. Actual 100 m and 1000 m isobaths included. Circled grid elements used in ζ_p calculation, eqn. (5).

Figure 6. (Sigma-t-20) x 100 at 10 m during summer (July-Sept.).

Figure 7. (Standard deviation) x 100 at 10 m during summer.

Figure 8. Number of observations per grid element at 10 m during summer.

Figure 9. Sigma-t field at 10 m during winter (Jan.-March).

Figure 10. Sigma-t field at 10 m during spring (April-June).

Figure 11. Sigma-t field at 10 m during summer (July-Sept.).

Figure 12. Sigma-t field at 10 m during autumn (Oct.-Dec.).

Figure 13. (Sigma-t-20) x 100 at depth 3000 m during summer. # indicates no data available.

Figure 14. Sea surface elevation (cm) calculated by equation (3) during summer.

Figure 15. Sea surface elevation (cm) calculated by equation (4) during summer.

Figure 16. Sea surface elevation (cm) during winter.

Figure 17. Sea surface elevation (cm) during spring.

Figure 18. Sea surface elevation (cm) during autumn.

Figure 19. (Bottom depth)/10 of the grid of Region 2.
Circled grid elements used in ζ_b calculation.

Figure 20. (Standard deviation) x 100 at 10 m during spring.

Figure 21. Number of observations per grid element at 10 m during spring.

Figure 22. Sigma-t field at 10 m during spring.

Figure 23. Sigma-t field at 10 m during summer.

Figure 25. Sea surface elevation (cm) during summer.

BOTTOM ETA VS DEPTH; SUMMER GEORGES BANK

Figure 26. ϵ_p vs bottom depth for Region 1 during summer. Values marked as x are from the circled elements of Figure 5 and were used in the ζ_p determination.

BOTTOM ETA VS DEPTH; SUMMER GEORGES BANK

Figure 27. ϵ_b vs bottom depth for Region 1 during summer. Top 400 m only.

BOTTOM ETA VS DEPTH; SPRING GEORGES BANK

Figure 28. ϵ_b vs bottom depth for Region 1 during spring. Top 400 m only.

BOTTOM ETA VS DEPTH; SPRING GRAND BANKS

Figure 29. ϵ vs bottom depth for Region 2 during spring. Values marked as x, + and Δ are from the grid elements marked in Figure 19.

BOTTOM ETA VS DEPTH; SPRING GRAND BANKS

Figure 30. ϵ_b vs bottom depth for Region 2 during spring. Top 800 m only.

BOTTOM ETA VS DEPTH: SUMMER GRAND BANKS

Figure 31. ϵ_b vs bottom depth for Region 2 during summer. Top 800 m only.

Appendix I

Revision of NODC Data Processing Program OCCOMP

OCCOMP, as written by the Information Processing Center (IPC) of W.H.O.I., reads and processes NODC data from either the new (binary) or old (card imaged) formats. Execution of the program in its original form proved too costly to be feasible for our particular application. For instance, processing the data from Marsden Square 1209 for months January through March cost approximately \$420. These 800 stations represented roughly 1/30 of the total data which required processing.

OCCOMP was modified, as will be described, to significantly reduce our processing cost. The changes, however, were directed to specific needs of this project and do not necessarily represent an improvement for all applications of the program.

Before dealing with the changes a general description of the original version of OCCOMP will be instructive.

The main program reads input options, positions input tapes and prints the header page. If interpolation to standard levels is desired these levels are read from cards. For each station a loop which calls the various processing subroutines is executed, as described below.

The main program calls subroutine STATIN which initializes indicators. STATIN calls either NEWINPUT or OLDINPUT depending on the form of the data (OLDINPUT will be ignored in this discussion). NEWINPUT inputs a binary record into buffer. If it is the first record for the particular station (indicated by a value of 1 or 0 for the first character) subroutine NEWMASTER is called. NEWMASTER decodes station information (location, date and time of sampling, wind speed, etc.) and returns to NEWINPUT. Subroutine NEWDETAIL is called from NEWINPUT once for each sampling depth. NEWDETAIL checks which data is recorded (temperature, salinity, O_2 , NO_2 , etc.) and

calls decoding entries for each one present. Quality indicators of depth, temperature and salinity are checked. If one of these indicates a bad datum (value of greater than 6) the data from that depth is rejected. After decoding data from all depths NEWINPUT checks to determine if there is an additional record for the station. If so, it continues to decode information for the station until all records have been processed, at which point it returns to STATIN which returns to the main program.

If less than 3 sampling depths of data have been accepted, the station is discarded and the loop begins again for the next station. Otherwise, the main program calls subroutine USELECT, a user supplied subroutine which establishes criteria for station selection. If USELECT determines the station is unwanted, then the computation loop is restarted for the next station.

If the station is wanted BASICOMP is called. BASICOMP calculates pressure, Brunt-Väisälä frequency, sound velocity, sigma-t, potential temperature, sigma-theta, specific volume anomaly, and dynamic depth from the temperature, salinity and depth data. These are output to the printer and stored in arrays for future writing on tape or disk.

BASICOMP returns to the main program which calls, at the user's request, subroutine STDINTRP. This subroutine interpolates temperature, potential temperature, salinity, sigma-theta and specific volume anomaly to standard depths. These values are placed in an array for future tape or disk storage. Subroutine VOLTRANS, which calculates geostrophic transport, can also be called.

The last subroutine to be called is SAVEARRAY which outputs three binary records per station to tape or disk. The first record contains master station information (as decoded by NEWMASTER); the second contains all input data and calculated quantities; and the third has all interpolated values.

In the revised version of OCCOMP only salinity, temperature and depth data are read and checked, all other data are ignored.

Unnecessary computations have been deleted. Certain subroutines have been merged and those dealing with data in the old format have been deleted.

The most important alteration is a change of the location at which the station is either selected or rejected by subroutine USELECT. In the new version USELECT is called from NEWINPUT immediately after the call to NEWMASTER. Rejecting a station at this point eliminates considerable computation time devoted to reading and checking unwanted data. This change can represent a significant savings for our application where as many as 9/10 of the stations may be rejected.

The following describes specifically all major changes made in OCCOMP.

Subroutine STATIN, whose main task is to call NEWINPUT and do some indexing, has been merged with NEWINPUT.

In the original version of OCCOMP the main program has a counter which counts all stations that are accepted. When this equals a user specified number the run is terminated. Unfortunately, prior to execution the number of stations accepted cannot be determined. For this reason the revised program counts the number of station encountered instead.

An option was added which allows the user to delete print-out of data and calculated values. This option is executed using variable IPROP which is entered in the first column of the second option card. If IPROP = 0 printout is deleted.

As previously mentioned, the revised version calls USELECT from subroutine NEWINPUT immediately after the call to NEWMASTER. If the station is wanted processing continues. If it's unwanted, a check is made to determine if the station has another record. If so, that record is skipped using subroutine SKPREC (WHOI account 3). Control is then returned to the main program which restarts the computation loop.

Unfortunately there is no way of determining how many additional records a rejected station may have. To avoid mistaking a continuation record of a rejected station with the first record

of a subsequent station a check is made on the first record of each station to determine if it's a continuation record. If so, that record is rejected and control returns to the main program which restarts the computation loop.

STDINTRP has taken over the computation chores of BASICOMP and BASICOMP has been deleted. The modified STDINTRP operates in the following manner.

Temperature and salinity are interpolated to standard depths. Pressure, sigma-t, specific volume and specific volume anomaly are calculated from these interpolated values. These are printed if IPROP $\neq$ 0. The depths, temperatures, salinities, sigma-t's (not sigma-theta as in the original OCCOMP), and specific volume anomalies are put into arrays for future tape or disk output. This procedure differs from that employed by the original version of OCCOMP which calculates all quantities at the observed depths and then interpolates.

Subroutine NEWDETAIL has been changed to decode only depth, temperature and salinity. Quantities not required in this application (O_2 , phosphate, ζ_i , NO_2 , NO_3 and pH) are ignored.

Only two records per station are output to tape or disk. The first contains master station information and the second has the interpolated values.

A number of very large arrays (over 6000 words in all) have been deleted.

Processing the same data base with the original and the modified versions of OCCOMP yielded nearly identical results. The differences were due to the different methods of interpolating to standard depths and never exceeded 2%.

These modifications saved a considerable amount of computer time for our application. The modified program processed the winter data from MARSDEN Square 1209 at a cost of roughly \$25 compared to \$420 required by the original OCCOMP.

Listing of the modified program follows.

Appendix II Software

PCL
COPY OCCOMP TO LP

C	PROGRAM OCCOMP DEC 22 1978	JCOC0001
C		JCOC0002
C	THIS PROGRAM IS A REVISED OF OCCOMP VERSION 6.0 WRITTEN	JCOC0003
C	BY MARY HUNT. IT DEALS WITH ONLY NODC DATA IN BINARY FORMAT	JCOC0004
C	ONLY TEMP, SAL AND PRESSURE DATA IS DECODED, SIGMA-T'S AND	JCOC0005
C	SPECIFIC VOLUME ANOMALIES ARE CALCULATED AT STANDARD DEPTHS	JCOC0006
C	FROM INTERPOLATED VALUES OF TEMP AND SAL.	JCOC0007
C		JCOC0008
C		JCOC0009
C	*****	JCOC0010
C	* * *	JCOC0011
C	* COMMON /LOGUNIT/ *	JCOC0012
C	* * *	JCOC0013
C	*****	JCOC0014
C		JCOC0015
C	THIS BLOCK CONTAINS LOGICAL UNIT NUMBERS	JCOC0016
C	OF ALL DEVICES AND FILES USED	JCOC0017
C	PRINTER, CARD READER, INPUT DEVICE, AND	JCOC0018
C	FILE FOR SAVING STUFF	JCOC0019
C		JCOC0020
C	THIS BLOCK IS USED BY ALL ROUTINES	JCOC0021
C		JCOC0022
C	COMMON/LOGUNIT/LP,	JCOC0023
C	* ICR,	JCOC0024
C	* INUNIT,	JCOC0025
C	* SAVFIL	JCOC0026
C		JCOC0027
C	INTEGER LP/108/	JCOC0028
C	INTEGER ICR/105/	JCOC0029
C	INTEGER INUNIT/8/	JCOC0030
C	INTEGER SAVFIL/1/	JCOC0031
C		JCOC0032
C		JCOC0033
C	*****	JCOC0034
C	* * *	JCOC0035
C	* COMMON /OPTIONS/ *	JCOC0036
C	* * *	JCOC0037
C	*****	JCOC0038
C		JCOC0039
C	THIS BLOCK CONTAINS INPUT OPTIONS AND	JCOC0040
C	PROCESSING FLAGS	JCOC0041
C		JCOC0042
C	THIS BLOCK IS USED BY	JCOC0043
C	NEWINPUT	JCOC0044
C	OLDINPUT	JCOC0045
C	BASICOMP	JCOC0046
C	STDINTRP	JCOC0047
C	VOLTRANS	JCOC0048
C		JCOC0049
C	COMMON/OPTIONS/ COMPOPT, INPOPT, SAVOPT,	JCOC0050
C	* BLKFACT, ENDATA	JCOC0051
C	COMMON/OPTIONS/ NUMSTAT, MAXOBS	JCOC0052
C	INTEGER COMPOPT, INPOPT, SAVOPT, BLKFACT	JCOC0053
C	LOGICAL ENDATA	JCOC0054
C	INTEGER NUMSTAT	JCOC0055
C	INTEGER MAXOBS/140/	JCOC0056

JCOC 0057
JCNC 0058
JCNC 0059
JCOC 0060
JCOC 0061
JCOC 0062
JCNC 0063
JCNC 0064
JCNC 0065
JCOC 0066
JCOC 0067
JCNC 0068
JCNC 0069
JCNC 0070
JCNC 0071
JCNC 0072
JCOC 0073
JCOC 0074
JCOC 0075
JCNC 0076
JCNC 0077
JCOC 0078
JCNC 0079
JCNC 0080
JCNC 0081
JCNC 0082
JCNC 0083
JCNC 0084
JCOC 0085
JCNC 0086
JCNC 0087
JCOC 0088
JCNC 0089
JCOC 0090
JCOC 0091
JCOC 0092
JCOC 0093
JCOC 0094
JCOC 0095
JCNC 0096
JCNC 0097
JCOC 0098
JCNC 0099
JCOC 0100
JCOC 0101
JCOC 0102
JCNC 0103
JCOC 0104
JCOC 0105
JCOC 0106
JCNC 0107
JCNC 0108
JCOC 0109
JCNC 0110
JCOC 0111
JCOC 0112
JCOC 0113
JCOC 0114
JCOC 0115
JCOC 0116

C	STATION HEADER RECORD	JCOC0117
C		JCOC0118
C	THIS BLOCK IS USED BY	JCOC0119
C	OLDINPUT	JCOC0120
C	NEWINPUT	JCOC0121
C	HEADPAGE	JCOC0122
C		JCOC0123
	COMMON/HEADING/COUNTRY,SHIP,	JCOC0124
*	LATUDE,LATNS,	JCOC0125
*	LONGUDE,LONGEW,	JCOC0126
*	YEAR,MONTH,DAY,	JCOC0127
*	CRUISENO,STATNO,	JCOC0128
*	BOTDEP, DEEPEST,	JCOC0129
*	TIMODAY,	JCOC0130
*	LATMIN, LONGMIN,	JCOC0131
*	HOURL,USIT	JCOC0132
C		JCOC0133
	INTEGER COUNTRY,SHIP(2)	JCOC0134
	INTEGER LATUDE,LATNS	JCOC0135
	INTEGER LONGUDE, LONGEW	JCOC0136
	INTEGER YEAR,MONTH,DAY	JCOC0137
	INTEGER CRUISENO,STATNO(3)	JCOC0138
	INTEGER BOTDEP, DEEPEST	JCOC0139
	INTEGER TIMODAY(4)	JCOC0140
	REAL LATMIN, LONGMIN	JCOC0141
	REAL HOURL	JCOC0142
C		JCOC0143
C		JCOC0144
	INTEGER HEADATA(16)	JCOC0145
C		JCOC0146
C		JCOC0147
C		JCOC0148
C	*****	JCOC0149
C	* * *	JCOC0150
C	* COMMON /INKERS/ *	JCOC0151
C	* * *	JCOC0152
C	*****	JCOC0153
C		JCOC0154
C	THIS BLOCK CONTAINS INDICATORS USED IN	JCOC0155
C	DECODING THE INPUT	JCOC0156
C		JCOC0157
C	THIS BLOCK IS USED BY	JCOC0158
C	OLDMASTR, NEWMASTR	JCOC0159
C	OLDECOD, NUDECOD	JCOC0160
C		JCOC0161
C		JCOC0162
	COMMON/INKERS/ DQUAL, D THERM, TPREC, TQUAL,	JCOC0163
+	SPREC, SQUAL, OPREC, OQUAL,	JCOC0164
+	IPREC, FPREC, SIPREC,	JCOC0165
+	NO2PR, NO3PR, PHPR	JCOC0166
	INTEGER DQUAL, D THERM, TPREC, TQUAL, SPREC, SQUAL	JCOC0167
	INTEGER OPREC, OQUAL, IPREC, FPREC, SIPREC	JCOC0168
	INTEGER NO2PR, NO3PR, PHPR	JCOC0169
C		JCOC0170
C		JCOC0171
C	*****	JCOC0172
C	* * *	JCOC0173
C	* COMMON /VARUSE/ *	JCOC0174
C	* * *	JCOC0175
C	*****	JCOC0176

