The University of Wisconsin
Computer Sciences Department
1210 West Dayton Street
Madison, Wisconsin  5370(

# A NUMERICAL APPROACH TO WIND-DRIVEN OCEAN CIRCULATION

by
Donald Greenspan

Appendix:  FORTRAN Program for Wind-Driven Circulation

by
Alvin B. Schubert

Technical Report #149

March, 1972

# ABSTRACT

A numerical method is developed for a widely studied, wind-driven ocean circulation model. Examples of flow patterns of the northern Pacific, which include large non-linear effects, are given.

# A NUMERICAL APPROACH TO WIND-DRIVEN OCEAN CIRCULATIOΓ

by
Donald Greenspan

## 1. INTRODUCTION

The study of wind-driven ocean circulation has been, and continues to be, of interest to mathematicians, fluid dynamicists, meteorologists, and geophysicists (see, e.g., references [1] - [7], [9] - [13], [15], [16], and the additional references contained therein). The purely mathematical approach (see, e.g., [4], [5], [12], [13]) to related dynamical problems has been based primarily on linearization and the application of singular perturbation techniques. Though this approach has yielded some general qualitative consistency with actual circulation patterns, it has not yielded results of acceptable quantitative accuracy ([4], [5]). For nonlinear models, the application of numerical methods using explicit step-ahead difference techniques has yielded a variety of interesting fluid phenomena (see, e.g., [2], [3], [7], [16]). However, for such techniques, no attempt is usually made to verify whether or not a stable time step $\Delta t$ is sufficiently small to insure a physically reliable degree of convergence. In addition, the excessive costs of such methods usually places them beyond the means of most researchers.

By limiting attention to steady, nonlinear, two-dimensional problems, we will develop in this paper a fast, economical numerical method for the study of wind-driven ocean circulation. The physical model is one of the most intensively studied of the last twenty years [4]. The method to be developed was applied, in an earlier form, to cavity flow problems [10] with arbitrary Reynold's numbers [8]. Throughout, it must be kept clearly in mind that, though the method will work for all choices of equation parameters, a physical steady state solution may not exist for all such parameter choices ([1], [2], [9]). Thus, physical insight is essential in the application of the method and in the interpretation of the results.

## 2. A NORTH PACIFIC MODEL

For clarity, let us direct attention to a prototype model of the wind-driven circulation of the Pacific Ocean between 15° and 55° north latitude, which is formulated in simplified coordinates as follows [4]. Let the basin be bounded by the isosceles trapezoid OABC shown in Figure 2.1, where OA is a segment of the line $y=x$, AB is a segment of the line $y=\pi$, and BC is a segment of the line $y=4\pi-x$. The differential equation to be satisfied by stream function $\psi$ in the interior $R$ of the basin is

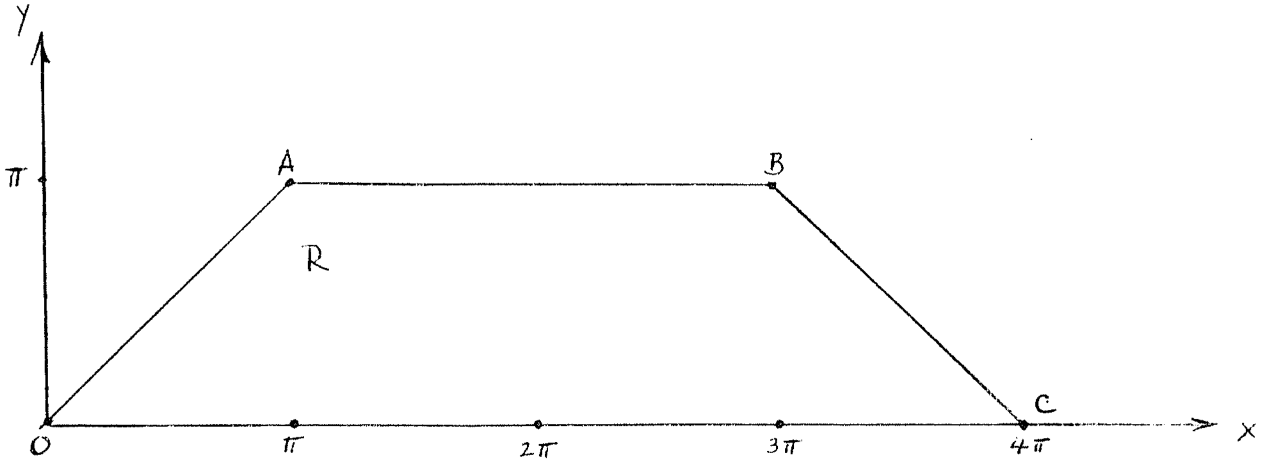


Figure 2.1

(2.1) $\varepsilon\ \Delta\Delta\psi+\alpha(\psi_y\Delta\psi_x-\psi_x\Delta\psi_y) - \psi_x = \sin y$ ,

where $\varepsilon$ and $\alpha$ are small, positive parameters. Equation (2.1) incorporates frictional resistance, fluid acceleration relative to the earth, the change with latitude of Coriolis acceleration, and an idealized wind stress distribution in which easterly winds predominate over the lower half of the basin while westerly winds predominate over the upper half. The boundary conditions to be satisfied are

(2.2) $\psi = 0$ , on OABC

(2.3) $\psi_n = 0$ , on OA and on BC

(2.4) $\psi_{yy} = 0$ , on AB and on OC.

## 3.1 The Numerical Method

It will be convenient, for the numerical method to be developed, to reformulate the problem of Section 2 as follows. First, introduce vorticity function $\omega$ by

$$(3.1)\qquad \Delta\psi = -\omega ,$$

so that (2.1) implies

$$(3.2)\qquad \Delta\omega + \frac{\alpha}{\varepsilon}\left[\psi_y\omega_x - \psi_x\omega_y\right] = -\frac{1}{\varepsilon}\left[\psi_x + \sin y\right] .$$

System (3.1)-(3.2) is, therefore, equivalent to (2.1). Note, immediately, that (3.1) is linear in $\psi$ , while (3.2) is linear in $\omega$ . Second, assume that (3.1) is valid, in the limit, as one approaches OABC from the interior R , and note that $\psi_t = 0$ , where $\psi_t$ is the derivative in the tangent direction on OA and BC. Thus, boundary conditions (2.2)-(2.4) can be reformulated as

$$(3.3)\qquad \psi = 0 \quad , \text{ on OABC}$$

$$(3.4)\qquad \psi_x = \psi_y = 0 \quad , \text{ on OA and on BC}$$

$$(3.5)\qquad \omega = 0 \quad , \text{ on AB and on OC.}$$

We will then consider the problem as defined by (3.1)-(3.5) and the numerical method is described as follows.

For a fixed positive integer $n$ , set $h = \frac{\pi}{n}$ . Starting at (0,0) with grid size $h$ , construct the set of interior grid points $R_h$ in R , and the set of boundary grid points $S_h$ in OA, AB, BC, and OC. Let $\overline{S}_h$ be that subset of $S_h$ which is not in AB or OC. Then for given tolerances $\varepsilon_1$ and $\varepsilon_2$ , let us proceed to construct on $R_h$ a sequence of discrete stream functions

(3.6) $\psi^{(0)}, \psi^{(1)}, \psi^{(2)}, \psi^{(3)}, \ldots,$

and on $R_h + \overline{S}_h$ a sequence of discrete vorticity functions

(3.7) $\omega^{(0)}, \omega^{(1)}, \omega^{(2)} \omega^{(3)}, \ldots,$

such that, for some positive integer $K$, both the following are valid informly:

(3.8) $|\psi^{(K)} - \psi^{(K+1)}| < \varepsilon_1$ , on $R_h$

(3.9) $|\omega^{(K)} - \omega^{(K+1)}| < \varepsilon_2$ , on $R_h + \overline{S}_h$ .

Initially, estimate $\psi^{(0)}$ on $R_h$ and $\omega^{(0)}$ on $R_h + \overline{S}_h$ . This may be done by using constant input values or by using previous numerical results (bootstrapping). To produce the kth iterate $\psi^{(k)}$ of (3.6), for $k=1,2,\ldots,$ write down at each point $((m+1)h, mh), m=1,2,\ldots,n-1$, the equation

(3.10) $9\psi((m+1)h, mh) = \psi((m+2)h, (m-1)h)$ ;

at each point $(4\pi - (m+1)h, mh), m=1,2,\ldots,n-1$, the equation

(3.11) $9\psi(4\pi - (m+1)h, mh) = \psi(4\pi - (m+2)h, (m-1)h)$ ,

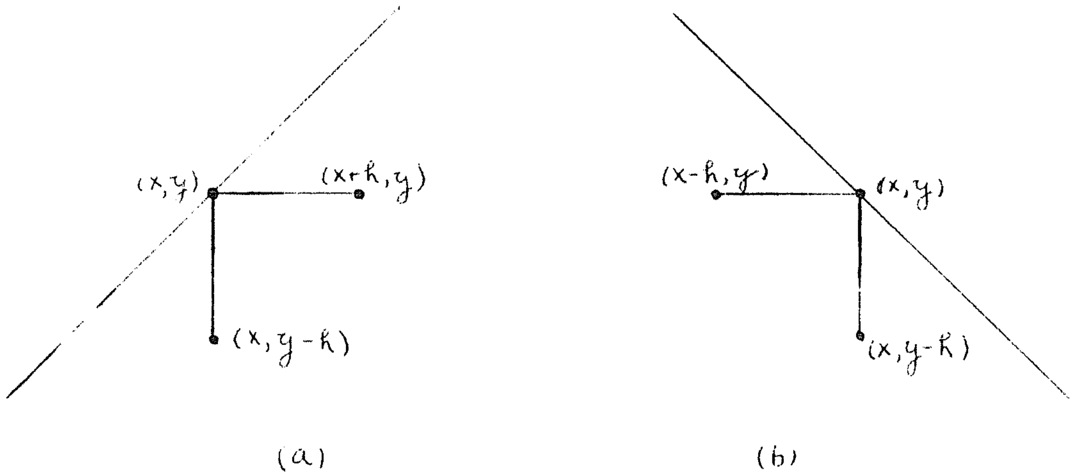
and at the remaining points of $R_h$ the discrete analogue