C		JCOC0177
C	THIS BLOCK CONTAINS VALUES OF INPUT	JCOC0178
C	PARAMETERS FOR ONE DEPTH	JCOC0179
C		JCOC0180
C	THIS BLOCK IS USED BY	JCOC0181
C	OLDMASTR, NEWMASTR	JCOC0182
C	OLCECOD, NUDECOD	JCOC0183
C		JCOC0184
C		JCOC0185
	COMMON/VARUSE/ TEMP, SALTY, DPTH, OX, PHOS, TPHOS, SIL,	JCOC0186
	+ NO2, NO3, PH	JCOC0187
	REAL DPTH, OX, PHOS, TPHOS, SIL	JCOC0188
	REAL NO2, NO3, PH	JCOC0189
	DOUBLE PRECISION TEMP, SALTY	JCOC0190
C		JCOC0191
C		JCOC0192
C	*****	JCOC0193
C	* * *	JCOC0194
C	* COMMON /INPUTVAR/ *	JCOC0195
C	* * *	JCOC0196
C	*****	JCOC0197
C		JCOC0198
C	THIS BLOCK CONTAINS VALUES OF INPUT VARIABLES	JCOC0199
C	DEPTH, TEMPERATURE, AND SALINITY	JCOC0200
C	LATITUDE AND LONGITUDE, IN DEGREES	JCOC0201
C	NGOOD IS NUMBER OF VALUES IN DEPTH, ETC. ARRAYS	JCOC0202
C		JCOC0203
C	THIS BLOCK IS USED BY	JCOC0204
C	OLDINPUT	JCOC0205
C	NEWINPUT	JCOC0206
C	BASICOMP	JCOC0207
C	STDINTRP	JCOC0208
C		JCOC0209
	COMMON/INPUTVAR/DEPTH,	JCOC0210
	* TEMPTURE,	JCOC0211
	* SALINITY,	JCOC0212
	* XLAT,XLONG,	JCOC0213
	* NGOOD	JCOC0214
C		JCOC0215
	DOUBLE PRECISION DEPTH(140)	JCOC0216
	DOUBLE PRECISION TEMPTURE(140)	JCOC0217
	DOUBLE PRECISION SALINITY(140)	JCOC0218
	DOUBLE PRECISION XLAT,XLONG	JCOC0219
	INTEGER NGOOD	JCOC0220
C		JCOC0221
C		JCOC0222
C	*****	JCOC0223
C	* * *	JCOC0224
C	* COMMON /OUTPART1/ *	JCOC0225
C	* * *	JCOC0226
C	*****	JCOC0227
C		JCOC0228
C	THIS BLOCK CONTAINS THE ARRAY OF FORMATTED PRINT LINES	JCOC0229
C	INDICATORS FOR COMPLETE DATA	JCOC0230
C	THESE ARRAYS ARE USED FOR LOCAL VARIABLES	JCOC0231
C	IN STDINTRP AND VOLTRANS	JCOC0232
C		JCOC0233
C	THIS BLOCK IS USED BY	JCOC0234
C	OLDINPUT	JCOC0235
C	NEWINPUT	JCOC0236

C	BASICOMP	JCOC0237
C		JCOC0238
CCCC	COMMON/OUTPART1/PRINTLIN,	JCOC0239
CCCC *	INDCTR,	JCOC0240
CCCC *	NPLIN	JCOC0241
C		JCOC0242
CCCC	INTEGER PRINTLIN(14,140)	JCOC0243
CCCC	INTEGER INDCTR(140)	JCOC0244
CCCC	INTEGER NPLIN/14/	JCOC0245
C		JCOC0246
C		JCOC0247
C	*****	JCOC0248
C	* *	JCOC0249
C	* COMMON /DATASAV/ *	JCOC0250
C	* *	JCOC0251
C	*****	JCOC0252
C		JCOC0253
C	THIS BLOCK CONTAINS ARRAYS WHICH ARE STORED	JCOC0254
C	ON TAPE OR DISK FOR LATER PROCESSING	JCOC0255
C		JCOC0256
	COMMON/DATASAV/ SAVDAT2(5,140),	JCOC0257
+	NDETAIL, NINT,	JCOC0258
+	NSAV1, NSAV2	JCOC0259
	REAL SAVDAT2	JCOC0260
	INTEGER NDETAIL, NINT	JCOC0261
	INTEGER NSAV1/19/, NSAV2/5/	JCOC0262
C		JCOC0263
C		JCOC0264
C	*****	JCOC0265
C	* *	JCOC0266
C	* COMMON /STANDARD/ *	JCOC0267
C	* *	JCOC0268
C	*****	JCOC0269
C		JCOC0270
C	THIS BLOCK CONTAINS STANDARD DEPTHS AND	JCOC0271
C	STANDARD PRESSURES AND NUMBER OF EACH	JCOC0272
C		JCOC0273
C	THIS BLOCK IS USED BY	JCOC0274
C	STDINTRP	JCOC0275
C	VOLTRANS	JCOC0276
C		JCOC0277
	COMMON/STANDARD/ STANDEP(140), STANPRES(140),	JCOC0278
+	NSDEP, NSPRES, IREF, IPROP	JCOC0279
	DOUBLE PRECISION STANDEP, STANPRES	JCOC0280
	INTEGER NSDEP, NSPRES, IREF	JCOC0281
C		JCOC0282
C		JCOC0283
C	*****	JCOC0284
C	* *	JCOC0285
C	* COMMON/PART123/ *	JCOC0286
C	* *	JCOC0287
C	*****	JCOC0288
C		JCOC0289
C	THIS BLOCK CONTAINS ARRAYS FOUND BY PART 1	JCOC0290
C	WHICH ARE USED BY PART 2 AND/OR PART 3	JCOC0291
C		JCOC0292
C	THIS BLOCK IS USED BY	JCOC0293
C	BASICOMP	JCOC0294
C	STDINTRP	JCOC0295
C	VOLTRANS	JCOC0296

JC0C0297
JC0C0298
JC0C0299
JC0C0300
JC0C0301
JC0C0302
JC0C0303
JC0C0304
JC0C0305
JC0C0306
JC0C0307
JC0C0308
JC0C0309
JC0C0310
JC0C0311
JC0C0312
JC0C0313
JC0C0314
JC0C0315
JC0C0316
JC0C0317
JC0C0318
JC0C0319
JC0C0320
JC0C0321
JC0C0322
JC0C0323
JC0C0324
JC0C0325
JC0C0326
JC0C0327
JC0C0328
JC0C0329
JC0C0330
JC0C0331
JC0C0332
JC0C0333
JC0C0334
JC0C0335
JC0C0336
JC0C0337
JC0C0338
JC0C0339
JC0C0340
JC0C0341
JC0C0342
JC0C0343
JC0C0344
JC0C0345
JC0C0346
JC0C0347
JC0C0348
JC0C0349
JC0C0350
JC0C0351
JC0C0352
JC0C0353
JC0C0354
JC0C0355
JC0C0356

C	FOR FURTHER PROCESSING	*	JCOC0357
C		*	JCOC0358
C	TOTSTAT IS NUMBER OF STATIONS TO PROCESS	*	JCOC0359
C	IF LEFT BLANK, WILL PROCESS ALL STATIONS	*	JCOC0360
C	(SETS TOTSTAT TO 10000)	*	JCOC0361
C		*	JCOC0362
C	CHECKS FOR TAPE POSITIONING,	*	JCOC0363
C	AND DOES IT IF REQUESTED	*	JCOC0364
C		*	JCOC0365
C	READS STANDARD DEPTHS AND/OR PRESSURES,	*	JCOC0366
C	AS REQUIRED	*	JCOC0367
C		*	JCOC0368
C	GETS TIME OF DAY AND PRINTS HEADER PAGE	*	JCOC0369
C		*	JCOC0370
C		*	JCOC0371
C		*	JCOC0372
C	*****	*	JCOC0373
C			JCOC0374
C	READ AND CHECK OPTION CARD		JCOC0375
C			JCOC0376
	READ(ICR,1000) MESS,COMPOPT,		JCOC0377
	+ INPOPT,RECSKP,		JCOC0378
	+ SAVOPT,TOTSTAT		JCOC0379
1000	FORMAT(A4,6X,5I5)		JCOC0380
	READ(ICR,1003) IPROP		JCOC0381
1003	FORMAT(I1)		JCOC0382
	READ(ICR,1003) NOFTS		JCOC0383
	IF (MESS .NE. MOPE) GO TO 500		JCOC0384
	IF (TOTSTAT .EQ. 0) TOTSTAT = 10000		JCOC0385
	OUTPUT 'OCCOMP VERSION MAY 1978'		JCOC0386
	OUTPUT COMPOPT,INPOPT,RECSKP,SAVOPT,TOTSTAT,IPROP,NOFTS		JCOC0387
	IF (COMPOPT .LT. 0 .OR. COMPOPT .GT. 3) GO TO 530		JCOC0388
	BLKFACT = INPOPT		JCOC0389
	IF (INPOPT .LE. 0) BLKFACT = 1		JCOC0390
C			JCOC0391
C	CHECK FOR TAPE POSITIONING		JCOC0392
C			JCOC0393
	IF ((INPOPT .EQ. 0) .OR. (RECSKP .LE. 0)) GO TO 20		JCOC0394
	CALL SETSKP (INDSKP)		JCOC0395
	CALL SKPREC (INUNIT, RECSKP)		JCOC0396
	IF (INDSKP .NE. 2) GO TO 510		JCOC0397
C			JCOC0398
C	GET STANDARD DEPTHS AND/OR		JCOC0399
C	PRESSURES, AS REQUIRED		JCOC0400
C			JCOC0401
	20 CONTINUE		JCOC0402
	IF ((COMPOPT .EQ. 1) .OR. (COMPOPT .EQ. 3))		JCOC0403
	* CALL READARAY(STANDEP,NSDEP)		JCOC0404
	IF (COMPOPT .LT. 2) GO TO 40		JCOC0405
C			JCOC0406
	READ(ICR,1010) REFLEVL		JCOC0407
1010	FORMAT (G)		JCOC0408
	CALL READARAY (STANPRES, NSPRES)		JCOC0409
C			JCOC0410
	IREF = 0		JCOC0411
	DO 30 IR=1,NSPRES		JCOC0412
	IF (STANPRES(IR) .EQ. REFLEVL) IREF = IR		JCOC0413
	30 CONTINUE		JCOC0414
	IF (IREF .EQ. 0) GO TO 520		JCOC0415
C			JCOC0416


```

C                                     GET TIME OF DAY AND PRINT FIRST PAGE
C                                     SET STATION COUNT TO ZERO
C
C 40 CONTINUE
C     CALL TODAY(TIMODAY)
C     CALL HEADPAGE
C     NUMSTAT = 0
C     ENDATA = .FALSE.
C     IF(NOFTS.EQ.0) GO TO 41
C     CALL SKPFIL(SAVFIL,NOFTS)
C 41  CALL TIC
C
C *****
C                                     *
C     *****                                     *
C     *                                     *
C     *      COMPUTATION SECTION      *
C     *                                     *
C     *                                     *
C     *****                                     *
C
C MAIN PROGRAM LOOP
C FOR EACH STATION, TAKES THE FOLLOWING STEPS:
C
C     BLANKS PRINT BUFFER
C     SETS THE SAVE DATA ARRAY TO DEFAULT
C
C     INPUTS DATA FOR STATION
C
C     DOES REQUIRED COMPUTATION AND OUTPUT
C     STORES DATA ARRAYS IF REQUESTED
C
C STOPS WHEN END OF INPUT FILE IS REACHED,
C OR SPECIFIED NUMBER OF STATIONS COMPLETED.
C
C *****
C
C REPEAT 100, WHILE .NOT. ENDATA .AND. NUMSTAT .LT. TOTSTAT
C
C     USIT = .TRUE.
C     NUMSTAT = NUMSTAT+1
C     CALL NEWINPUT
C     IF(.NOT.USIT) GO TO 100
C     IF(NGOOD.LT.3)OUTPUT ' STATION DELETED,<3GOOD DEPTHS';GO TO 100
C
C CALL BASICOMP
C NINT = 0
C IF ( (COMPOPT .EQ. 1) .OR. (COMPOPT .EQ. 3 ))
C +   CALL STDINTRP
C
C IF ( COMPOPT .GE. 2 ) CALL VOLTRANS
C IF ( SAVOPT .NE. 0 ) CALL SAVARAY
C
C 100 CONTINUE
C
C     CALL TOC(SECTP)
C     WRITE(LP,3000) SECTP
C     IF ( SAVOPT .NE. 0 ) END FILE SAVFIL
C     CALL EXIT
C