(3.12) $-4\psi(x,y) + \psi(x+h,y) + \psi(x,y+h) + \psi(x-h,y) + \psi(x,y-h) = -h^2\omega^{(k-1)}(x,y)$

of (3.1). Difference equations (3.10) and (3.11) insure good approximations for (3.4) near OA and BC. Solve the resulting linear algebraic system by successive overrelaxation with overrelaxation factor $r_\psi$ and denote the solution by $\overline{\psi}^{(k)}$ . Then, on $R_h$ , $\psi^{(k)}$ is defined by the smoothing formula

$$(3.13)\quad \psi^{(k)}=\rho\psi^{(k-1)}+(1-\rho)\overline{\psi}^{(k)} \ , \ 0 \le \rho \le 1 \ .$$

To produce the $k^{th}$ iterate $\omega^{(k)}$ of sequence (3.7) for $k=1,2,\ldots,$ proceed as follows. Let $(x,y)$ be a point of $\overline{S}_h$ which is in OA, as shown in Figure 2.2a. Then at each such point write down the approximation



(a)

(b)

Figure 2.2

$$(3.14)\quad \bar{\omega}^{(k)}(x,y)=\frac{-2}{h^2}\,\psi^{(k)}(x+h,y)-\frac{2}{h^2}\psi^{(k)}(x,y-h)\ .$$

Approximation (3.14) can be derived readily by inserting Taylor series for $\psi(x+h,y)$ and $\psi(x,y-h)$ into

$$\psi_{xx}(x,y)+\psi_{yy}(x,y)=\alpha_0\psi(x,y)+\alpha_1\psi_x(x,y)+\alpha_2\psi_y(x,y)$$
$$+\alpha_3\psi(x+h,y)+\alpha_4\ \psi(x,y-h)\ ,$$

by setting corresponding coefficients equal, and by utilizing (3.1), (3.3) and (3.4). Next, let $(x,y)$ be a point of $\bar{S}_h$ which is in BC, as shown in Figure 2.2b. Then, at each such point write down the approximation

$$(3.15)\quad \bar{\omega}^{(k)}(x,y)=-\frac{2}{h^2}\psi^{(k)}(x-h,y)-\frac{2}{h^2}\psi^{(k)}(x,y-h)\ .$$

Smooth $\omega$ on the boundary by

$$(3.16)\quad \omega^{(k)}(x,y)=\mu_1\omega^{(k-1)}(x,y)+(1-\mu_1)\bar{\omega}^{(k)}(x,y),0\leq\mu_1\leq 1\ .$$

Next, at each point $(x,y)$ in $R_h$ , write down a difference analogue of (3.2) as follows. First, set

$$A=\frac{\psi(x,y+h)+\psi(x,y-h)}{2h}\ ,\quad B=\frac{\psi(x+h,y)-\psi(x-h,y)}{2h}\ ,$$

and define M and N as follows:

$$M = \frac{\omega(x+h,y)-\omega(x,y)}{h} , \quad \text{if} \quad A \geq 0$$

$$M = \frac{\omega(x,y)-\omega(x-h,y)}{h} , \quad \text{if} \quad A < 0$$

$$N = \frac{\omega(x,y)-\omega(x,y-h)}{h} , \quad \text{if } B \geq 0$$

$$N = \frac{\omega(x,y+h)-\omega(x,y)}{h} , \quad \text{if } B < 0 .$$

Then, at $(x,y)$ , the difference analogue of (3.2) is

$$(3.17) \quad -4\omega(x,y)+\omega(x+h,y)+\omega(x,y+h)+\omega(x-h,y)+\omega(x,y-h)$$

$$+ \frac{\alpha h^2}{\varepsilon}\left[AM - BN\right] = -\frac{h^2}{\varepsilon}\left[B + \sin y\right] .$$

One now solves the linear algebraic system defined by (3.17 with boundary values defined by (3.16) by successive over-relaxation with $r_\omega$ and denotes the solution by $\overline{\omega}^{(k)}$ . Then, on $R_h$ , $\omega^{(k)}$ is defined by the smoothing formula

$$\omega^{(k)}=\mu_2\omega^{(k-1)}+(1-\mu_2)\overline{\omega}^{(k)}, \; 0\leq\mu_2\leq 1 .$$

The iteration is continued until (3.8) and (3.9) are valid. After checking that $\psi^{(K+1)}$ and $\omega^{(K+1)}$ are actually solutions of the difference equations, these are taken to be the approximations of $\psi(x,y)$ and $\omega(x,y)$ , respectively.

## 4. Examples

From the variety of examples run on the UNIVAC 1108 at the University of Wisconsin, let us show how to generate the solution for a typical problem of physical interest [5] that in which $\varepsilon = 0.005$, $\alpha = 0.15$ . Throughout, let $n = 10$.

Since no choices of $\rho$ , $\mu_1$ , $\mu_2$ , $\varepsilon_1$ , and $\varepsilon_2$ ever yielded convergence for initial values $\psi^{(0)}=\omega^{(0)}=0$ , a bootstrap procedure was utilized. Beginning with $\psi^{(0)}=\omega^{(0)}=0$ , the problem was solved numerically for $\varepsilon=0.1$ , $\alpha=0$ . Then, using each new result as $\psi^{(0)}$ and $\omega^{(0)}$ for the next case, the problem was solved in succession for $\varepsilon=.05$ , .03, .01, .005, .001, each with $\alpha=0$ . Except for $\varepsilon=.03$ , which varied little from $\varepsilon=.05$ , these results are shown in Figures 4.1-4.5. Figures 4.6-4.7 show the results of bootstrapping from $\varepsilon=0.05$, $\alpha=0$ to $\alpha=0.2$, 0.1 , while Figures 4.8-4.9 show the results of bootstrapping from $\varepsilon=.005$, $\alpha=0$ to $\alpha=0.04$, 0.15 . The running times never exceeded five minutes for any individual case and the other parameter choices are shown in the TABLE.

TABLE

| $\varepsilon$ | $\alpha$ | $\rho$ | $\mu_1$ | $\mu_2$ | $\varepsilon_1$ | $\varepsilon_2$ | $r_\psi$ | $r_\omega$ |
|---|---|---|---|---|---|---|---|---|
| 0.1 | 0 | 0.05 | 0.1 | 0.9 | $10^{-3}$ | $10^{-2}$ | 1.7 | 1.6 |
| 0.05 | 0 | 0.05 | 0.05 | 0.95 | $10^{-2}$ | $10^{-1}$ | 1.7 | 1.6 |
| 0.03 | 0 | 0.02 | 0.02 | 0.98 | 0.03 | 0.3 | 1.7 | 1.6 |
| 0.01 | 0 | 0.005 | 0.005 | 0.995 | 0.045 | 0.45 | 1.7 | 1.6 |
| 0.005 | 0 | 0.0025 | 0.0025 | 0.9975 | 0.085 | 0.55 | 1.7 | 1.6 |
| 0.001 | 0 | 0.0005 | 0.0005 | 0.9995 | 0.1 | 0.7 | 1.7 | 1.6 |
| 0.05 | 0.2 | 0.05 | 0.05 | 0.95 | 0.005 | 0.01 | 1.7 | 1.2 |
| 0.05 | 1.0 | 0.05 | 0.05 | 0.95 | 0.005 | 0.01 | 1.7 | 1.2 |
| 0.005 | .04 | 0.0025 | 0.0025 | 0.9975 | 0.05 | 0.3 | 1.7 | 1.0 |
| 0.005 | .15 | 0.0025 | 0.0025 | 0.9975 | 0.05 | 0.3 | 1.7 | 1.0 |

## 5. Remarks

Beyond the general flow patterns exhibited in Figures 4.1-4.9, it is worth noting that one can detect, from these, certain effects of varying $\varepsilon$ and $\alpha$. Thus, Figures 4.1-4.5 indicate that if one neglects fluid acceleration relative to the earth ($\alpha=0$), then a decrease in frictional resistance ($\varepsilon \to 0$) results in a tendency to develop a secondary vortex in the lower-left hand section of the basin. However, inclusion of a moderate amount of fluid acceleration tends to negate this effect (see Figures 4.4, 4.8).

Numerically, the results described in Section 4 should be considered more qualitative than quantitative because of the relatively large grid size and convergence tolerances. However, the method can be applied with smaller grid sizes and tolerances at a moderately increased cost in computer time. Exploratory examples with $n = 20$ and parameter choices like those in the TABLE indicate that each case requires 5-20 minutes of computing time.