```

JCOC0417
 JCOC0418
 JCOC0419
 JCOC0420
 JCOC0421
 JCOC0422
 JCOC0423
 JCOC0424
 JCOC0425
 JCOC0426
 JCOC0427
 JCOC0428
 JCOC0429
 JCOC0430
 JCOC0431
 JCOC0432
 JCOC0433
 JCOC0434
 JCOC0435
 JCOC0436
 JCOC0437
 JCOC0438
 JCOC0439
 JCOC0440
 JCOC0441
 JCOC0442
 JCOC0443
 JCOC0444
 JCOC0445
 JCOC0446
 JCOC0447
 JCOC0448
 JCOC0449
 JCOC0450
 JCOC0451
 JCOC0452
 JCOC0453
 JCOC0454
 JCOC0455
 JCOC0456
 JCOC0457
 JCOC0458
 JCOC0459
 JCOC0460
 JCOC0461
 JCOC0462
 JCOC0463
 JCOC0464
 JCOC0465
 JCOC0466
 JCOC0467
 JCOC0468
 JCOC0469
 JCOC0470
 JCOC0471
 JCOC0472
 JCOC0473
 JCOC0474
 JCOC0475
 JCOC0476

C		JCOC0477
C	*****	JCOC0478
C		JCOC0479
C	ERROR ROUTINES	JCOC0480
C		JCOC0481
C	*****	JCOC0482
C		JCOC0483
C		JCOC0484
	500 CONTINUE	JCOC0485
	WRITE(LP,2010)	JCOC0486
	2010 FORMAT(' OCCOMP OPTION CARD MISSING')	JCOC0487
	CALL ABORT	JCOC0488
C		JCOC0489
	510 CONTINUE	JCOC0490
	WRITE(LP,2020) INDSKP	JCOC0491
	2020 FORMAT(' SKPREC ERROR. IND = ',I3)	JCOC0492
	CALL ABORT	JCOC0493
C		JCOC0494
	520 CONTINUE	JCOC0495
	WRITE(LP,2030) REFLEVL	JCOC0496
	2030 FORMAT(' REFERENCE LEVEL ',G13.5,' NOT AT STANDARD PRESSURE')	JCOC0497
	CALL ABORT	JCOC0498
C		JCOC0499
	530 CONTINUE	JCOC0500
	WRITE(LP,2040) COMPOPT	JCOC0501
	2040 FORMAT(' INVALID COMPOPT ',I5)	JCOC0502
	CALL ABORT	JCOC0503
C		JCOC0504
	3000 FORMAT(F10.3,' SECONDS TO PROCESS DATA')	JCOC0505
C		JCOC0506
	END	JCOC0507
C	HEADPAGE.1 APRIL 27, 1978	JCOC0508
	SUBROUTINE HEADPAGE	JCOC0509
C		JCOC0510
C	PART OF PROGRAM OCCOMP	JCOC0511
C	HEADPAGE PRINTS INTRODUCTORY PAGE	JCOC0512
C	ENTRY PRNTHD PRINTS PAGE HEADINGS FOR STATION	JCOC0513
C	ALL COMMUNICATION BY LABELLED COMMON	JCOC0514
C		JCOC0515
C	PROGRAMMER: MARY HUNT	JCOC0516
C	DATE: FEBRUARY, 1977	JCOC0517
C		JCOC0518
C	*****	JCOC0519
C		JCOC0520
	COMMON/LOGUNIT/ LP, ICR, INUNIT, SAVFIL	JCOC0521
	INTEGER LP, ICR, INUNIT, SAVFIL	JCOC0522
C		JCOC0523
C	*****	JCOC0524
C		JCOC0525
	COMMON/HEADING/ COUNTRY, SHIP(2), LATUDE, LATNS,	JCOC0526
	+ LONGUDE, LONGEW,	JCOC0527
	+ YEAR, MONTH, DAY, CRUISENO,	JCOC0528
	+ STATNO(3), BOTDEP, DEEPEST, TIMODAY(4),	JCOC0529
	+ LATMIN, LONGMIN, HOUR,USIT	JCOC0530
	INTEGER COUNTRY, SHIP, LATUDE, LATNS, LONGUDE, LONGEW	JCOC0531
	INTEGER YEAR, MONTH, DAY, CRUISENO	JCOC0532
	INTEGER STATNO, BOTDEP, TIMODAY	JCOC0533
	INTEGER DEEPEST	JCOC0534
	REAL LATMIN, LONGMIN, HOUR	JCOC0535
C		JCOC0536


```

C*****
C
  INTEGER ROMAN(12)
  DATA ROMAN/' I',' II',' III',' IV',' V',' VI',
  *           'VII',' VIII',' IX',' X',' XI',' XII'/
C
C *****
C
  WRITE(LP,2030)
2030 FORMAT('1PROGRAM OCCOMP',
  *        T23,'REVISION APRIL, 1978'////)
  WRITE(LP,2040)
2040 FORMAT(T4,'THE UNITS OF QUANTITIES USED IN THE',
  *        T40,'FOLLOWING CALCULATIONS ARE' ////
2      T4,'DEPTH',T40,'METERS'//
3      T4,'PRESSURE',T40,'DECIBARS'//
4      T4,'TEMPERATURE',T40,'DEGREES CELSIUS'//
5      T4,'SALINITY',T40,'PARTS PER THOUSAND'//
6      T4,'OXYGEN',T40,'MILLILITERS PER LITER'//
7      T4,'INCRGANIC PHOSPHATE (PO4)',
8      T40,'MICROGRAM-ATOMS PER LITER'//
9      T4,'TOTAL PHOSPHORUS (PHOS)',
A      T40,'MICROGRAM-ATOMS PER LITER'//
B      T4,'SILICATE-SILICON',T40,'MICROGRAM-ATOMS PER LITER'//
B      T4,'NITRATES (NO3)',T40,'MICROGRAM-ATOMS PER LITER'//
C      T4,'ANOMALY OF DENSITY AT ATMOSPHERIC',
D      T40,'GRAMS PER LITER'/
E      T4,'PRESSURE (SIGMA-THETA)'//
F      T4,'POTENTIAL TEMPERATURE (THETA)',
G      T40,'DEGREES CELSIUS'//
H      T42,'-5'/
I      T4,'SPECIFIC VOLUME ANOMALY (DELTA)',
J      T40,'10 MILLILITERS PER GRAM'//
K      T65,'2',T77,'2'/
L      T4,'DYNAMIC HEIGHT',
M      T40,'DYNAMIC METERS (10 METERS PER SECOND)'//
N      T4,'SOUND VELOCITY',T40,'METERS PER SECOND'//
O      T42,'8',T63,'2'/T4,'POTENTIAL ENERGY',
P      T40,'10 ERGS PER CENTIMETER'//
Q      T4,'BRUNT-VAISALA FREQUENCY',
R      T40,'CYCLES PER HOUR'//
S      T4,'BRUNT-VAISALA PERIOD',T40,'MINUTES'//
T      T4,'VELOCITY',T40,'CENTIMETERS PER SECOND'//
U      T42,'6'/T4,'VOLUME TRANSPORT',
V      T40,'10 CUBIC METERS PER SECOND')
C
  RETURN
C
C *****
C
  ENTRY PRNTHAD
C
C *****
C
  WRITE(LP,2000) TIMODAY
2000 FORMAT(
  *      T1,'1COUNTRY',T11,'SHIP',
  *      T18,'CRUISE',T27,'STATION',
  *      T40,'DATE',T51,'GMT',
  *      T58,'LATITUDE',T70,'LONGITUDE',
  *      T82,'DEPTH',T104,4A4)

```

JCNC0537
 JCNC0538
 JCNC0539
 JCNC0540
 JCNC0541
 JCNC0542
 JCNC0543
 JCNC0544
 JCNC0545
 JCNC0546
 JCNC0547
 JCNC0548
 JCNC0549
 JCNC0550
 JCNC0551
 JCNC0552
 JCNC0553
 JCNC0554
 JCNC0555
 JCNC0556
 JCNC0557
 JCNC0558
 JCNC0559
 JCNC0560
 JCNC0561
 JCNC0562
 JCNC0563
 JCNC0564
 JCNC0565
 JCNC0566
 JCNC0567
 JCNC0568
 JCNC0569
 JCNC0570
 JCNC0571
 JCNC0572
 JCNC0573
 JCNC0574
 JCNC0575
 JCNC0576
 JCNC0577
 JCNC0578
 JCNC0579
 JCNC0580
 JCNC0581
 JCNC0582
 JCNC0583
 JCNC0584
 JCNC0585
 JCNC0586
 JCNC0587
 JCNC0588
 JCNC0589
 JCNC0590
 JCNC0591
 JCNC0592
 JCNC0593
 JCNC0594
 JCNC0595
 JCNC0596

```

C      WRITE(LP,2010)  CCOUNTRY,SHIP,
*      CRUISENO,STATNO,
*      DAY,ROMAN(MONTH),YEAR,HOURL,
*      LATUDE,LATMIN,LATNS,
*      LONGUDE,LONGMIN,LONGEW,
*      BOTDEP
2010 FORMAT(T5,A3,T12,A4,A2,
*      T19,A3,T27,A3,A4,A2,
*      T37,I2,'-',A4,'-',I2,T50,F4.1,
*      T57,I2,F5.1,1X,A1,
*      T69,I3,F5.1,1X,A1,
*      T82,I5//)
C
C      RETURN
C
C      END
C      NEWINPUT.3      MAY 1, 1978
C      SUBROUTINE NEWINPUT
C
C      THIS SUBROUTINE IS PART OF PROGRAM OCCOMP
C      IT INPUTS DATA FOR ONE STATION IN 1974 NODC FORMAT
C
C      PROGRAMMER:  MARY HUNT
C      DATE:      MARCH, 1977
C
C      *****
C      COMMON/LOGUNIT/ LP, ICR, INUNIT, SAVFIL
C      INTEGER LP, ICR, INUNIT, SAVFIL
C
C      *****
C      COMMON/OPTIONS/ COMPOPT, INPOPT, SAVOPT,
*      BLKFACT, ENDDATA
C      COMMON/OPTIONS/ NUMSTAT, MAXOBS
C      INTEGER COMPOPT, INPOPT, SAVOPT, BLKFACT,
C      LOGICAL ENDDATA
C      INTEGER NUMSTAT
C      INTEGER MAXOBS
C
C      *****
C      COMMON/BUFFERS/ INBUFR(1600), NUBUF(20), INDEX
C      INTEGER INBUFR, NUBUF, INDEX
C
C      *****
C      COMMON/ALPHNUM/ IBLNK, ASTRISK, ZEE, CUE, PEA,
+      JAY, ONE, THREE, FOUR,
+      ESS, EEE, ENN, UU
C      INTEGER IBLNK, ASTRISK, ZEE, CUE, PEA
C      INTEGER JAY, ONE, THREE, FOUR
C      INTEGER ESS, EEE, ENN, UU
C
C      COMMON/INPUTVAR/ DEPTH(140),TEMPTURE(140),SALINITY(140),
&      XLAT,XLONG,NGOOD

```

JCOC0597
 JCOC0598
 JCOC0599
 JCOC0600
 JCOC0601
 JCOC0602
 JCOC0603
 JCOC0604
 JCOC0605
 JCOC0606
 JCOC0607
 JCOC0608
 JCOC0609
 JCOC0610
 JCOC0611
 JCOC0612
 JCOC0613
 JCOC0614
 JCOC0615
 JCOC0616
 JCOC0617
 JCOC0618
 JCOC0619
 JCOC0620
 JCOC0621
 JCOC0622
 JCOC0623
 JCOC0624
 JCOC0625
 JCOC0626
 JCOC0627
 JCOC0628
 JCOC0629
 JCOC0630
 JCOC0631
 JCOC0632
 JCOC0633
 JCOC0634
 JCOC0635
 JCOC0636
 JCOC0637
 JCOC0638
 JCOC0639
 JCOC0640
 JCOC0641
 JCOC0642
 JCOC0643
 JCOC0644
 JCOC0645
 JCOC0646
 JCOC0647
 JCOC0648
 JCOC0649
 JCOC0650
 JCOC0651
 JCOC0652
 JCOC0653
 JCOC0654
 JCOC0655
 JCOC0656


```

DOUBLE PRECISION DEPTH,TEMPTURE,SALINITY,XLAT,XLONG
INTEGER NGOOD
C
C*****
C
COMMON/DATASAV/          SAVDAT2(5,140),
+      NDETAIL, NINT,
+      NSAV1, NSAV2
REAL      SAVDAT2
INTEGER NDETAIL, NINT
INTEGER NSAV1, NSAV2
C
C*****
C
COMMON/LOCAL/          LCHAR, INKER, NW, ITST,
+      JUNK(1687)
INTEGER      LCHAR, INKER, NW, ITST
INTEGER JUNK
C
C*****
C
LOCAL VARIABLES.  MUST NOT BE CHANGED BY
OTHER ROUTINES.
C
INTEGER ND, NDET
INTEGER HEADATA(16)
LOGICAL USIT
INTEGER CONINO
COMMON/HEADING/ COUNTRY, SHIP(2), LATUDE, LATNS,
+      LONGUDE, LONGEW,
+      YEAR, MONTH, DAY, CRUISENO,
+      STATNO(3), BOTDEP, DEEPEST, TIMODAY(4),
+      LATMIN, LONGMIN, HOUR,USIT
EQUIVALENCE (HEADATA(1),COUNTRY), (HEADATA(2),SHIP)
EQUIVALENCE (HEADATA(4),LATUDE), (HEADATA(5),LATNS)
EQUIVALENCE (HEADATA(6),LONGUDE), (HEADATA(7),LONGEW)
EQUIVALENCE (HEADATA(8),YEAR), (HEADATA(9),MONTH)
EQUIVALENCE (HEADATA(10),DAY), (HEADATA(11),CRUISENO)
EQUIVALENCE (HEADATA(12),STATNO), (HEADATA(15),BOTDEP)
EQUIVALENCE (HEADATA(16), DEEPEST )
C
C      READ RECORD FROM TAPE AND
C      CHECK FOR ERROR OR EOF
C
C      INTO = 0
C      NGOOD = 0
C      NDETAIL = 0
10 CONTINUE
CALL BUFFERIN ( INUNIT, 0, INBUFR, 1600, INKER, NW, ITST )
C
IF ( INKER .EQ. 3 ) ENDATA = .TRUE. ; RETURN
IF ( INKER .NE. 4 ) GO TO 20
C
IF ( NUMSTAT .EQ. 0 )
+ WRITE(LP,2000) ITST ;
+ CALL ABORT
2000 FORMAT ( ' READ ERROR ON FIRST RECORD. CODE IS ',Z2 )
C
WRITE(LP,2010) ITST
2010 FORMAT(' READ ERROR. CODE IS ',Z2 )