Finally, so that any researcher can reproduce our results the computer program used is made readily available in [14].

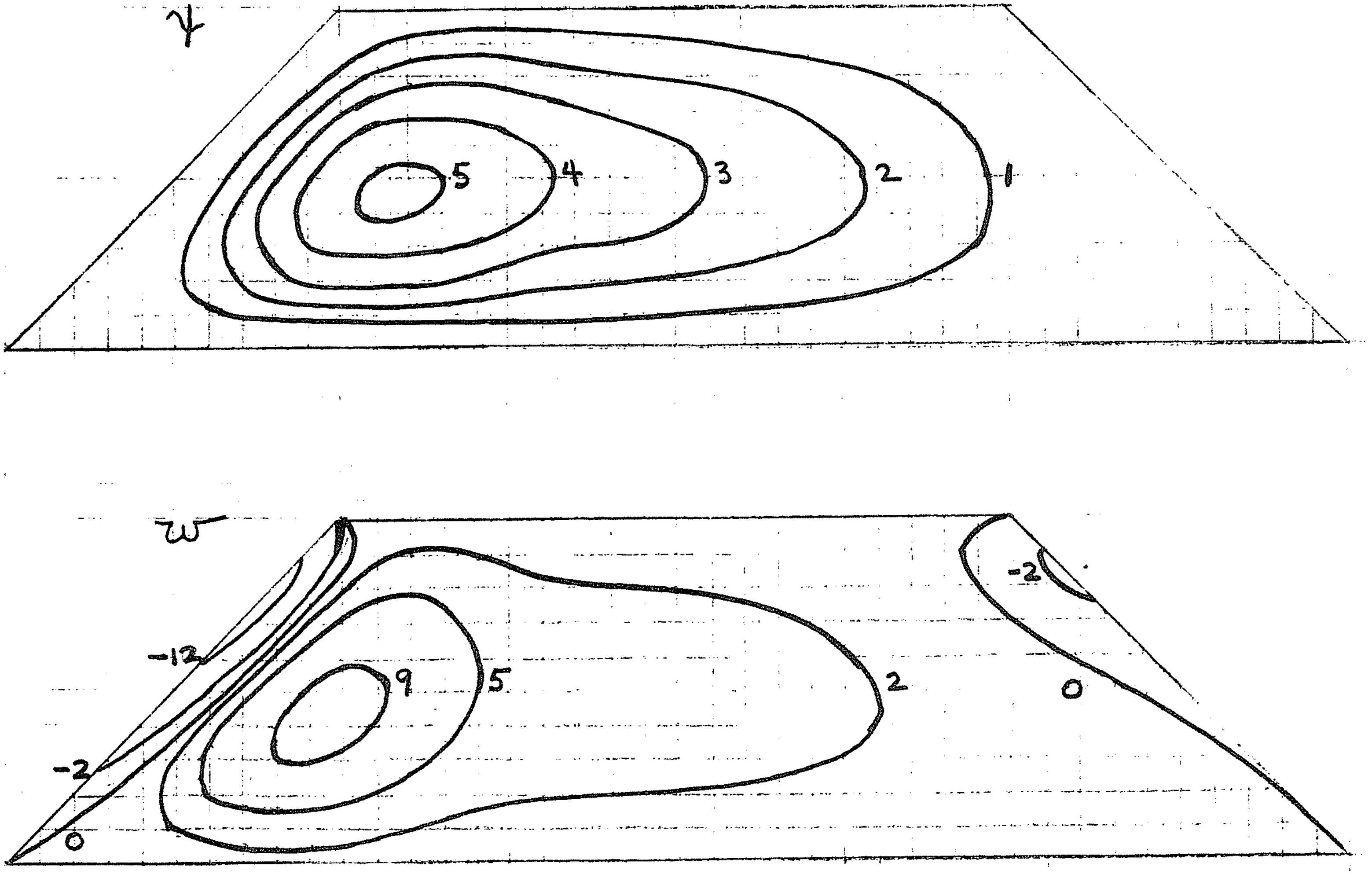


FIGURE 4.1: $\varepsilon = 0.1, \alpha = 0$

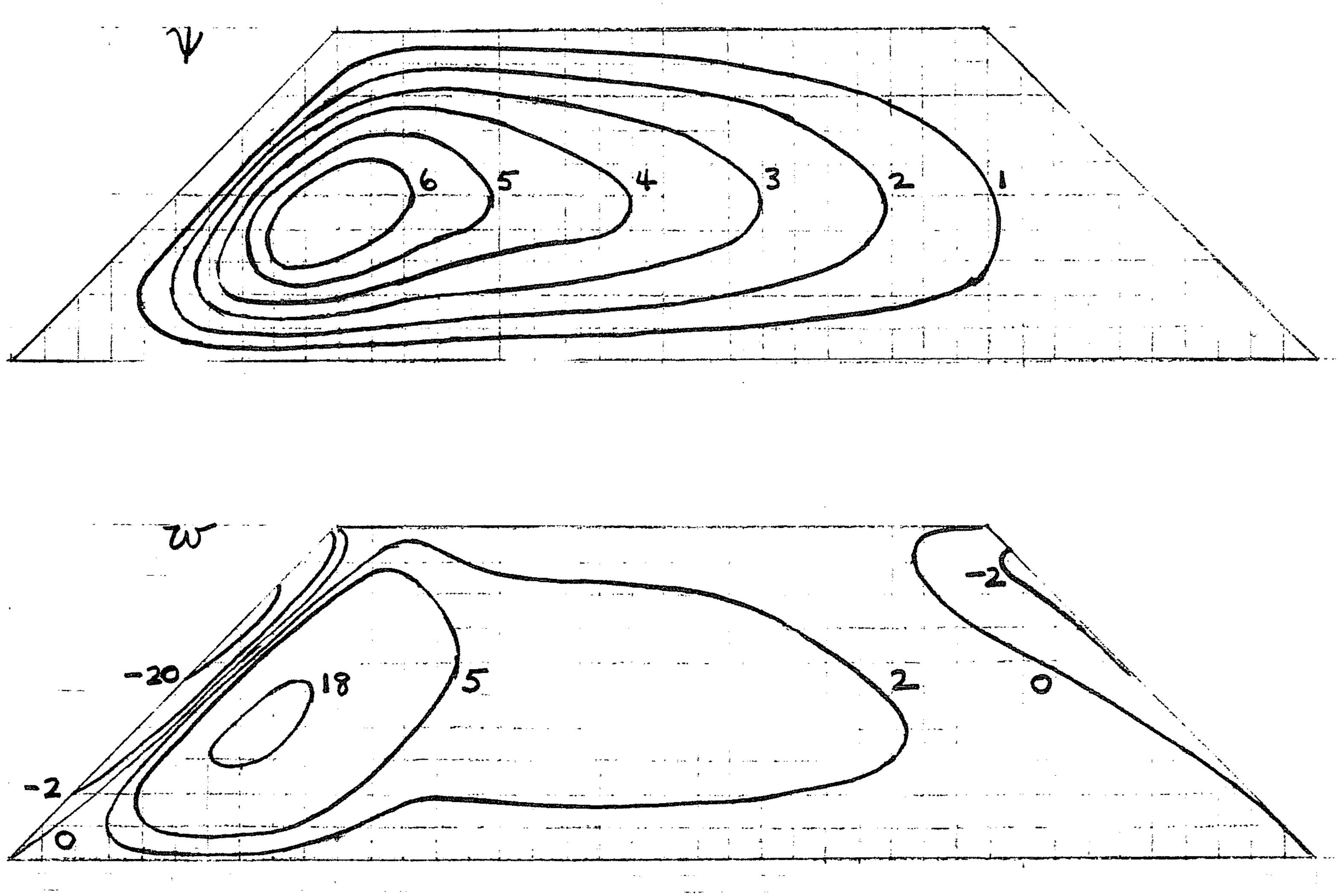


FIGURE 4.2 : ε = 0.05 , α = 0

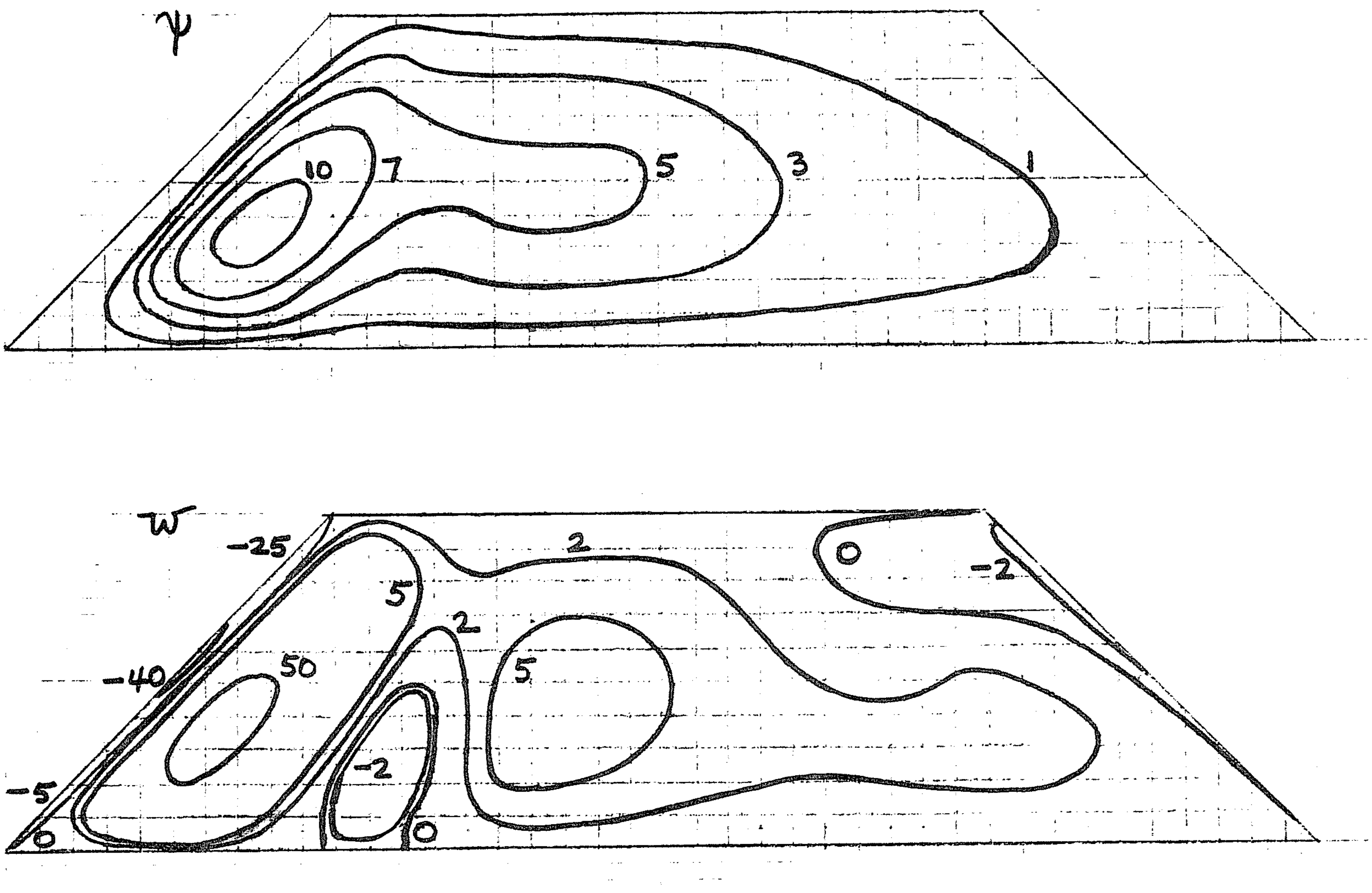


FIGURE 4.3: $\varepsilon = 0.01$, $\alpha = 0$

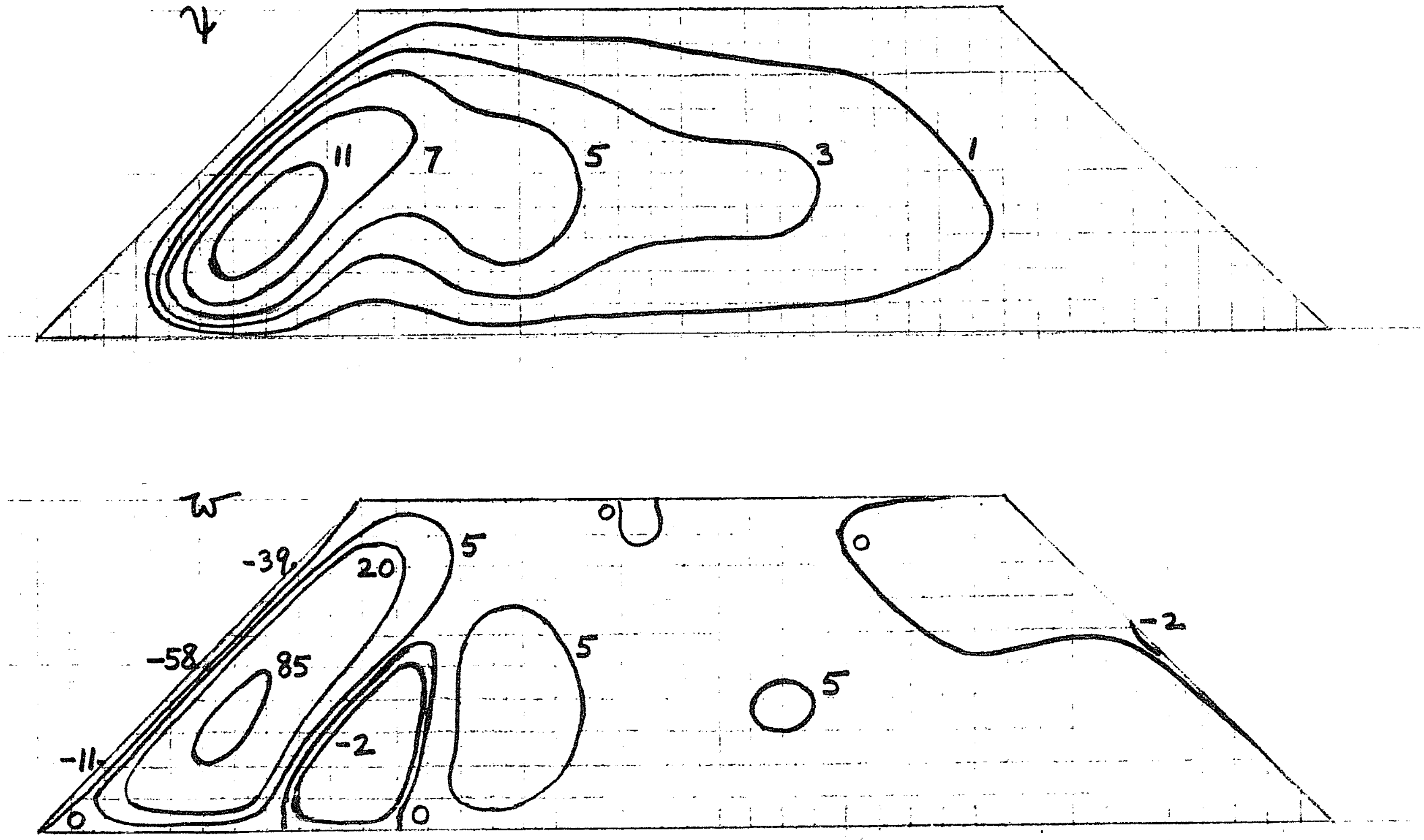


FIGURE 4.4: $\varepsilon=0.005$, $\alpha=0$

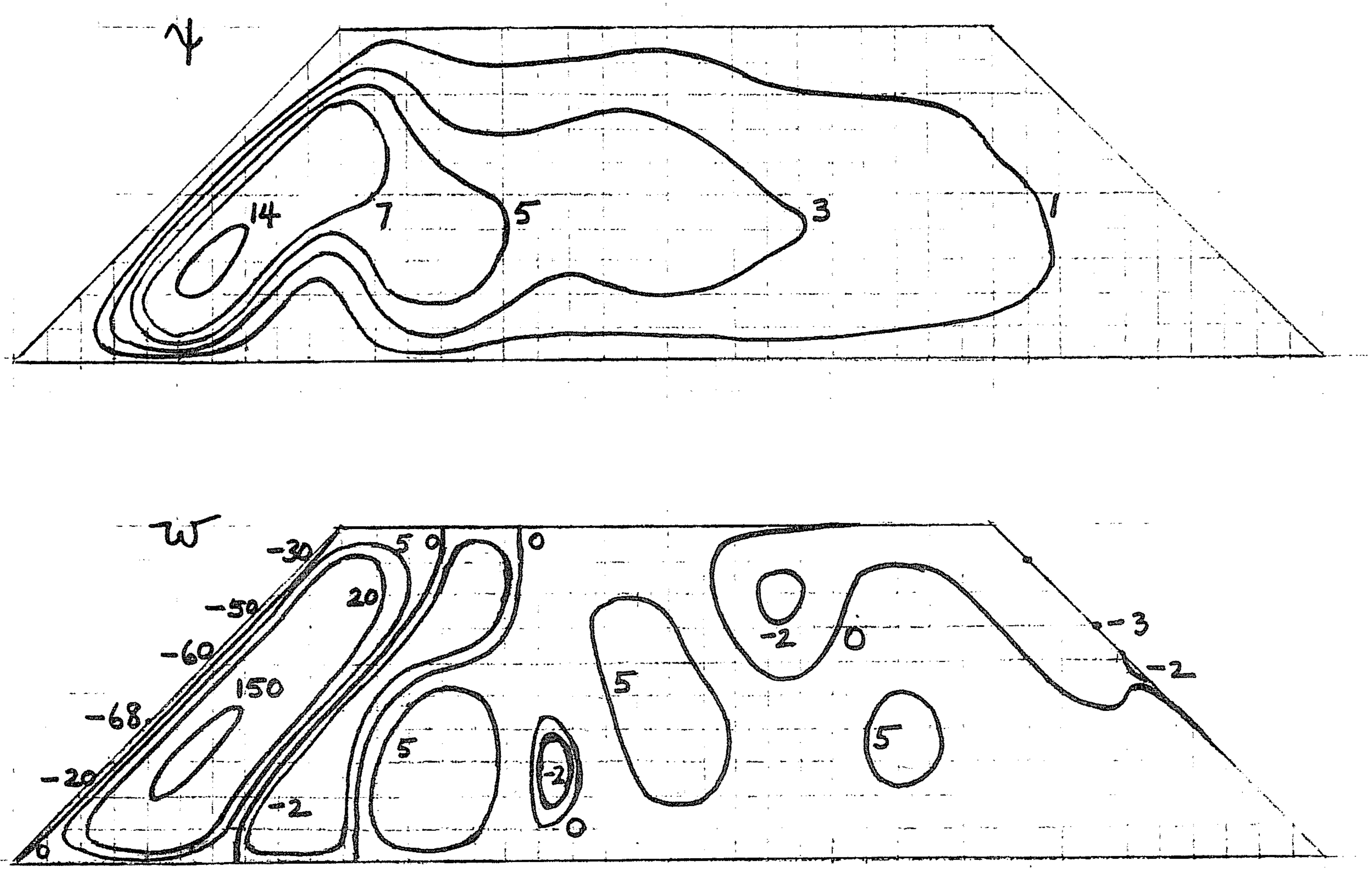


FIGURE 4.5 : $\varepsilon = 0.001$, $\alpha = 0$

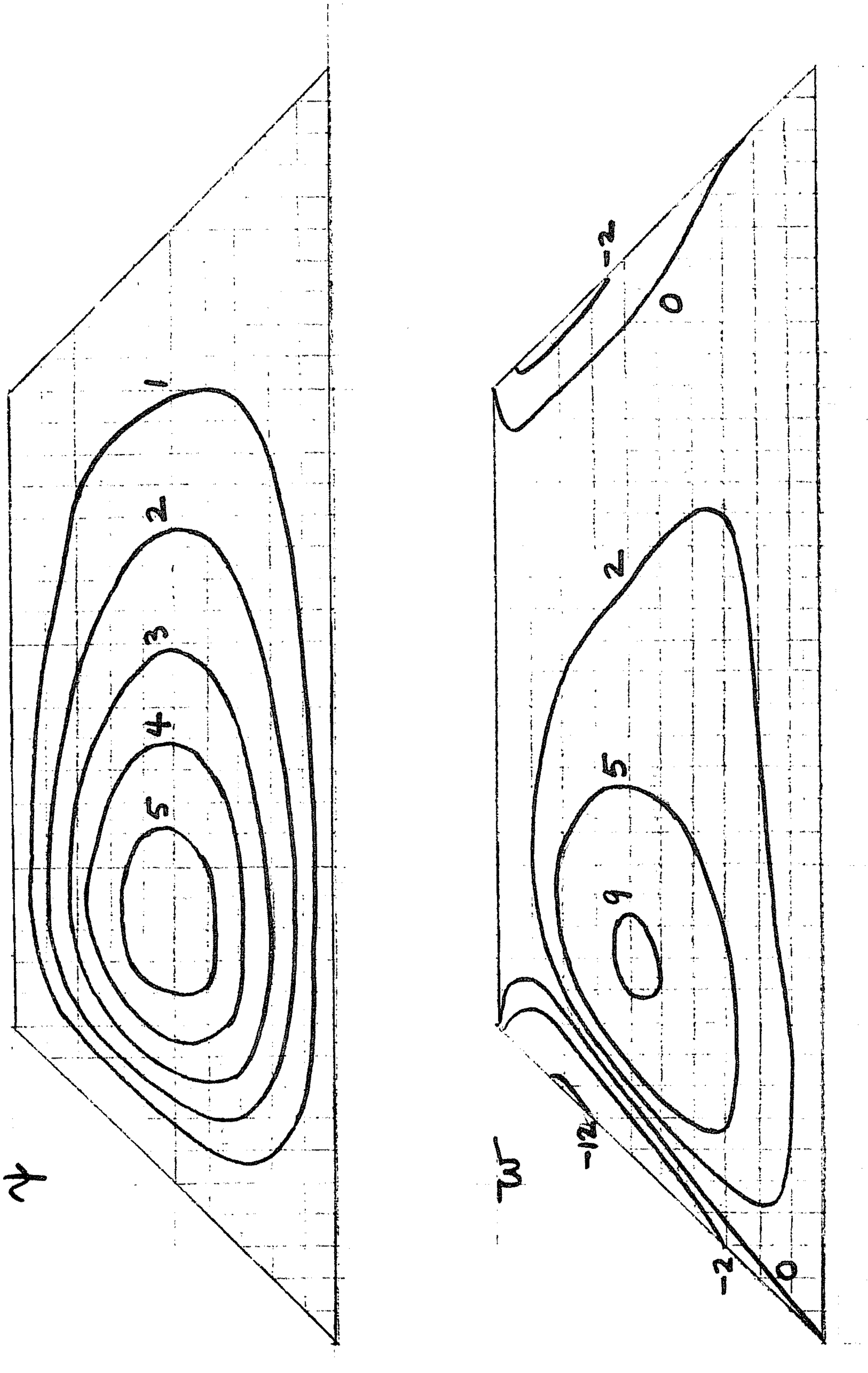


FIGURE 4.6: ε=0.05, α=0.2

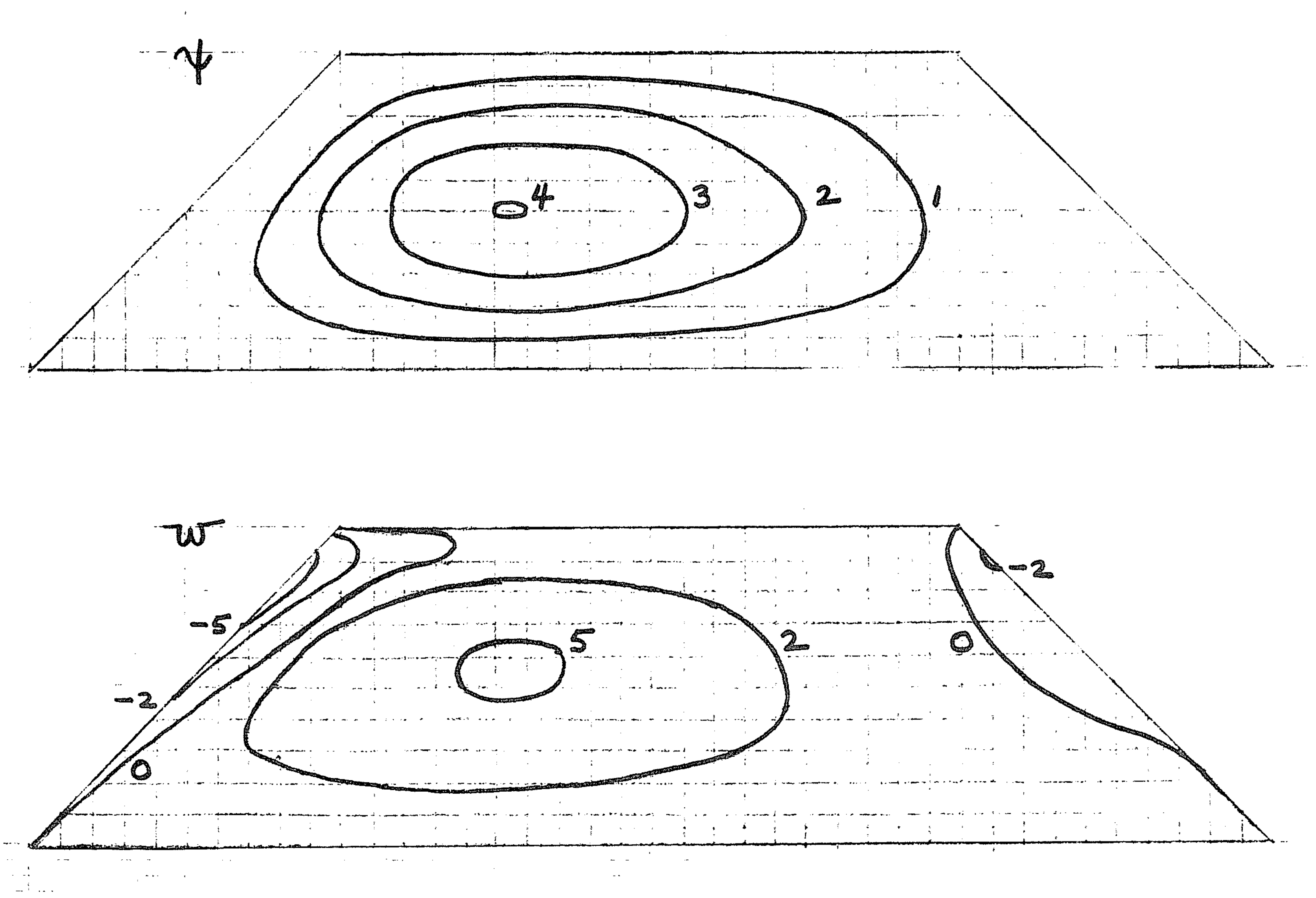


FIGURE 4.7: ε = 0.05, α = 1.0

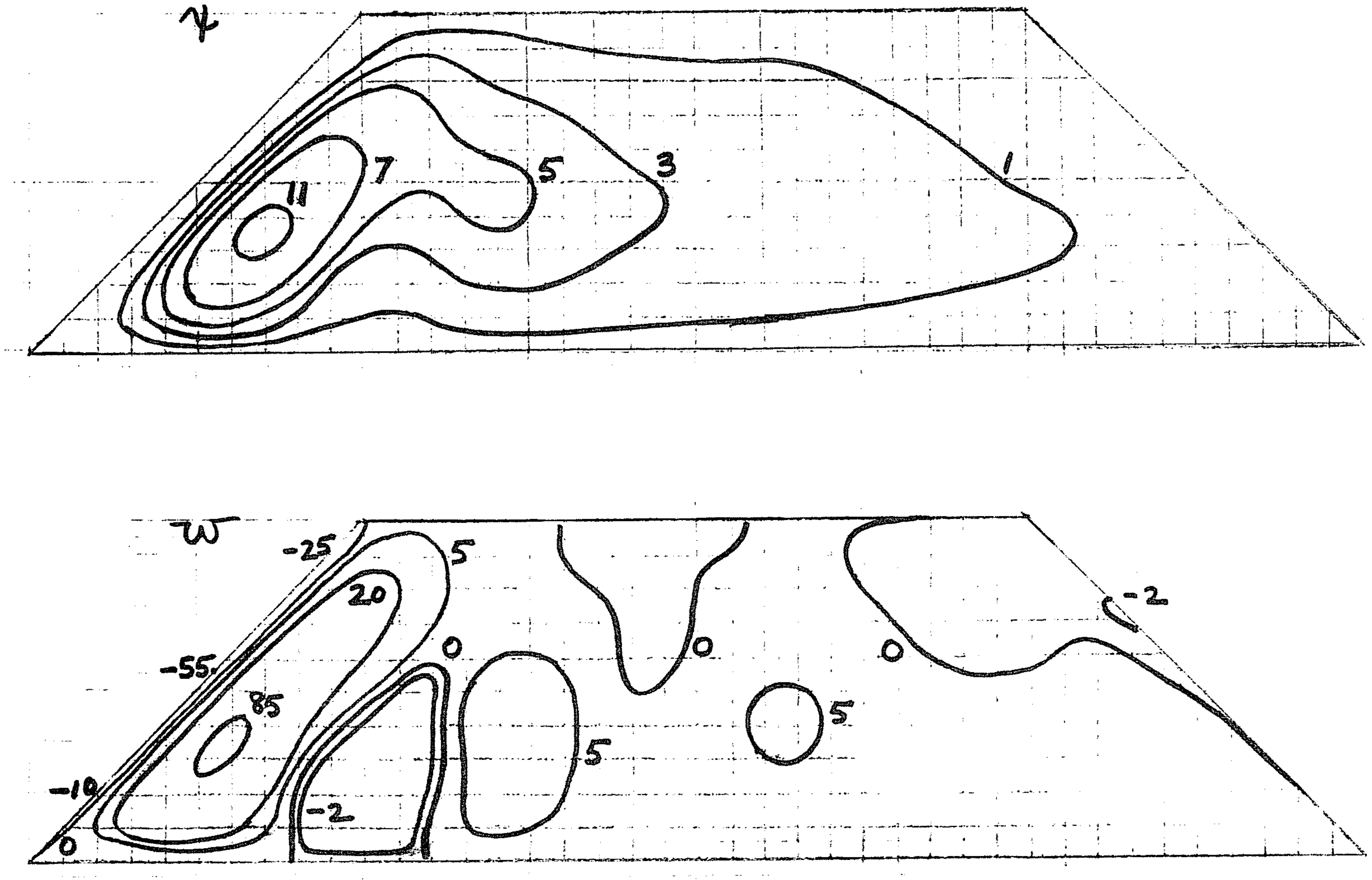


FIGURE 4.8: $\varepsilon = 0.005$, $\alpha = 0.04$

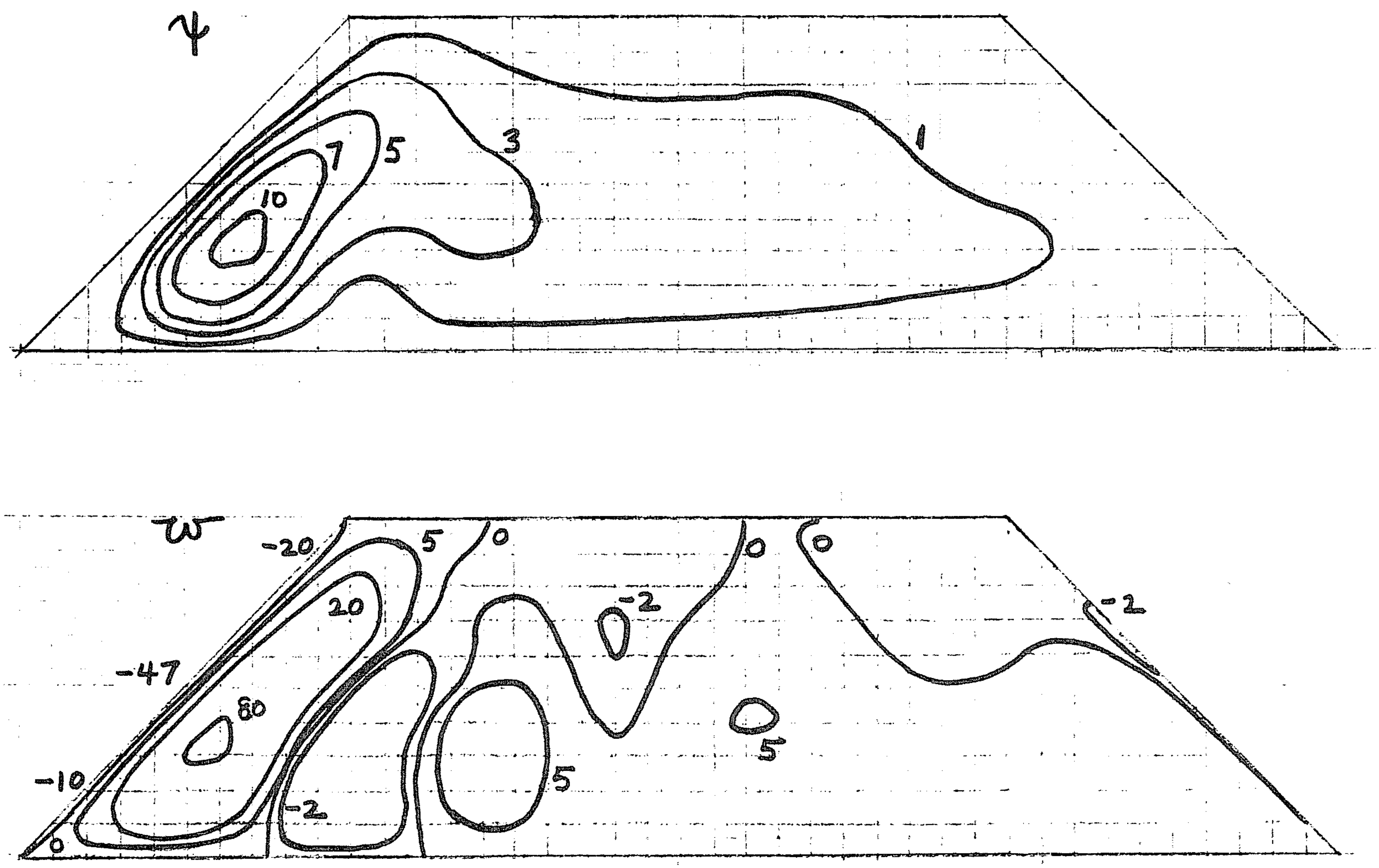


FIGURE 4.9: ε = 0.005, α = 0.15

# REFERENCES

1. R. Beardsley, "A laboratory model of a wind-driven ocean circulation", J. Fluid Mech., 38, 1969, p. 255.

2. K. Bryan, "A numerical investigation of a non-linear model of a wind-driven ocean", J. Atmos. Sci., 20, 1963, p. 594.

3. ____________, "A numerical method for the study of the circulation of the world ocean", J. Comp. Phys., 4, 1969, p. 347.