```

JCOC 0657
 JCOC 0658
 JCOC 0659
 JCOC 0660
 JCOC 0661
 JCOC 0662
 JCOC 0663
 JCOC 0664
 JCOC 0665
 JCOC 0666
 JCOC 0667
 JCOC 0668
 JCOC 0669
 JCOC 0670
 JCOC 0671
 JCOC 0672
 JCOC 0673
 JCOC 0674
 JCOC 0675
 JCOC 0676
 JCOC 0677
 JCOC 0678
 JCOC 0679
 JCOC 0680
 JCOC 0681
 JCOC 0682
 JCOC 0683
 JCOC 0684
 JCOC 0685
 JCOC 0686
 JCOC 0687
 JCOC 0688
 JCOC 0689
 JCOC 0690
 JCOC 0691
 JCOC 0692
 JCOC 0693
 JCOC 0694
 JCOC 0695
 JCOC 0696
 JCOC 0697
 JCOC 0698
 JCOC 0699
 JCOC 0700
 JCOC 0701
 JCOC 0702
 JCOC 0703
 JCOC 0704
 JCOC 0705
 JCOC 0706
 JCOC 0707
 JCOC 0708
 JCOC 0709
 JCOC 0710
 JCOC 0711
 JCOC 0712
 JCOC 0713
 JCOC 0714
 JCOC 0715
 JCOC 0716

<pre> C GO TO 10 C C HAVE A RECORD IN CORE NOW. C SET INDEX AND DECODE STATION HEADER INFO. C 20 CONTINUE INDEX = 1 NDET = (NW/20) - 2 DECODE(4,1010,INBUFR) CONIND 1010 FORMAT (I1) C IF (CONIND.GT.1.AND.INTO.EQ.0)WRITE(108,801);USIT =.FALSE.; & RETURN 801 FORMAT(' FIRST RECORD IN STATION IS A CONTINUATION RECORD, & STATION DELETED') IF (CONIND .GT. 1) GO TO 25 INTO = 1 C CALL NEWMASTR(INBUFR) CALL USELECT(HEADATA,USIT) IF(USIT) GO TO 25 IF((CONIND.GT.0).AND.(CONIND.LT.9)) CALL SKPREC(INUNIT,1) RETURN C C NOW DECODE DETAIL OBSERVATIONS C FROM THIS RECORD C 25 CONTINUE INDEX = INDEX + 40 DO 40 ND=1,NDET DECODE (80,1000,INBUFR(INDEX)) LCHAR IF ((LCHAR .NE. THREE) .AND. (LCHAR .NE. FOUR)) GO TO 30 NDETAIL = NDETAIL + 1 IF (NDETAIL .GT. MAXOBS) GO TO 50 CALL NEWDETAIL (INBUFR(INDEX)) 30 CONTINUE INDEX = INDEX + 20 40 CONTINUE 1000 FORMAT(T80,A1) C C CHECK IF ANOTHER RECORD FOR C THIS STATION C IF ((CONIND .GT. 0) .AND. (CONIND .LT. 9)) GO TO 10 C 50 CONTINUE DEEPEST = DEPTH(NGOOD) RETURN END C NEWMASTR.3 MAY 1, 1978 SUBROUTINE NEWMASTR (BUFR) C INTEGER BUFR (1) C DECODES STATION HEADER INFORMATION NODC 1974 FORMAT C PROGRAMMER: MARY HUNT DATE: MARCH, 1977 C </pre>	<pre> JCOC0717 JCOC0718 JCOC0719 JCOC0720 JCOC0721 JCOC0722 JCOC0723 JCOC0724 JCOC0725 JCOC0726 JCOC0727 JCOC0728 JCOC0729 JCOC0730 JCOC0731 JCOC0732 JCOC0733 JCOC0734 JCOC0735 JCOC0736 JCOC0737 JCOC0738 JCOC0739 JCOC0740 JCOC0741 JCOC0742 JCOC0743 JCOC0744 JCOC0745 JCOC0746 JCOC0747 JCOC0748 JCOC0749 JCOC0750 JCOC0751 JCOC0752 JCOC0753 JCOC0754 JCOC0755 JCOC0756 JCOC0757 JCOC0758 JCOC0759 JCOC0760 JCOC0761 JCOC0762 JCOC0763 JCOC0764 JCOC0765 JCOC0766 JCOC0767 JCOC0768 JCOC0769 JCOC0770 JCOC0771 JCOC0772 JCOC0773 JCOC0774 JCOC0775 JCOC0776 </pre>
---	--

C		JCNC0777
C	*****	JCNC0778
C		JCNC0779
	COMMON/ALPHNUM/ IBLNK, ASTRISK, ZEE, CUE, PEA,	JCNC0780
	+ JAY, ONE, THREE, FOUR,	JCNC0781
	+ ESS, EEE, ENN, UU	JCNC0782
	INTEGER IBLNK, ASTRISK, ZEE, CUE, PEA	JCNC0783
	INTEGER JAY, ONE, THREE, FOUR	JCNC0784
	INTEGER ESS, EEE, ENN, UU	JCNC0785
C		JCNC0786
C	*****	JCNC0787
C		JCNC0788
	COMMON/HEADING/ COUNTRY, SHIP(2), LATUDE, LATNS,	JCNC0789
	+ LONGUDE, LONGEW,	JCNC0790
	+ YEAR, MONTH, DAY, CRUISENO,	JCNC0791
	+ STATNO(3), BOTDEP, DEEPEST, TIMODAY(4),	JCNC0792
	+ LATMIN, LONGMIN, HOUR,USIT	JCNC0793
	INTEGER COUNTRY, SHIP, LATUDE, LATNS, LONGUDE, LONGEW	JCNC0794
	INTEGER YEAR, MONTH, DAY, CRUISENO	JCNC0795
	INTEGER STATNO, BOTDEP, TIMODAY	JCNC0796
	INTEGER DEEPEST	JCNC0797
	REAL LATMIN, LONGMIN, HOUR	JCNC0798
C		JCNC0799
C	*****	JCNC0800
C		JCNC0801
	COMMON/VARUSE/ TEMP, SALTY, DPTH, OX, PHOS, TPHOS, SIL,	JCNC0802
	+ NO2, NO3, PH	JCNC0803
	DOUBLE PRECISION TEMP, SALTY	JCNC0804
	REAL DPTH, OX, PHOS, TPHOS, SIL	JCNC0805
	REAL NO2, NO3, PH	JCNC0806
C		JCNC0807
C	*****	JCNC0808
C		JCNC0809
	COMMON/INPUTVAR/ DEPTH(140), TEMPTURE(140), SALINITY(140),	JCNC0810
	+ XLAT, XLONG, NGOOD	JCNC0811
	DOUBLE PRECISION DEPTH, TEMPTURE, SALINITY, XLAT, XLONG	JCNC0812
	INTEGER NGOOD	JCNC0813
C		JCNC0814
C	*****	JCNC0815
C		JCNC0816
C		JCNC0817
C	*****	JCNC0818
C		JCNC0819
	COMMON/DATASAV/ SAVDAT2(5,140),	JCNC0820
	+ NDETAIL, NINT,	JCNC0821
	+ NSAV1, NSAV2	JCNC0822
	REAL SAVDAT2	JCNC0823
	INTEGER NDETAIL, NINT	JCNC0824
	INTEGER NSAV1, NSAV2	JCNC0825
C		JCNC0826
C	*****	JCNC0827
C		JCNC0828
	COMMON/INKERS/ DQUAL, DTherm, TPrec, TQUAL,	JCNC0829
	+ SPRec, SQUAL, OPRec, OQUAL,	JCNC0830
	+ IPRec, FPRC, SIPRec,	JCNC0831
	+ NO2PR, NO3PR, PHPR	JCNC0832
	INTEGER DQUAL, DTherm, TPrec, TQUAL, SPRec, SQUAL	JCNC0833
	INTEGER OPRec, OQUAL, IPRec, FPRC, SIPRec	JCNC0834
	INTEGER NO2PR, NO3PR, PHPR	JCNC0835
C		JCNC0836

```

C *****
C
C      DECODE(160,1000,BUFR)
C      +          COUNTRY, LATNS, LATUDE, LATMIN,
C      +          LONGEW, LONGUDE, LONGMIN,
C      +          YEAR, MONTH, DAY, HOUR, SHIP(1), SHIP(2),
C      +          BOTDEP, CRUISENO, STATNO(1), STATNO(2), STATNO(3),
C      +          NEXT
C
C      1000 FORMAT( T2,A3,      T27,A1,I2,F3.1,
C      +          T33,A1,I3,F3.1,
C      +          T41,I2,F3.1, T50,A4,A2,
C      +          T56,I5, T95,A3,A3,A4,A2,
C      +          T159,I1 )
C
C      XLAT = LATUDE + LATMIN/60.0
C      IF ( LATNS .EQ. ESS ) XLAT = -XLAT
C      XLONG = LONGUDE + LONGMIN/60.0
C      IF ( LONGEW .NE. EEE ) XLONG = -XLONG
C
C      RETURN
C
C
C *****
C
C      ENTRY NEWDETAIL ( BUFR )
C
C      HANDLES OBSERVED DEPTH INFORMATION FOR ONE DEPTH
C      IN NEW NODC FORMAT
C
C      DECODE CONTENTS OF RECORD
C
C      DECODE(80,1010,BUFR) DPTH, DQUAL, DTHERM,
C      +          TEMP, TPREC, TQUAL,
C      +          SALTY, SPREC, SQUAL
C
C      1010 FORMAT( I5,I1,A1,
C      +          T8,F5.3,2I1,
C      +          T15,F5.3,2I1)
C
C      CHECK FOR VALIDITY OF TEMP.
C      AND SAL AT THIS DEPTH
C
C      IF ( ( DQUAL .GT. 6 ) .OR.
C      +    ( TQUAL .GT. 6 ) .OR.
C      +    ( SQUAL .GT. 6 ) ) RETURN
C
C      IF( (TPREC .EQ. 9) .OR. (SPREC .EQ. 9) )
C      +    RETURN
C
C      THIS OBSERVATION IS A KEEPER
C
C      NGOOD = NGOOD + 1
C      DEPTH(NGOOD) = DPTH
C      TEMPTURE(NGOOD) = TEMP
C      SALINITY(NGOOD) = SALTY

```

JCOC0837
 JCOC0838
 JCOC0839
 JCOC0840
 JCOC0841
 JCOC0842
 JCOC0843
 JCOC0844
 JCOC0845
 JCOC0846
 JCOC0847
 JCOC0848
 JCOC0849
 JCOC0850
 JCOC0851
 JCOC0852
 JCOC0853
 JCOC0854
 JCOC0855
 JCOC0856
 JCOC0857
 JCOC0858
 JCOC0859
 JCOC0860
 JCOC0861
 JCOC0862
 JCOC0863
 JCOC0864
 JCOC0865
 JCOC0866
 JCOC0867
 JCOC0868
 JCOC0869
 JCOC0870
 JCOC0871
 JCOC0872
 JCOC0873
 JCOC0874
 JCOC0875
 JCOC0876
 JCOC0877
 JCOC0878
 JCOC0879
 JCOC0880
 JCOC0881
 JCOC0882
 JCOC0883
 JCOC0884
 JCOC0885
 JCOC0886
 JCOC0887
 JCOC0888
 JCOC0889
 JCOC0890
 JCOC0891
 JCOC0892
 JCOC0893
 JCOC0894
 JCOC0895
 JCOC0896


```

CCCC  INDCTR(NGOOD) = NDETAIL
C
C      RETURN
C
C      END
C      STDINTRP.3      MAY 1, 1978
C      SUBROUTINE STDINTRP
C
C      PART OF PROGRAM OCCOMP
C      INTERPOLATES TEMPERATURE, THETA, SALINITY, SIGMA-THETA,
C      AND DELTA AT STANDARD DEPTHS.
C
C      PROGRAMMER:  MARY HUNT
C      DATE:        FEBRUARY, 1977
C
C      *****
C
C      COMMON/LOGUNIT/ LP, ICR, INUNIT, SAVFIL
C      INTEGER LP, ICR, INUNIT, SAVFIL
C
C      *****
C
C      COMMON/OPTIONS/ COMPOPT, INPOPT, SAVOPT,
C      *                BLKFACT, ENDATA
C      COMMON/OPTIONS/ NUMSTAT, MAXOBS
C      INTEGER COMPOPT, INPOPT, SAVOPT, BLKFACT
C      LOGICAL ENDATA
C      INTEGER NUMSTAT, MAXOBS
C
C      *****
C
C      COMMON/INPUTVAR/ DEPTH(140), TEMPTURE(140), SALINITY(140),
C      *                XLAT, XLONG, NGOOD
C      DOUBLE PRECISION DEPTH, TEMPTURE, SALINITY, XLAT, XLONG
C      INTEGER NGOOD
C
C      *****
C
C      DIMENSION IK1(140), IK3(140)
C
C      *****
C
C      *****
C
C      COMMON/STANDARD/ STANDEP(140), STANPRES(140),
C      *                NSDEP, NSPRES, IREF, IPROP
C      DOUBLE PRECISION STANDEP, STANPRES
C      INTEGER NSDEP, NSPRES, IREF
C
C      *****
C
C      COMMON/DATASAV/
C      *                NDETAIL, NINT,
C      *                NSAV1, NSAV2
C      REAL            SAVDAT2
C      INTEGER NDETAIL, NINT
C      INTEGER NSAV1, NSAV2