4. G. F. Carrier, "Singular perturbation theory and geophysics", in Studies in Applied Mathematics, edited by A. H. Taub, Prentice-Hall, Englewood Cliffs, N. J., 1971, p. 1.

5. G. F. Carrier and A. R. Robinson, "On the theory of a wind-driven ocean circulation", J. Fluid Mech., 12, 1962, p. 49.

6. D. Cox, "A mathematical model of the Indian Ocean", Deep-Sea Res., 17, 1970, p. 47.

7. A. Gill and K. Bryan, "Effects of geometry on the circulation of a three-dimensional southern-hemisphere ocean model", Deep-Sea Res., 18, 1971, p. 685.

8. D. Greenspan, "Computer power and its impact on applied mathematics", in Studies in Applied Mathematics, edited by A. H. Taub, Prentice-Hall, Englewood Cliffs, N. J., 1971, p. 65.

9. H. P. Greenspan, The Theory of Rotating Fluids, Cambridge Univ. Press, London, 1968.

10. M. Kawaguti, "Numerical solution of the Navier-Stokes equations for the flow in a two-dimensional cavity", J. Phys. Soc. Japan, 16, 1961, p. 2307.

11. G. W. Morgan, "On the wind-driven ocean circulation", Tellus, 8, 1956, p. 301.

12. W. H. Munk, "On the wind-driven ocean circulation", J. Meteorology, 7, 1950, p. 79.

13. W. H. Munk and G. F. Carrier, "The wind-driven circulation in ocean basins of various shapes", Tellus, 2, 1950, p. 158.

14. A. B. Schubert, "FORTRAN program for wind-driven circulation", TR. #149, Dept. Comp. Sci., Univ. of Wis., Madison, 1972.