```

```

JCOC0897
JCOC0898
JCOC0899
JCOC0900
JCOC0901
JCOC0902
JCOC0903
JCOC0904
JCOC0905
JCOC0906
JCOC0907
JCOC0908
JCOC0909
JCOC0910
JCOC0911
JCOC0912
JCOC0913
JCOC0914
JCOC0915
JCOC0916
JCOC0917
JCOC0918
JCOC0919
JCOC0920
JCOC0921
JCOC0922
JCOC0923
JCOC0924
JCOC0925
JCOC0926
JCOC0927
JCOC0928
JCOC0929
JCOC0930
JCOC0931
JCOC0932
JCOC0933
JCOC0934
JCOC0935
JCOC0936
JCOC0937
JCOC0938
JCOC0939
JCOC0940
JCOC0941
JCOC0942
JCOC0943
JCOC0944
JCOC0945
JCOC0946
JCOC0947
JCOC0948
JCOC0949
JCOC0950
JCOC0951
JCOC0952
JCOC0953
JCOC0954
JCOC0955
JCOC0956

```



```

C C C C
INTERPOLATIONS COMPLETED.
PRINT PAGE AND COLUMN HEADINGS.

      IF(IPROP.EQ.0) GO TO 78
      CALL PRNTHD
      WRITE(LP,2010)
2010 FORMAT( T8,'STANDARD DEPTH', T26,'TEMP', T36,'THETA',
+          T47,'SAL', T55,'SIGMA-', T66,'DELTA' /
+          T55,'THETA',T75,'PRESS' //)

C C C C
                        PRINT INTERPOLATED ARRAYS AND
                        SAVE FOR OUTPUT TO TAPE OR DISK.

78   INX = 1
      DO 80 IN=FRST,LST
        CALL DSIGMT(XTEM(IN),XSAL(IN),SNTEMP,XSGTH(IN))
        CALL DSPVOL(PRESX(IN),XTEM(IN),SNTEMP,XSGTH(IN),ALPHA1)
        CALL DSVANM(PRESX(IN),ALPHA1,XDEL(IN))
        IF(IPROP.EQ.0) GO TO 79
        WRITE(LP,2020) STANDEP(IN),XTEM(IN),IK1(IN),XSAL(IN),
& IK3(IN),XSGTH(IN),XDEL(IN),PRESX(IN)
2020  FORMAT(T11,I5,T25,F6.3,A1,T45,F6.3,A1,T55,F6.3,T65,
& F6.1,T75,F6.1)
79   IF(SAVOPT.EQ.0) GO TO 80
      SAVDAT2(1,INX) = STANDEP(IN)
      SAVDAT2(2,INX) = XTEM(IN)
      SAVDAT2(3,INX) = XSAL(IN)
      SAVDAT2(4,INX) = XSGTH(IN)
      SAVDAT2(5,INX) = XDEL(IN)

      INX = INX + 1

      80 CONTINUE

      RETURN
      END
SAVARAY.1              APRIL 27, 1978
SUBROUTINE SAVARAY

PART OF PROGRAM OCCOMP
STORES INFORMATION TO BE SAVED AS A BINARY FILE
ON LOGICAL UNIT # 1, USING BUFFEROUT

PROGRAMMER: MARY HUNT
DATE: FEBRUARY, 1977

*****
COMMON/LOGUNIT/ LP, ICR, INUNIT, SAVFIL
INTEGER LP, ICR, INUNIT, SAVFIL
*****
COMMON/HEADING/ COUNTRY, SHIP(2), LATUDE, LATNS,
+ LONGUDE, LONGEW,

```

```

+          YEAR, MONTH, DAY, CRUISENO,
+          STATNO(3), BOTDEP, DEEPEST, TIMODAY(4),
+          LATMIN, LONGMIN, HOUR,USIT
INTEGER COUNTRY, SHIP, LATUDE, LATNS, LONGUDE, LONGEW
INTEGER YEAR, MONTH, DAY, CRUISENO
INTEGER STATNO, BOTDEP, TIMODAY
INTEGER DEEPEST
REAL LATMIN, LONGMIN, HOUR
C
C*****
C
C
C*****
C
COMMON/INPUTVAR/ DEPTH(140), TEMPTURE(140), SALINITY(140),
+                XLAT, XLONG, NGOOD
DOUBLE PRECISION DEPTH, TEMPTURE, SALINITY, XLAT, XLONG
INTEGER NGOOD
C
C*****
C
COMMON/DATASAV/          SAVDAT2(5,140),
+                NDETAIL, NINT,
+                NSAV1, NSAV2
REAL          SAVDAT2
INTEGER NDETAIL, NINT
INTEGER NSAV1, NSAV2
C
C*****
C
COMMON/LOCAL/ BLAT, BLON,
+                NOUT1, NOUT2, JUNK(1687)
REAL BLAT, BLON
INTEGER NOUT1, NOUT2, JUNK
REAL RJUNK(1687)
EQUIVALENCE (JUNK,RJUNK)
C
C*****
C
C
BLAT = XLAT
BLON = XLONG
C
JUNK(1) = COUNTRY
JUNK(2) = SHIP(1)
JUNK(3) = SHIP(2)
JUNK(4) = CRUISENO
JUNK(5) = STATNO(1)
JUNK(6) = STATNO(2)
JUNK(7) = STATNO(3)
JUNK(8) = DAY
JUNK(9) = MONTH
JUNK(10) = YEAR
RJUNK(11) = HOUR
RJUNK(12) = BLAT
RJUNK(13) = BLON
JUNK(14) = BCTDEP
JUNK(15) = NDETAIL
JUNK(16) = NINT
CALL BUFFEROUT ( SAVFIL, 1, JUNK, 16, INKER )

```

```

JCOC1077
JCOC1078
JCOC1079
JCOC1080
JCOC1081
JCOC1082
JCOC1083
JCOC1084
JCOC1085
JCOC1086
JCOC1087
JCOC1088
JCOC1089
JCOC1090
JCOC1091
JCOC1092
JCOC1093
JCOC1094
JCOC1095
JCOC1096
JCOC1097
JCOC1098
JCOC1099
JCOC1100
JCOC1101
JCOC1102
JCOC1103
JCOC1104
JCOC1105
JCOC1106
JCOC1107
JCOC1108
JCOC1109
JCOC1110
JCOC1111
JCOC1112
JCOC1113
JCOC1114
JCOC1115
JCOC1116
JCOC1117
JCOC1118
JCOC1119
JCOC1120
JCOC1121
JCOC1122
JCOC1123
JCOC1124
JCOC1125
JCOC1126
JCOC1127
JCOC1128
JCOC1129
JCOC1130
JCOC1131
JCOC1132
JCOC1133
JCOC1134
JCOC1135
JCOC1136

```


CCCC	IF (INKER .NE. 2) GO TO 100	JCOC1137
CCCC	NOUT1 = NSAV1 * NDETAIL	JCOC1138
CCCC	CALL BUFFEROUT (SAVFIL, 1, NOUT1, INKER)	JCOC1139
	IF (INKER .NE. 2) GO TO 100	JCOC1140
C		JCOC1141
	NOUT2 = NSAV2 * NINT	JCOC1142
	IF (NINT .GT. 0)	JCOC1143
+	CALL BUFFEROUT (SAVFIL, 1, SAVDAT2, NOUT2, INKER)	JCOC1144
	IF (INKER .NE. 2) GO TO 100	JCOC1145
C		JCOC1146
	RETURN	JCOC1147
C		JCOC1148
C		JCOC1149
	100 CONTINUE	JCOC1150
C		JCOC1151
	WRITE(LP,1000) INKER	JCOC1152
1000	FORMAT(' OUTPUT ERROR ON SAVE FILE. IND. = ',I2)	JCOC1153
	STOP	JCOC1154
C		JCOC1155
C		JCOC1156
	END	JCOC1157
	SUBROUTINE READARRAY (SD, NSD)	JCOC1158
C		JCOC1159
	DOUBLE PRECISION SD(1)	JCOC1160
C		JCOC1161
	NSD = 1	JCOC1162
10	CONTINUE	JCOC1163
	READ(105,1000,END=50) SD(NSD)	JCOC1164
1000	FORMAT(1)	JCOC1165
	NSD = NSD + 1	JCOC1166
	GO TO 10	JCOC1167
C		JCOC1168
C		JCOC1169
	50 CONTINUE	JCOC1170
	NSD = NSD - 1	JCOC1171
	RETURN	JCOC1172
C		JCOC1173
	END	JCOC1174

COPY QCCOMP TO CP
PCL PROCESSING TERMINATED

COPY PICKOUT TO LP

```

C      PROGRAM PICKOUT READS THE FILE OF NCDC DATA CREATED BY
C      PROGRAM CCCMP, FILTERS OUT SIGMA-TS WHICH HAVE BAD TEMP
C      ,SAL OR WHOSE DEPTHS DON'T MATCH STANDARD DEPTHS,
C      THE SIGMA-TS ARE RECORDED ON TAPE OR DISK FILE
C
C      PROGRAMMER: JAMES CHURCHILL W.H.O.I. 12/19/78
C
C      COMMON CC,SC1,SC2,CN,SN1,SN2,SN3,IDA,IMO,IYR,GMT,FLAT,FLONG,
#      WD,ND2,ND3,DATA(180)
C      COMMON CCP,SC1P,SC2P,CNP,SN1P,SN2P,SN3P,IDAP,IMGP,IYRP
&      ,FLATP,FLONP,WDP,ND3P,ST(30)
C      DIMENSION DM(28)
C      INTEGER CC,SC1,SC2,CN,SN1,SN2,SN3,SC1P,SC2P,CNP,
&      SN1P,SN2P,SN3P,CCP,WD,WDP
C      DATA(DM(J),J=1,28)/1.,10.,20.,30.,40.,50.,60.,75.,100.,
#      125.,150.,200.,250.,300.,400.,500.,600.,700.,800.,900.,
%      1000.,1200.,1500.,1750.,2000.,2500.,3000.,4000./
C
C      INPUT NUMBER OF FILES CN INPUT TAPE TO SKIP AND NUMBER
C      TO READ
C
C      READ(105,10) NOFTS,NTODO
C      IF(NCFTS.GT.0) CALL SKPFIL(2,NOFTS)
C      NDCNE = 0
10     FORMAT(212)
C
C      READ MASTER RECORD
C
1     CALL BUFFER IN (2,1,CC,16,ISTAT,NIN,MCNERR)
C      IF(ISTAT.EQ.3) GO TO 1000
C      IF(MCNERR.NE.0) OUTPUT,NIN,MCNERR
C      CCP=CC
C      SC1P = SC1
C      SC2P = SC2
C      CNP = CN
C      SN1P = SN1
C      SN2P = SN2
C      SN3P = SN3
C      IMCP = IMO
C      IDAP = IDA
C      IYRP = IYR
C      FLATP = FLAT
C      FLONP = FLONG
C      WDP = WD
C      ND3P = ND3
C
C      READ RECORD OF INTERPOLATED VALUES
C
C      IF(ND3.EQ.0)WRITE(108,90);GO TO 1
C      IF(ND3.GT.28) WRITE(108,91); GO TO 1
C      CALL BUFFER IN(2,1,DATA,ND3*5,ISTAT)
C      IF(ISTAT.EQ.3) GO TO 1000
C      IF(ISTAT.EQ.4) WRITE(108,70)
C
C      EXTRACT SIGMA-TS, SET TO ZERO SIGMA-TS WHICH HAVE BAD
C      TEMP,SAL, OR DEPTH
C

```



```

      NSK=0
      DE1=DATA(1)
      IF (DM(1).EQ.DE1) GO TO 45
      ST(1)=0.
      NSK=1
      DO 59 I=2,28
      IF (DM(I).EQ.DE1) GO TO 45
      ST(I) = 0.
      NSK= I
59    CONTINUE
      WRITE(108,92)DE1
      GO TO 1
45    NDL = C
      DO 100 I=1,ND3
      ST(I+NSK)=DATA((I-1)*5+4)
      DE=DATA((I-1)*5+1)
      TEMP=DATA((I-1)*5+2)
      SAL=DATA((I-1)*5+3)
      IF (DE.NE.DM(I+NSK)) WRITE(108,80)DM(I+NSK),DE ; GO TO 95
      IF (TEMP.GT.40..OR.TEMP.LT.-2.) GO TO 95
      IF (SAL.GT.39..OR.SAL.LT.26.) GO TO 95
      GO TO 100
95    NDL = NDL+1
      ST(I+NSK)=0.
100   CCNTINUE
      ND3P = ND3P+NSK
C
C    WRITE MASTER RECORD AND SIGMA-TS ON DEVICE 1
C
      CALL BUFFER OUT(1,1,CCP,14,ISTAT,N)
      CALL BUFFER OUT(1,1,ST,ND3P,ISTAT2,N2)
      NCC=ND3-NDL
      IF (ISTAT.EQ.2.AND.ISTAT2.EQ.2)
& WRITE(108,50)SC1P,SC2P,CNP,SN1P,SN2P,SN3P,IMOP,1DAP,
& IYRP,FLATP,FLCNP,NDO,NDL
      OUTPLT WD,WDP
      GO TO 1
50    FORMAT(' CRUISE: ',3A4,' STATION: ',3A4,3I3,3X,
& 'LAT:',F7.2,' LCNG:',F7.2,3X,14,' SIGMATS LOADED',14,' REJECTED')
70    FORMAT(' INPUT ERROR')
80    FORMAT(' DEPTH',F9.1,' MISTAKENLY RECORDED AS',F9.1)
90    FORMAT(' NO INTERPOLATED VALUES,STATION SKIPPED')
91    FORMAT(' TOO MANY DEPTHS,STATION SKIPPED')
92    FORMAT(' FIRST DEPTH',F10.2,'CANNOT BE FOUND,STATION SKIPPED')
1000  NDCNE = NDONE+1
      IF (NDCNE.LT.NTODC) GO TO 1
      END FILE 1
      CALL EXIT
      END

```

WEOF LP

COPY GRID TO LP.

```

C      PROGRAM GRID
C      PROGRAM GRID AVERAGES SIGMATS AT STANDARD LEVELS
C      WITHIN A GEOGRAPHIC ARRAY OF GRID BOXES.  THE DATA IS
C      FROM NODC, PROCESSED BY PROGRAMS OCCOMP AND PICKOUT.
C      DENSITY INVERSIONS IN THE AVERAGED SIGMATS ARE FLAGGED
C      AND CORRECTED.  THERE IS AN OPTION TO FILTER DATA > 5 STD
C      FROM THE AVERAGE.  ANOTHER OPTION CALLS FOR THE
C      CALCULATION OF THE FOLLOWING INTEGRALS:
C      FIE = INTEGRAL(ETA*DZ) ; LIMITS 0 TO H
C      FIZ = INTEGRAL(ETA*Z/H *DZ) ; LIMITS 0 TO H
C      WHERE:  H = BOTTOM DEPTH
C              ETA = (DEN-SO)/SO ; SO = ABYSSAL DENSITY
C      THE AVERAGE, STANDARD DEVIATION, AND # OF OBSERVATIONS
C      FOR EACH GRID ELEMENT ARE OUTPUT TO BINARY FILE.
C      SUBROUTINES REQUIRED: STD,INTG
C
C      PROGRAMMER: JAMES CHURCHILL, W.H.O.I. 12/15/78
C
C      COMMON /IN/CC,SC1,SC2,CN,SN1,SN2,SN3,IDA,IMC,IYR,FLAT,FLONG,
6      WD,ND,ST(30)
C      COMMON NGRID,IOPTI,FLATC(270),FLONC(270),BOTE(270),
6      AST(28,270),NST(28,270),SDST(28,270),NOMAX(270),
6      BOTDEP(270),FIE(270),FIZ(270)
C      DIMENSION DM(28),DLIB(4),ELIB(4)
C      INTEGER CC,SC1,SC2,CN,SN1,SN2,SN3,WD
C      DATA(DM(J),J=1,28)/1.,10.,20.,30.,40.,50.,60.,75.,100.,125.,
6      150.,200.,250.,300.,400.,500.,600.,700.,800.,900.,1000.,
6      1200.,1500.,1750.,2000.,2500.,3000.,4000./
C
C      INPUT PROCESSING OPTIONS:
C      IF IOPTI.NE.0 DO NOT DO DEPTH INTEGRATIONS
C      IF IOPSD.NE.0 DELETE DATA > 5 STD FROM THE AVERAGE
C
C      READ(105,6) IOPTI,IOPTSD
6      FORMAT(2I2)
C
C      INPUT ABYSSAL DENSITY AND LIBRARY SIGMATS TO FILL IN
C      MISSING DATA.
C      CALCULATE ETAS FROM THE SIGMATS
C
C      READ(105,7) NLIB,SO
C      READ(105,8)(DLIB(I),ELIB(I),I=1,NLIB)
7      FORMAT(I2,F7.4)
8      FORMAT(F6.0,F6.2)
C      WRITE(108,790) SC
C      DO 10 I=1,NLIB
C      WRITE(108,795) DLIB(I),ELIB(I)
C      DENI = 1. + ELIB(I)/1000.
C      ELIB(I) = (DENI-SC)/SO
10     CONTINUE
C
C      INPUT DIMENSIONS OF GRID
C
C      READ(5,11) NGRID,DLAT,DLOH
11     FORMAT(I3,2F5.2)
C      READ(5,20)(FLATC(I),FLONC(I),BOTDEP(I),I=1,NGRID)
20     FORMAT(2F10.2,F10.0)

```



```

C
C      INITIALIZE
C
      DO 75 IGR = 1,NGRID
      NDMAX(IGR) = 0
      DC 50 I=1,28
      ASI(I,IGR)= SDST(I,IGR)=0.
      NST(I,IGR)=0
50     CONTINUE
75     CONTINUE

C
C      READ IN STATION DATA STATION BY STATION
C
5     CALL BUFFER IN(1,1,CC,14,ISTAT)
      IF(ISTAT.EQ.3) GO TO 200
      FLCNG = -FLCNG
      CALL BUFFER IN(1,1,ST,ND,ISTAT)
      IF(ISTAT.EQ.3) GO TO 200

C
C      ACCUMULATE DATA WITHIN GRID BOXES
C
      DO 100 IGR = 1,NGRID
      IF(FLAT.LT.(FLATC(IGR)-DLAT).OR.FLAT.GT.(FLATC(IGR)
& +DLAT)) GO TO 100
      IF(FLONG.LT.(FLONG(IGR)-DLON).OR.FLCNG.GT.
& (FLONG(IGR)+DLON)) GO TO 100
      IF(ND.GT.NDMAX(IGR)) NDMAX(IGR)=ND
      DO 150 I=1,ND
      IF(ST(I).LE.0) GO TO 150
      AST(I,IGR) = AST(I,IGR) + ST(I)
      NST(I,IGR) = NST(I,IGR) + 1
      SDST(I,IGR) = SDST(I,IGR)+ST(I)*ST(I)
150    CONTINUE
100    CONTINUE
      GO TO 5

C
C      CALCULATE AVERAGES AND STANDARD DEVIATIONS WITHIN EACH
C      GRID BOX
C
200    DC 300 IGR = 1,NGRID
      DO 250 I=1,NDMAX(IGR)
      IF(NST(I,IGR).EQ.0) GO TO 250
      FN = NST(I,IGR)
      AST(I,IGR) = AST(I,IGR)/FN
      SUN = SDST(I,IGR)-FN*AST(I,IGR)*AST(I,IGR)
      IF(SUN.LE.0.) SDST(I,IGR) = 0.; GO TO 250
      SDST(I,IGR) = SQRT(SUN/(FN-1.))
250    CONTINUE
300    CONTINUE

C
C      IF IOPTSD>0 DELETE DATA GREATER THAN 5 STD FROM THE AVERAGE
C      (BOTH THE STD AND THE AVERAGE ARE WITHOUT THE DATUM BEING
C      CHECKED)
C
      IF(ICPTSD.GT.0) WRITE(108,780);CALL STD(AST,NST,SDST,FLONG,
& FLATC,DM,DLCN,DLAT,NGRID)

C
C      FOR EACH GRID OUTPUT THE AVERAGE FOR ALL LEVELS
C
      DO 400 IGR = 1,NGRID

```

```

WRITE(108,800) FLNC(IGR),FLATC(IGR),DLGN,DLAT
WRITE(108,810)
DO 350 I=1,NDMAX(IGR)
IF(NST(I,IGR).EQ.0) GO TO 350
IF(I.EQ.1) GO TO 225

C
C CHECK FOR DENSITY INVERSION. IF PRESENT SET INVERTED
C VALUE EQUAL TO THE VALUE ABOVE AND FLAG WITH A -SIGN.
C
DIFF = AST(I,IGR) - ABS(AST(I-1,IGR))
IF(DIFF.LT.0) WRITE(108,830)-DIFF;AST(I,IGR)=-AST(I-1,IGR)
225 WRITE(108,820)DM(I),NST(I,IGR),ABS(AST(I,IGR)),SDST(I,IGR)
350 CONTINUE

C
C DO DEPTH INTEGRATION IF IOPT = 0
C
IF(IOPTI.GT.0) GO TO 400
CALL INTG(BOTDEP(IGR),NDMAX(IGR),AST(1,IGR),
& FIE(IGR),FIZ(IGR),BOTE(IGR),SO,DLIB(1),ELIB(1))
WRITE(108,840) -FIE(IGR),-BOTDEP(IGR),FIZ(IGR),-BCTDEP(IGR)
WRITE(108,850) BOTE(IGR)
400 CONTINUE
780 FORMAT(///40X'VALUES EXCEEDING 5STD DEV OF THE AVERAGES'//
& 4X,'LAT',3X,'LONG',3X,'DEPTH',6X,'CRUISE',5X,
& 'VALUE',4X,'OLD VALUES',4X,'NEW VALUES'/40X,'DEL',3X,
& 'NUM',3X,'AVE',3X,'STDEV',2X,'NUM',3X,'AVE',4X,'STDEV'//)
790 FORMAT('1ABBYSSAL DENSITY =',F8.4//
& ' LIBRARY DENSITIES FOR GRIDS DEEPER THAN AVAILABLE DATA'
& //7X,'DEPTH',5X,'DENSITY')
795 FORMAT(F12.0,F12.2)
800 FORMAT(///// ' AVERAGED SIGMA-TS FOR GRID',F6.2,'DEG W',
& F6.2,'DEG N'/20X,'SIZE:',F5.2,'DEG LONG X ',F5.2,
& 'DEG LAT'/)
810 FORMAT(' DEPTH',5X,'NUMBER',8X,'AVE',6X,'ST DEV'/11X,
& 'OF CBS',7X,'SIGT')
820 FORMAT(F6.0,I9,F13.2,F11.2)
830 FORMAT(' *****DENSITY INVERSION OF',F7.2)
840 FORMAT(' 17X,'0'/' -INTEGRAL(ETA*DZ) =',F8.4/
& 16X,F6.0/21X,'0'/' INTEGRAL(ETA*Z/H *DZ) =',F8.4/20X,F6.0)
850 FCRMAT(' BOTTOM ETA =',F8.5)

C
C OUTPUT DATA TO DISK OR TAPE
C
CALL BUFFER OUT(2,1,NGRID,2,ISTAT)
CALL BUFFER OUT(2,1,FLATC,NGRID,ISTAT)
CALL BUFFER OUT(2,1,FLNC,NGRID,ISTAT)
CALL BUFFER OUT(2,1,BOTDEP,NGRID,ISTAT)
CALL BUFFER OUT(2,1,AST,NGRID*28,ISTAT)
CALL BUFFER OUT(2,1,NST,NGRID*28,ISTAT)
CALL BUFFER OUT(2,1,SDST,NGRID*28,ISTAT)
IF(ICPTI.GT.0) GO TO 1000
CALL BUFFER OUT(2,1,FIE,NGRID,ISTAT)
CALL BUFFER OUT(2,1,FIZ,NGRID,ISTAT)
CALL BUFFER OUT(2,1,BOTE,NGRID,ISTAT)
1000 END FILE 2
CALL EXIT
END

```

WEOF LP

COPY STD TO LP

```

SUBROUTINE STD(AST,NST,SDST,FLCNC,FLATC,DM,DLCN,DLAT,NGRID)
C
C   SUBROUTINE STD RECEIVES FORM THE CALLING PROGRAM
C   AVERAGE SIGMA-TS, STANDARD DEVIATIONS AND # OF OBSERVATIONS
C   FOR EACH ELEMENT OF A GEOGRAPHIC GRID. THE DEVICE
C   CONTAINING THE STATION DATA IS REWIND AND REREAD.
C   FOR EACH ELEMENT ALL VALUES GREATER THAN 5 STANDARD
C   DEVIATIONS FROM THE AVERAGE ARE DELETED. THE CORRECTED
C   AVERAGE, STANDARD DEVIATION AND # OF OBSERVATIONS ARE
C   RETURNED.
C
C   PROGRAMMER: J. GURCHILL, W.H.O.I. 12/22/78
C
COMMON /IN/CC,SC1,SC2,CN,SN1,SN2,SN3,IDA,IMO,IYR,FLAT,FLCNG,
& WD,ND,ST(30)
DIMENSION AST(28,270),SDST(28,270),NST(28,270),FLCNC(270),
& FLATC(270),DM(28)
INTEGER CC,SC1,SC2,CN,SN1,SN2,SN3,WD
REWIND 1
C
C   INPUT DATA STATION BY STATION
C
5 CALL BUFFER IN(1,1,CC,14,ISTAT)
IF(ISTAT.EQ.3) GO TO 200
FLCNG = -FLCNG
CALL BUFFER IN(1,1,ST,ND,ISTAT)
IF(ISTAT.EQ.3) GO TO 200
C
C   FIND THE GRIDS IN WHICH THE CONSIDERED STATION LIES
C
DO 100 IGR = 1,NGRID
IF(FLAT.LT.(FLATC(IGR)-DLAT).OR.FLAT.GT.(FLATC(IGR)
& + DLAT)) GO TO 100
IF(FLCNG.LT.(FLCNC(IGR)-DLON).OR.FLCNG.GT.
& (FLCNC(IGR)+DLON)) GO TO 100
C
C   CONSIDER EACH LEVEL EXCEPT IF THE # OF OBSERVATIONS IS
C   .LE.3 OR IF THE ST. DEV. IS .LE.0.02
C
DO 150 I=1,ND
IF(SDST(I,IGR).LE..02) GO TO 150
IF(NST(I,IGR).LE.3) GO TO 150
IF(ST(I).LE.0.) GO TO 150
FN = NST(I,IGR)
C
C   CALCULATE THE AVERAGE AND THE STD. DEV. WITH THE CONSIDERED
C   STATION DELETED (ANEW,SDNEW).
C
ANEW = (AST(I,IGR)*FN-ST(I))/(FN-1.)
SDNEW = (FN-1.)*(SDST(I,IGR)*SDST(I,IGR)-ANEW*ANEW)
SDNEW = SDNEW + FN*AST(I,IGR)*AST(I,IGR)-ST(I)*ST(I)
IF(SDNEW.LE.0.) SDNEW = .001
SDNEW = SQRT(SDNEW/(FN-2.))
C
C   IF THE CONSIDERED SIGMA-T IS 5*SDNEW FROM ANEW, REPLACE
C   THE AVERAGE BY ANEW AND THE STANDARD DEVIATION BY SDNEW
C   AND DECREASE THE # OF OBS. BY 1

```

C

```

      IF (ST(I).LT.ANEW-5.*SDNEW.OR.ST(I).GT.ANEW+5.*SDNEW)GO TO 130
      GO TO 150
130   WRITE(108,800)FLATC(IGR),FLONC(IGR),DM(I),SC1,SC2,CN,ST(I),
      & NST(I,IGR),AST(I,IGR),SDST(I,IGR),NST(I,IGR)-1,ANEW,SDNEW
      SDST(I,IGR) = SDNEW
      AST(I,IGR) = ANEW
      NST(I,IGR) = NST(I,IGR)-1
150   CONTINUE
100   CONTINUE
      GO TO 5
800   FORMAT(2F7.2,F8.0,3X,3A4,F7.2,I5,2F7.2,I5,2F7.2)
200   RETURN
      END