15. H. U. Sverdrup, "Wind-driven currents in a baroclinic ocean; with application to the equatorial currents of the eastern Pacific", Proc. Nat. Acad. Sci., 33, 1947, p. 318.

16. G. Veronis, "Wind-driven ocean circulation-Part 2. Numerical solutions of the nonlinear problem", Deep-Sea Res., 13, 1966, p. 31.

# APPENDIX

```
@RUN SCHUBERT,ETC.

       PRELIMINARY TAPE AND DRUM FILE ASSIGNMENTS AND COPYING TO BE MADE
       IF THERE IS TAPE INPUT OR IF THERE IS TO BE TAPE OUTPUT.

@ASG,TH  10.,T,$0158
@ASG,T   11.,F2/11/TRK/20
@ASG,T   12.,F2/11/TRK/20
@REWIND  10.
@COPY,GC 10.,11.
@REWIND  10.


@FOR,ISZ  ,NORPAC

C      PARAMETER VALUES TO BE SET FOR EACH RUN

C          N = PI/GRID SIZE
C      MAXIT = MAXIMUM NUMBER OF SOR ITERATIONS TO BE ALLOWED IN
C              EACH OUTER ITERATION.
C      IDREC = NUMBER OF TAPE RECORD CONTAINING INITIAL VALUES FOR
C              OUTER ITERATION FOR COMPUTING STREAM (PSI) AND VORTICITY
C              (OMEGA) FUNCTIONS. (IDREC = 0 IMPLIES NO TAPE INPUT.)
C       NREC = TOTAL NUMBER OF RECORDS ON INPUT TAPE FILE.

       PARAMETER N=10,NP1=N+1,N4=4*N,N4P1=N4+1,N3P1=3*N+1
       PARAMETER MAXIT=200,IDREC=5,NREC=12

       REAL MU(2),MU1(2)
       DIMENSION PSI(N4P1,NP1,5),OMEGA(N4P1,NP1,5),C(N4P1,NP1,5),
      * D(N4P1,NP1),SINY(NP1),HSQOM(N4P1,NP1)
       DATA PI/3.14159265/

       IEND=0
       ITREC=0
       ISOL=0
       ALFCON=-1.
       EPSCON=-1.

C      INPUT INITIAL VALUES FROM TAPE IF REQUIRED.

       IF(IDREC.EQ.0) GO TO 3

C      ASSUME DATA HAS BEEN COPIED FROM TAPE TO DRUM FILE 11, WHICH,
C      ALONG WITH DRUM FILE 12, HAS BEEN ASSIGNED FOR THIS RUN.
C      AT THE END OF THIS RUN, DRUM FILE 12 WILL CONTAIN THE CONTENTS
C      OF THE INPUT TAPE FILE PLUS POSSIBLE ADDITIONAL RECORDS COMPOSED
C      OF SOLUTIONS OBTAINED FROM DATA CASES EXECUTED THIS RUN.  IF AND
C      ONLY IF THE RUN TERMINATES NORMALLY, THE CONTENTS OF DRUM FILE 12
C      MAY, IF DESIRED, BE COPIED OUT TO TAPE TO BE USED AS INPUT
C      FOR LATER RUNS.

       DO 1 K=1,IDREC
       ITREC=K
      READ(11,END=2) ((PSI(I,J,5),I=1,N4P1),J=1,NP1),((OMEGA(I,J,5),I=1,
     *  N4P1),J=1,NP1)
    1 WRITE(12) ((PSI(I,J,5),I=1,N4P1),J=1,NP1),((OMEGA(I,J,5),I=1,N4P1)
     * ,J=1,NP1)
      WRITE(6,85)
      CALL PRINT(PSI(1,1,5))
      CALL PRINT(OMEGA(1,1,5))
      IF(IDREC.EQ.NREC) GO TO 3
      I1=IDREC+1
      DO 301 K=I1,NREC
      ITREC=K
      READ(11,END=2) ((PSI(I,J,3),I=1,N4P1),J=1,NP1),((OMEGA(I,J,3),I=1,
     * N4P1),J=1,NP1)
  301 WRITE(12) ((PSI(I,J,3),I=1,N4P1),J=1,NP1),((OMEGA(I,J,3),I=1,N4P1)
     * ,J=1,NP1)
      GO TO 3
    2 ISTAT=INSTAT(0.)
      WRITE(6,88) ITREC,ISTAT
      IF(ITREC.LE.IDREC) STOP
    3 IF(IEND.NE.0) STOP

C     INPUT PARAMETER VALUES DEFINING COMPUTATIONAL DATA CASE
C     TO BE EXECUTED.

C     ALPHA = VALUE OF THE GREEK LETTER OF THE SAME NAME USED IN
C             THE MATHEMATICAL DESCRIPTION OF THE PROBLEM TO BE SOLVED.
C       EPS = VALUE OF THE GREEK LETTER EPSILON USED IN THE MATHEMATICAL
C             DESCRIPTION OF THE PROBLEM TO BE SOLVED.

C        C1 = UNIFORM STARTING VALUE FOR OUTER ITERATION FOR PSI (IF
C             NOT READ FROM TAPE OR FROM STORAGE--SEE IUSE).
C        C2 = SAME AS C1 EXCEPT FOR OMEGA.

C      EPS1 = OUTER ITERATION CONVERGENCE TOLERANCE FOR PSI.
C      EPS2 = OUTER ITERATION CONVERGENCE TOLERANCE FOR OMEGA.

C        RP = INNER ITERATION OVER-RELAXATION FACTOR FOR PSI.
C        RW = INNER ITERATION OVER-RELAXATION FACTOR FOR OMEGA.

C     ISAVE = VARIABLE CONTROLLING WHETHER OR NOT A SOLUTION OBTAINED
C             FOR THIS DATA CASE IS TO BE SAVED IN MEMORY AND/OR ON
C             TAPE TO BE USED BY SUCCEEDING DATA CASES AND/OR LATER RUNS
C             AS OUTER ITERATION STARTING VALUES.
C             ISAVE =  0 IMPLIES DON'T SAVE SOLUTION OBTAINED FOR
C                        THIS CASE.
C             ISAVE =  1 IMPLIES SAVE SOLUTION OBTAINED FOR THIS
C                        CASE IN MEMORY ONLY.
C             ISAVE =  2 IMPLIES SAVE SOLUTION OBTAINED FOR THIS
C                        CASE ON DRUM ONLY.
C             ISAVE =  3 IMPLIES SAVE SOLUTION ON DRUM AND IN MEMORY.
C             ISAVE = 11 IMPLIES SAVE RESULT AFTER MAXIMUM NUMBER OF
C                        OUTER ITERATIONS IN MEMORY ONLY.
C             ISAVE = 12 IMPLIES SAVE RESULT AFTER MAXIMUM NUMBER OF
C                        OUTER ITERATIONS ON DRUM ONLY.
```

```
C             ISAVE = 13 IMPLIES SAVE RESULT AFTER MAXIMUM NUMBER OF
C                        OUTER ITERATIONS IN MEMORY AND ON DRUM.

C       IUSE = VARIABLE CONTROLLING WHETHER THE OUTER ITERATION STARTING
C              VALUES ARE TO BE TAKEN TO BE THE VALUES C1,C2 UNIFORMLY
C              FOR PSI AND OMEGA, RESPECTIVELY, OR WHETHER THEY ARE TO
C              BE READ FROM DRUM OR ANOTHER AREA OF MEMORY.
C              IUSE = 0 IMPLIES STARTING VALUES TAKEN AS C1,C2.
C              IUSE = 1 IMPLIES TAKE STARTING VALUES FROM MEMORY (IF
C                       SOME PREVIOUSLY COMPUTED SOLUTION IS KNOWN TO
C                       RESIDE THERE--OTHERWISE TAKE INITIAL VALUES FROM
C                       DRUM, UNLESS THERE WAS NO DRUM INPUT, IN WHICH
C                       CASE USE C1,C2 AS STARTING VALUES FOR A DEFAULT
C                       COURSE OF ACTION.
C              IUSE = 2 IMPLIES TAKE STARTING VALUES FROM DRUM INPUT,
C                       UNLESS THERE WAS NONE, IN WHICH CASE USE C1,C2
C                       AS STARTING VALUES FOR DEFAULT ACTION.

C      MXITER = MAXIMUM NUMBER OF OUTER ITERATIONS TO BE ALLOWED