```

WEOF LP

COPY INTG TO LP

```

C      SUBROUTINE INTG
C      SUBROUTINE INTG RECEIVES SIGMATS AT STANDARD LEVELS FROM
C      THE CALLING PROGRAM.  THE FOLLOWING INTEGRALS ARE
C      RETURNED:
C          FIE = INTEGRAL(ETA*DZ) ; LIMITS = 0 TO H
C          FIZ = INTEGRAL(ETA*Z/H *DZ) ; LIMITS 0 TO H
C      WHERE:  ETA = (DEN-S0)/S0 ; S0 = ABYSAL DENSITY
C              H = BOTTOM DEPTH
C      THE VALUE OF ETA AT THE BOTTOM IS ALSO RETURNED.
C      IF THE AVAILABLE DATA DOES NOT EXTEND TO THE
C      BOTTOM, THEN A LIBRARY OF STANDARD VALUES IS USED.
C
C      PROGRAMMER: JAMES CHURCHILL, W.H.O.I.  12/17/78
C
C      SUBROUTINE INTG(BOTDEP,NDMAX,AST,FIE,FIZ,BOTE,S0,DLIB,ELIB)
C      DIMENSION DM(28),AST(28),Z(28),DI(28),E(28),ELIB(4),DLIB(4)
C      DATA(DM(J),J=1,28)/0.,10.,20.,30.,40.,50.,60.,75.,100.,125.,
& 150.,200.,250.,300.,400.,500.,600.,700.,800.,900.,1000.,
& 1200.,1500.,1750.,2000.,2500.,3000.,4000./
C      NDMA = NDMAX
C
C      CALCULATE THE ETA AND ETA*DEPTH FROM THE SIGMATS
C
C      DO 100 I=1,NDMA
C          IF(AST(I).EQ.0.) NDMAX = I-1; GO TO 101
C          DEN1 = 1. + ABS(AST(I))/1000.
C          E(I) = (DEN1-S0)/S0
C          IF(DM(I).GE.BOTDEP) D2 = DM(I); GO TO 200
C          Z(I) = E(I)*DM(I)
C          DI(I) = DM(I)
100      CONTINUE
C
C      IF THE BOTTOM DEPTH IS DEEPER THAN AVAILABLE DATA USE
C      THE LIBRARY VALUES
C
101      WRITE(108,800)BOTDEP,DM(NDMAX)
C          I=NDMAX+1
C          DO 150 J=1,4
C              IF(DLIB(J).LT.BOTDEP) GO TO 175
C              E(I) = ELIB(J)
C              D2 = DLIB(J)
C              GO TO 200
175      IF(DLIB(J).LE.DM(NDMAX)) GO TO 150
C          E(I) = ELIB(J)
C          DI(I) = DLIB(J)
C          Z(I) = E(I)*DI(I)
C          I=I+1
150      CONTINUE
C
C      INTERPOLATE ETA TO THE BOTTOM DEPTH
C
200      DI(I) = BOTDEP
C          D1 = DI(I-1)
C          E(I) = E(I-1)+(DI(I)-D1)*(E(I)-E(I-1))/(D2-D1)
C          BOTE = E(I)
C          Z(I) = E(I)*DI(I)
C          NBOD = I

```

```

C
C      CALCULATE TRAPESICD INTEGRALS
C
      DO 300 I=2,NBOD
      EA = (E(I)+E(I-1))/2.
      ZA = (Z(I)+Z(I-1))/2.
      FIE = FIE + EA*(DI(I)-DI(I-1))
      FIZ = FIZ + ZA*(CI(I)-CI(I-1))
300    CONTINUE
      FIZ = -FIZ/BOTDEP
800    FORMAT( ' BOTTOM DEPTH OF',F6.0,' IS GREATER THAN DEEPEST DATA OF '
&,F6.0)
      RETURN
      END

```

WEOF LP

COPY TYGRID TO LP

```

C      PROGRAM TYGRID TYPES OUT SIGMATS, STANDARD DEVIATIONS OR
C      NUMBER OF OBSERVATIONS ON A GEOGRAPHIC GRID
C
C      PROGRAMMER: J. CHURCHILL W.H.C.I. 12/22/78
C
      COMMON NGRID, IOPT, FLATC(270), FLONC(270), AST(28,270), NST(28,270)
& , SDST(28,270), FIE(270), FIZ(270), BOTDEP(270), BCTE(270)
      DIMENSION IAA(270), IAP(270), DM(28), JTITLE(20)
      DATA (DM(J), J=1,28)/1.,10.,20.,30.,40.,50.,60.,75.,100.,125.,
& 150.,200.,250.,300.,400.,500.,600.,700.,800.,900.,1000.,
& 1200.,1500.,1750.,2000.,2500.,3000.,4000./
C
C      INPUT DIMENSIONS AND INCREMENTS OF OUTPUT GRID
C
      READ(105,5) FLONMIN, FLONMAX, DLON
      READ(105,5) FLATMIN, FLATMAX, DLAT
      FORMAT(3F6.2)
      NLAT = (FLATMAX-FLATMIN)/DLAT + 1
      READ(105,6) (JTITLE(I), I=1,20)
      FORMAT(20A4)
      NLON = (FLONMAX-FLONMIN)/DLAT + 1
C
C      INPUT GRID OF INPUT DATA AND DATA
C
      CALL BUFFER IN(2,1,NGRID,2,ISTAT)
      CALL BUFFER IN(2,1,FLATC,NGRID,ISTAT)
      CALL BUFFER IN(2,1,FLONC,NGRID,ISTAT)
      CALL BUFFER IN(2,1,BOTDEP,NGRID,ISTAT)
      CALL BUFFER IN(2,1,AST,NGRID*28,ISTAT)
      CALL BUFFER IN(2,1,NST,NGRID*28,ISTAT)
      CALL BUFFER IN(2,1,SDST,NGRID*28,ISTAT)
      IF(IOPT.GT.0) GO TO 7
      CALL BUFFER IN(2,1,FIE,NGRID,ISTAT)
      CALL BUFFER IN(2,1,FIZ,NGRID,ISTAT)
      CALL BUFFER IN(2,1,BOTE,NGRID,ISTAT)
C
C      INPUT DESIRED LEVEL AND, AND DESIRED OUTPUT
C      NWANT = : 1 FOR SIGMATS, 2 FOR ST DEV, 3 FOR # OF OBS
C      4 FOR INTEGRAL(ETA)DZ, 5 FOR INTEGRAL(ETA*Z)DZ,
C      6 FOR BOTTOM DEPTH, 7 FOR GRID NUMBERS, 8 FOR BOTTOM DENSITY
C
      NDEP = 0
      READ(105,10,END=1000)NDEP,NWANT
      FORMAT(2I2)
      IF(NDEP.GT.28) GO TO 1000
C
C      TRANSFORM DATA TO DESIRED FORM FOR PRINTOUT
C      AND PRINT HEADING
C
      GO TO (100,110,120,160,180,195,198,201) NWANT
100  DO 130 I=1,NGRID
      IAA(I) = (AST(NDEP,I)-SIGN(20.,AST(NDEP,I)))*100. +
& SIGN(.5,AST(NDEP,I))
130  CONTINUE
      WRITE(108,800) (JTITLE(I), I=1,20)
      GO TO 200
110  DO 140 I=1,NGRID

```

```

      IAA(I) = SDST(NDEP,I)*100. + .5
140  CONTINUE
      WRITE(108,810) (JTITLE(I),I=1,20)
      GO TO 200
120  DO 150 I=1,NGRID
      IAA(I) = NST(NDEP,I)
150  CONTINUE
      WRITE(108,820) (JTITLE(I),I=1,20)
      GO TO 200
160  DO 170 I=1,NGRID
      IAA(I) = -FIE(I)*100. + .5
170  CONTINUE
      WRITE(108,822) (JTITLE(I),I=1,20)
      GO TO 210
180  DO 190 I=1,NGRID
      IAA(I) = -FIZ(I)*100. + 0.50
190  CONTINUE
      WRITE(108,824) (JTITLE(I),I=1,20)
      GO TO 210
195  DO 197 I=1,NGRID
      IAA(I) = BCTDEP(I)/10.
197  CONTINUE
      WRITE(108,826) (JTITLE(I),I=1,20)
      GO TO 210
198  DO 199 I=1,NGRID
      IAA(I) = I
199  CONTINUE
      WRITE(108,827) (JTITLE(I),I=1,20)
      GO TO 210
201  DO 202 I=1,NGRID
      ST = (((BCTE(I)+1.)*1.028)-1.)*1000.
      IAA(I) = (ST-20.)*100. + .50
202  CONTINUE
      WRITE(108,828) (JTITLE(I),I=1,20)
      GO TO 210
200  WRITE(108,830) DM(NDEP)
210  WRITE(108,840) FLCNMAX,FLATMAX
C
C      TYPE OUT DATA LINE BY LINE
C
      FLAT = FLATMAX
      DO 400 I=1,NLAT
      FLCN = FLCNMMAX
      DO 300 J=1,NLCN
      DO 250 II=1,NGRID
      IF(FLAT.EQ.FLATC(II).AND.FLCN.EQ.FLONC(II)) GO TO 270
250  CONTINUE
      IAP(J) = 0
      GO TO 280
270  IAP(J) = IAA(I)
280  FLCN = FLCN-DLCN
300  CONTINUE
      IF(FLAT.LE.FLATMIN)WRITE(108,855)NLCN, (IAP(IP),IP=1,NLCN)
      & : GO TO 400
      WRITE(108,850) NLCN, (IAP(IP),IP=1,NLCN)
      FLAT = FLAT-DLAT
400  CONTINUE
      WRITE(108,860)NLCN*4-2,NLCN*4-12,FLCNMIN,FLATMIN
      GO TO 8
800  FORMAT('1',20A4/50X,'SIGMATS')

```



```
810  FCRMAT('1',20A4/48X,'STD.DEVIATIONS')
820  FORMAT('1',20A4/45X,'NO OF OBSERVATIONS')
822  FORMAT('1',20A4/41X,'-INTEGRAL(ETA*CZ) -CM')
824  FORMAT('1',20A4/40X,'INTEGRAL(ETA*Z/H *CZ) -CM')
826  FORMAT('1',20A4/47X,'BOTTOM DEPTH/10')
827  FORMAT('1',20A4/47X,'GRID NUMBERS')
828  FCRMAT('1',20A4/44X,'BOTTOM DENSITIES')
830  FORMAT(43X,'FOR DEPTH',F6.0)
840  FORMAT(1X,F6.2,' ',F6.2/' ' +')
850  FCRMAT(NI4/)
855  FORMAT(NI4)
860  FORMAT(NX,'+' /NX,F6.2,' ',F6.2)
1000 CALL EXIT
      END
```

WEOF LP

COPY GRPRS TO LP

```

C      PROGRAM GRPRS
C      PROGRAM GRPRS READS A BINARY FILE CONTAINING A GRID
C      OF AVERAGED SIGMATS AT STANDARD LEVELS COMPUTED BY
C      PROGRAM GRID. THE SEA SURFACE LEVEL CALCULATIONS
C      ARE DONE AS FOLLOWS:
C          JD = -INTEGRAL(ETA*DZ) ; LIMITS = 0 TO -H
C          JM = INTEGRAL(ETA*Z/H *DZ) ; LIMITS = 0 TO -H
C          JH = INTEGRAL(JM/H *DH) ; LIMITS = -H TO HMAX
C          JB = -INTEGRAL(BOTETA*DH) ; LIMITS = -H TO HMAX
C      WHERE: H=BOTTOM DEPTH ; ETA=(DEN-ABYSALL DEN)/ABYSALL DEN
C          JB AND JH ARE SPECIFIED AS PIECEWISE LINEAR FUNCTIONS
C          OF BOTTOM DEPTH
C
C          ELEV.1 = JD - JM + JH
C          ELEV.2 = JD + JB
C
C      SUBROUTINES REQUIRED: INTG,TYGR,SKPREC(WHOL.3)
C
C      PROGRAMMER: JAMES CHURCHILL, W.H.O.I. 12/15/78
C
C      COMMON NGRID IOPT,FLATC(270),FLONC(270),BOTDEP(270),
& AST(28,270),FIE(270),FIZ(270),BOTE(270),NDMAX(270)
C      DIMENSION F1(270),FIEM(270),NPT(10),DLIB(4),ELIB(4),
& JTITLE(20),D(25),FJH(25),FJB(25)
C
C      INPUT OPTIONS: IOPTI = 0 DO NOT DO INTEGRATIONS;
C      IOPTY = 0 DO NOT TYPE OUTPUT GRID
C
C      READ(105,1) IOPTI,IOPTY
1      FORMAT(2I2)
C
C      INPUT ABBYSAL DENSITY AND LIBRARY DENSITIES
C
C      READ(105,7) NLIB,SO
C      READ(105,8)(DLIB(I),ELIB(I),I=1,NLIB)
7      FORMAT(12,F7.4)
8      FORMAT(F6.0,F6.2)
C      WRITE(108,790) SO
C      DO 10 I=1,NLIB
C      WRITE(108,795) DLIB(I),ELIB(I)
C      DENI = 1. + ELIB(I)/1000.
C      ELIB(I) = (DENI-SO)/SO
10     CONTINUE
C
C      INPUT DIMENSIONS AND INCREMENTS OF OUTPUT GRID
C
C      READ(105,5) FLONMIN,FLONMAX,DLON
C      READ(105,5) FLATMIN,FLATMAX,DLAT
5      FORMAT(3F6.2)
C      NLAT = (FLATMAX-FLATMIN)/DLAT + 1
C      NLON = (FLONMAX-FLONMIN)/DLON + 1
C
C      INPUT TITLE FOR OUTPUT GRID
C
C      READ(105,9)(JTITLE(I),I=1,20)
9      FORMAT(20A4)
C

```



```

C      INPUT NUMBER OF DEPTHS AT WHICH VALUES OF LINE INTEGRALS
C      ARE GIVEN
C
11     READ(105,11) NDEP
        FORMAT(I2)
C
C      INPUT VALUES OF DEPTH, JH AND JB IN DECREASING DEPTH ORDER
C
12     READ(105,12)(D(J),FJH(J),FJB(J),J=1,NDEP)
        FORMAT(F5.0,2F10.4)
C
C      INPUT GRID OF INPUT DATA AND DATA
C
        CALL BUFFER IN(2,1,NGRID,2,ISTAT)
        CALL BUFFER IN(2,1,FLATC,NGRID,ISTAT)
        CALL BUFFER IN(2,1,FLONC,NGRID,ISTAT)
        CALL BUFFER IN(2,1,BOTDEP,NGRID,ISTAT)
        CALL BUFFER IN(2,1,AST,NGRID*28,ISTAT)
        IF(ICPT.GT.0) GO TO 100
        IF(IQPTI.GT.0) GO TO 100
        CALL SKPREC(2,2)
        CALL BUFFER IN(2,1,FIE,NGRID,ISTAT)
        CALL BUFFER IN(2,1,FIZ,NGRID,ISTAT)
        CALL BUFFER IN(2,1,BOTE,NGRID,ISTAT)
C
        DO 99 I=1,NGRID
          FIZ(I) = -FIZ(I)
99     CONTINUE
C
        GO TO 200
C
C      IF DESIRED COMPUTE DEPTH INTEGRALS, JC AND JM
C
100    DO 150 IGR = 1,NGRID
C
        NDMAX(IGR) = 28
C
        CALL INTG(BOTDEP(IGR),NDMAX(IGR),AST(1,IGR),FIE(IGR),
8      FIZ(IGR),BOTE(IGR),SO,DLIB(1),ELIB(1))
150    CONTINUE
200    WRITE(108,800)
        WRITE(108,805)(D(J),FJH(J)*100.,FJB(J)*100.,J=1,NDEP)
        WRITE(108,807)
C
C      FOR EACH GRID INTERPOLATE JB AND JH TO THE BOTTOM DEPTH
C      AND CALCULATE THE SURFACE ELEVATION FUNCTIONS
C
        DO 250 IGR = 1,NGRID
          ZH = BOTDEP(IGR)
          DO 225 I=2,NDEP
            IF(D(I).LE.ZH) GO TO 230
225    CONTINUE
          I=NDEP
230    FLF=FJH(I)+(FJH(I)-FJH(I-1))*(ZH-D(I))/(D(I)-D(I-1))
          FLE=FJB(I)+(FJB(I)-FJB(I-1))*(ZH-D(I))/(D(I)-D(I-1))
          F1(IGR) = -FIE(IGR) - FIZ(IGR) + FLF
          FIEM(IGR) = -FIE(IGR) + FLE
          WRITE(108,810) IGR,FLATC(IGR),FLONC(IGR),BOTDEP(IGR),
8      -FIE(IGR),FIZ(IGR),BOTE(IGR),F1(IGR),FIEM(IGR)
250    CONTINUE

```

```

C
C      IF DESIRED PRINT RESULTS ON A GEOGRAPHIC GRID
C
      IF(10PTY.EQ.0) GO TO 1000
      DO 510 I=1,NGRID
      FIE(I) = -FIE(I)*100.
      FIZ(I) = FIZ(I)*100.
      FI(I) = FI(I)*100.
      FIEM(I) = FIEM(I)*100.
      BOTDEP(I) = BOTDEP(I)/10.
510  CONTINUE
      WRITE(108,850)(JTITLE(I),I=1,20)
      CALL TYGR(FLATMIN,FLATMAX,FLCNMIN,FLCNMAX,NLAT,NLON,
&  DLON,DLAT,NGRID,FLATC,FLONC,BOTDEP)
      WRITE(108,860)(JTITLE(I),I=1,20)
      CALL TYGR(FLATMIN,FLATMAX,FLCNMIN,FLCNMAX,NLAT,NLON,
&  DLON,DLAT,NGRID,FLATC,FLONC,FIE)
      WRITE(108,870)(JTITLE(I),I=1,20)
      CALL TYGR(FLATMIN,FLATMAX,FLCNMIN,FLCNMAX,NLAT,NLON,
&  DLON,DLAT,NGRID,FLATC,FLONC,FIZ)
      WRITE(108,900)(JTITLE(I),I=1,20)
      CALL TYGR(FLATMIN,FLATMAX,FLCNMIN,FLCNMAX,NLAT,NLON,
&  DLON,DLAT,NGRID,FLATC,FLONC,FI)
      WRITE(108,910)(JTITLE(I),I=1,20)
      CALL TYGR(FLATMIN,FLATMAX,FLCNMIN,FLCNMAX,NLAT,NLON,
&  DLON,DLAT,NGRID,FLATC,FLONC,FIEM)
790  FORMAT('1ABBYSSAL DENSITY =',F8.4//
&  ' LIBRARY DENSITIES FOR GRIDS DEEPER THAN AVAILABLE DATA'
&  //7X,'DEPTH',5X,'DENSITY')
795  FORMAT(F12.0,F12.2)
800  FORMAT('1',40X,' INTEGRALS'//' KEY:')
&  ' JD = -INTEGRAL(ETA*DZ) ; LIMITS = 0 TO -H'/
&  ' JM = INTEGRAL(ETA*Z/H *DZ) ; LIMITS = 0 TO -H'/
&  ' JH = INTEGRAL(JM/H *DH) ; LIMITS = -H TO -HMAX'/
&  ' JB = -INTEGRAL(BOTETA*DH) ; LIMITS = -H TO -HMAX'/
&  ' PIECEWISE LINEAR FUNCTIONS WITH ENDPPOINTS:'/
&  ' DEPTH',8X,' JH',8X,' JB'/3X,'M',10X,'CM',8X,'CM')
805  FORMAT(F6.0,2F10.2)
807  FORMAT(/' J1 = JD-JM+JH'/
&  ' JDM = JD+JB'//
&  ' GR NC',7X,'LAT',6X,'LONG',3X,'BOTDEP',8X,'JD',
&  8X,'JM',8X,'BOTE',8X,'J1',7X,'JDM')
810  FORMAT(I4,3X,F7.2,3X,F7.2,4X,F6.0,2F10.4,E12.3,2F10.4)
850  FORMAT('1',20A4/47X,'BOTTOM DEPTH/10')
860  FORMAT('1',20A4/44X,'-INTEGRAL(ETA*DZ) -CM')
870  FORMAT('1',20A4/42X,'INTEGRAL(ETA*Z/H *DZ) -CM')
900  FORMAT('1',20A4/32X,
&  'INT(JM/H*DH)-INT(ETA*DZ)-INT(ETA*Z/H*DH) -CM')
910  FORMAT('1',20A4/40X,'-INT(BOTE*DH) -INT(ETA*DZ) -CM')
1000 CALL EXIT
      END

```

WEOF LP

COPY TYGRS TO LP

```

C      SUBRCUTINE TYGR
C      SUBRCUTINE TYGR TYPES VALUES FROM THE CALLING PROGRMA ON
C      A GECGRAPHIC GRID
C
C      PROGRAMMER J. CHURCHILL, W.H.O.I.    12/19/78
C
      SUBRCUTINE TYGR(FLATMIN,FLATMAX,FLONMIN,FLONMAX,NLAT,
& NLCN,DLON,DLAT,NGRID,FLATC,FLCNC,FNU)
      DIMENSION FLATC(270),FLCNC(270),FNU(270),IAA(270),IAP(270)
      DO 10 I=1,NGRID
        IAA(I) = FNU(I) + .5
10      CONTINUE
        WRITE(108,840) FLONMAX,FLATMAX
C
C      TYPE OUT DATA LINE BY LINE
C
      FLAT = FLATMAX
      DO 400 I=1,NLAT
        FLCN = FLONMAX
        DO 300 J=1,NLCN
          DO 250 II=1,NGRID
            IF(FLAT.EQ.FLATC(II).AND.FLCN.EQ.FLCNC(II)) GO TO 270
250      CONTINUE
            IAP(J) = 0
            GO TO 280
270      IAP(J) = IAA(II)
280      FLCN = FLCN-DLCN
300      CONTINUE
            IF(FLAT.LE.FLATMIN)WRITE(108,855)NLON,(IAP(IP),IP=1,NLON)
& ; GO TO 400
            WRITE(108,850) NLCN,(IAP(IP),IP=1,NLCN)
            FLAT = FLAT-DLAT
400      CONTINUE
            WRITE(108,860)NLCN*4-2,NLON*4-12,FLONMIN,FLATMIN
840      FORMAT(1X,F6.2,' ',' ',F6.2/' ' +')
850      FORMAT(NI4/)
855      FORMAT(NI4)
860      FORMAT(NX,'+' /NX,F6.2,' ',' ',F6.2)
      RETURN
      END

```

WEOF LP

BIBLIOGRAPHIC DATA SHEET	1. Report No. WHOI-80-9	2.	3. Recipient's Accession No.
	4. Title and Subtitle CALCULATION OF MEAN SEASONAL SEA SURFACE ELEVATION FIELDS OVER THE NORTH ATLANTIC CONTINENTAL SHELF AND SLOPE USING HISTORICAL HYDROSTATION DATA		5. Report Date February 1980
7. Author(s) James H. Churchill		6.	
9. Performing Organization Name and Address Woods Hole Oceanographic Institution Woods Hole, MA 02543		8. Performing Organization Rept. No.	
		10. Project/Task/Work Unit No.	
		11. Contract/Grant No. DE-AC02-79-EV10005	
12. Sponsoring Organization Name and Address Department of Energy		13. Type of Report & Period Covered Technical	
		14.	
15. Supplementary Notes			
16. Abstracts <p>This report details the calculation and results of a simple coastal elevation field model which was presented in a recent paper (Csanady, 1979). Seasonal mean sea surface elevation fields were calculated over the shelf slope region off the east coast of North America north of Cape Hatteras by applying the model to a grid of averaged sigma-t's at standard depths. The sigma-t grid was calculated utilizing a data base of approximately 30,000 hydrographic stations from NODC archives. Difficulties involved with the numerical implementation of the model and economizing adaptations to the WHOI-NODC data processing program, OCCOMP, are discussed.</p>			
17. Key Words and Document Analysis. 17a. Descriptors 1. Coastal pressure field calculation 2. Model of coastal pressure field 3. East Coast continental shelf & slope pressure field calculation			
17b. Identifiers/Open-Ended Terms			
17c. COSATI Field/Group			
18. Availability Statement		19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 88
		20. Security Class (This Page) UNCLASSIFIED	22. Price

<p>Woods Hole Oceanographic Institution WHOI-80-9</p> <p>CALCULATION OF MEAN SEASONAL SEA SURFACE ELEVATION FIELDS OVER THE NORTH ATLANTIC CONTINENTAL SHELF AND SLOPE USING HISTORICAL HYDRO-STATION DATA by James H. Church111. 88 pages. February 1980. Prepared for the Department of Energy under Grant DE-AC02-79-EV10005.</p> <p>This report details the calculation and results of a simple coastal elevation field model which was presented in a recent paper (Csanady, 1979). Seasonal mean sea surface elevation fields were calculated over the shelf slope region off the east coast of North America north of Cape Hatteras by applying the model to a grid of averaged sigma-t's at standard depths. The sigma-t grid was calculated utilizing a data base of approximately 30,000 hydrographic stations from NODC archives. Difficulties involved with the numerical implementation of the model and economizing adaptations to the WHOI-NODC data processing program, OCCOMP, are discussed.</p> <p>1. Coastal pressure field calculation</p> <p>2. Model of coastal pressure field</p> <p>3. East Coast continental shelf & slope pressure field calculation</p> <p>I. Church111, James H. II. DE-AC02-79-EV10005</p> <p>This card is UNCLASSIFIED</p>	<p>Woods Hole Oceanographic Institution WHOI-80-9</p> <p>CALCULATION OF MEAN SEASONAL SEA SURFACE ELEVATION FIELDS OVER THE NORTH ATLANTIC CONTINENTAL SHELF AND SLOPE USING HISTORICAL HYDRO-STATION DATA by James H. Church111. 88 pages. February 1980. Prepared for the Department of Energy under Grant DE-AC02-79-EV10005.</p> <p>This report details the calculation and results of a simple coastal elevation field model which was presented in a recent paper (Csanady, 1979). Seasonal mean sea surface elevation fields were calculated over the shelf slope region off the east coast of North America north of Cape Hatteras by applying the model to a grid of averaged sigma-t's at standard depths. The sigma-t grid was calculated utilizing a data base of approximately 30,000 hydrographic stations from NODC archives. Difficulties involved with the numerical implementation of the model and economizing adaptations to the WHOI-NODC data processing program, OCCOMP, are discussed.</p> <p>1. Coastal pressure field calculation</p> <p>2. Model of coastal pressure field</p> <p>3. East Coast continental shelf & slope pressure field calculation</p> <p>I. Church111, James H. II. DE-AC02-79-EV10005</p> <p>This card is UNCLASSIFIED</p>	<p>Woods Hole Oceanographic Institution WHOI-80-9</p> <p>CALCULATION OF MEAN SEASONAL SEA SURFACE ELEVATION FIELDS OVER THE NORTH ATLANTIC CONTINENTAL SHELF AND SLOPE USING HISTORICAL HYDRO-STATION DATA by James H. Church111. 88 pages. February 1980. Prepared for the Department of Energy under Grant DE-AC02-79-EV10005.</p> <p>This report details the calculation and results of a simple coastal elevation field model which was presented in a recent paper (Csanady, 1979). Seasonal mean sea surface elevation fields were calculated over the shelf slope region off the east coast of North America north of Cape Hatteras by applying the model to a grid of averaged sigma-t's at standard depths. The sigma-t grid was calculated utilizing a data base of approximately 30,000 hydrographic stations from NODC archives. Difficulties involved with the numerical implementation of the model and economizing adaptations to the WHOI-NODC data processing program, OCCOMP, are discussed.</p> <p>1. Coastal pressure field calculation</p> <p>2. Model of coastal pressure field</p> <p>3. East Coast continental shelf & slope pressure field calculation</p> <p>I. Church111, James H. II. DE-AC02-79-EV10005</p> <p>This card is UNCLASSIFIED</p>
---	---	---