C      FOR THIS CASE.

C       IEND = FLAG VARIABLE FOR LAST DATA CASE.
C              IEND .EQ. 0 IMPLIES MORE DATA CASES FOLLOW.
C              IEND .NE. 0 IMPLIES THIS IS THE LAST DATA CASE.

      READ(5,99,END=40) ALPHA,EPS,C1,C2,EPS1,EPS2,RHO,MU,RP,RW,ISAVE,
     * IUSE,MXITER,IEND
      IF(ABS(ALPHA-ALFCON).LT. 2.E-8 .AND. ABS(EPS-EPSCON).LT. 2.E-8)
     *  GO TO 3
      ALFCON=-1.
      EPSCON=-1.
    4 WRITE(6,98)        ALPHA,EPS,C1,C2,EPS1,EPS2,RHO,MU,RP,RW

C     COMPUTE VARIOUS CONSTANTS FOR THIS DATA CASE.

      ITST=MAX0(1,MXITER-4)
      H=PI/N
      HSQ=H*H
      HSQE=HSQ/EPS
      TWOH=2.*H
      AHE=ALPHA*H/EPS
      AHE4=4.*AHE
      HSQ2=2./HSQ
      RHO1=1.-RHO
      MU1(1)=1.-MU(1)
      MU1(2)=1.-MU(2)

C     SET SOR TOLERANCES FOR PSI (EP1) AND OMEGA (EP2).

      EP1=AMIN1(EPS1/1000.,5.E-5)
      EP2=AMIN1(EPS2/1000.,5.E-4)

      RP1=1.-RP
      RP4=RP/4.
      RP9=RP/9.
      RW1=1.-RW
      DO 5 J=2,N
    5 SINY(J)=SIN((J-1)*H)
C     INITIALIZE OUTER ITERATES.
      IF(IUSE.EQ.0) GO TO 6
      IF(IUSE.NE.1) GO TO 1105
      IF(ISCL.EQ.0) GO TO 1105
      L=4
      WRITE(6,89)
      GO TO 1205
 1105 IF(IDREC.EQ.0) GO TO 6
      L=5
      WRITE(6,84)
 1205 DO 105 J=1,NP1
      J2=N4P1-J+1
      DO 105 I=J,J2
      PSI(I,J,3)=PSI(I,J,L)
  105 OMEGA(I,J,3)=OMEGA(I,J,L)
      GO TO 8
    6 DO 7 J=2,N
      OMEGA(J,J,3)=C2
      OMEGA(N4P1-J+1,J,3)=C2
      JP1=J+1
      J2=N4P1-J
      DO 7 I=JP1,J2
      PSI(I,J,3)=C1
    7 OMEGA(I,J,3)=C2
    8 ITER=0
C     UPDATE PREVIOUS OUTER ITERATES.
   10 ITER=ITER+1
      DO 11 J=2,N
      OMEGA(J,J,1)=OMEGA(J,J,3)
      OMEGA(N4P1-J+1,J,1)=OMEGA(N4P1-J+1,J,3)
      HSQOM(J,J)=HSQ*OMEGA(J,J,1)
      HSQOM(N4P1-J+1,J)=HSQ*OMEGA(N4P1-J+1,J,1)
      JP1=J+1
      J2=N4P1-J
      DO 11 I=JP1,J2
      PSI(I,J,1)=PSI(I,J,3)
      OMEGA(I,J,1)=OMEGA(I,J,3)
   11 HSQOM(I,J)=HSQ*OMEGA(I,J,1)
      IT=0
C     COMPUTE PSI BY SUCCESSIVE OVER-RELAXATION.
   13 IT=IT+1
      DO 14 J=2,N
      JP1=J+1
      J2=N4P1-J
      DO 14 I=JP1,J2
   14 PSI(I,J,2)=PSI(I,J,3)
      ICONV=0
      DO 16 J=2,N
      JP2=J+2
      J2=N4P1-J-1
      PSI(J+1,J,3)=RP1*PSI(J+1,J,2)+RP9*PSI(JP2,J-1,3)
      IF(ABS(PSI(J+1,J,3)-PSI(J+1,J,2)).GT.EP1) ICONV=1
      DO 15 I=JP2,J2
      PSI(I,J,3)=RP1*PSI(I,J,2)+RP4*(PSI(I+1,J,2)+PSI(I,J+1,2)+
     * PSI(I-1,J,3)+PSI(I,J-1,3)+HSQOM(I,J))
      IF(ABS(PSI(I,J,3)-PSI(I,J,2)).GT.EP1) ICONV=1
   15 CONTINUE
      PSI(N4P1-J,J,3)=RP1*PSI(N4P1-J,J,2)+RP9*PSI(N4P1-J-1,J-1,3)
      IF(ABS(PSI(N4P1-J,J,3)-PSI(N4P1-J,J,2)).GT.EP1) ICONV=1
   16 CONTINUE
      IF(ICONV.EQ.0) GO TO 17
      IF(IT.LT.MAXIT) GO TO 13

C     SOR ITERATION FAILED TO CONVERGE. PRINT ERROR MESSAGE AND GO TO
C     NEXT DATA CASE.

      WRITE(6,97) ITER
      CALL PRINT(PSI(1,1,3))
      GO TO 3

C     CONVERGENCE OBTAINED FOR PSI. SMOOTH SOLUTION.

   17 ICONVP=0
      NITP=IT
      DO 18 J=2,N
      JP1=J+1
      J2=N4P1-J
      DO 18 I=JP1,J2
      PSI(I,J,3)=RHO*PSI(I,J,1)+RHO1*PSI(I,J,3)
      IF(ABS(PSI(I,J,3)-PSI(I,J,1)).GT.EPS1) ICONVP=1
   18 CONTINUE

C     COMPUTE OMEGA ON BOUNDARIES OA AND BC.

      ICONVW=0
      DO 20 J=2,N
      OMEGA(J,J,3)=-HSQ2*(PSI(J+1,J,3)+PSI(J,J-1,3))*MU1(1)
     $  +MU(1)*OMEGA(J,J,1)
      OMEGA(N4P1-J+1,J,3)=-HSQ2*(PSI(N4P1-J,J,3)+PSI(N4P1-J+1,J-1,3))
     $  *MU1(1)+MU(1)*OMEGA(N4P1-J+1,J,1)
      IF(ABS(OMEGA(J,J,3)-OMEGA(J,J,1)).GT.EPS2) ICONVW=1
      IF(ABS(OMEGA(N4P1-J+1,J,3)-OMEGA(N4P1-J+1,J,1)).GT.EPS2) ICONVW=1
      OMEGA(J,J,2)=OMEGA(J,J,3)
   20 OMEGA(N4P1-J+1,J,2)=OMEGA(N4P1-J+1,J,3)

C     COMPUTE OMEGA ON INTERIOR BY SUCCESSIVE OVER-RELAXATION.

C     COMPUTE COEFFICIENTS FOR SOR ITERATION.
```

```
      DO 22 J=2,N
      JP1=J+1
      J2=N4P1-J
      DO 22 I=JP1,J2
      A=(PSI(I,J+1,3)-PSI(I,J-1,3))/TWOH
      B=(PSI(I+1,J,3)-PSI(I-1,J,3))/TWOH
      C(I,J,1)=-AHE*(ABS(A)+ABS(B))-4.
      D(I,J)=HSQE*(B+SINY(J))/C(I,J,1)
      C(I,J,5)=(1.+AHE*AMAX1(B,0.))/C(I,J,1)
      C(I,J,3)=(1.-AHE*AMIN1(B,0.))/C(I,J,1)
      C(I,J,4)=(1.-AHE*AMIN1(A,0.))/C(I,J,1)
      C(I,J,2)=(1.+AHE*AMAX1(A,0.))/C(I,J,1)
   22 CONTINUE

C     SOR LOOP FOR OMEGA

      IT=0
   24 IT=IT+1
      DO 26 J=2,N
      JP1=J+1
      J2=N4P1-J
      DO 26 I=JP1,J2
   26 OMEGA(I,J,2)=OMEGA(I,J,3)
      ICONV=0
      DO 28 J=2,N
      JP1=J+1
      J2=N4P1-J
      DO 28 I=JP1,J2
      OMEGA(I,J,3)=RW1*OMEGA(I,J,2)-RW*(C(I,J,2)*
     $ OMEGA(I+1,J,2)+C(I,J,3)*OMEGA(I,J+1,2)+C(I,J,4)*OMEGA(I-1,J,3)
     $ +C(I,J,5)*OMEGA(I,J-1,3)+D(I,J))
      IF(ABS(OMEGA(I,J,3)-OMEGA(I,J,2)).GT.EP2) ICONV=1
   28 CONTINUE
      IF(ICONV.EQ.0) GO TO 30
      IF(IT.LT.MAXIT) GO TO 24

C     SOR ITERATION FAILED TO CONVERGE. WRITE ERROR MESSAGE AND GO TO
C     NEXT DATA CASE.

      WRITE(6,96) ITER
      CALL PRINT(OMEGA(1,1,3))
      GO TO 3

C     CONVERGENCE OBTAINED FOR OMEGA. SMOOTH SOLUTION.

   30 NITW=IT
      DO 32 J=2,N
      JP1=J+1
      J2=N4P1-J
      DO 32 I=JP1,J2
      OMEGA(I,J,3)=MU(2)*OMEGA(I,J,1)+MU1(2)*OMEGA(I,J,3)
      IF(ABS(OMEGA(I,J,3)-OMEGA(I,J,1)).GT.EPS2) ICONVW=1
   32 CONTINUE
```

```
C     PRINT OUT PERTINENT INFORMATION FOR CURRENT OUTER ITERATION.

      WRITE(6,93) ITER,NITP,NITW
      IF(ITER.GT.3) GO TO 33
      CALL PRINT(PSI(1,1,3))
      CALL PRINT(OMEGA(1,1,3))
      GO TO 133
   33 IF(ITER.GE.ITST) CALL PRINT(PSI(1,1,3))
      IF(ITER.GE.ITST) CALL PRINT(OMEGA(1,1,3))

C     TEST FOR CONVERGENCE OF OUTER ITERATION.

  133 IF(ICONVP+ICONVW .EQ.0) GO TO 34
      IF(ITER.LT.MXITER) GO TO 10

C     OUTER ITERATION FAILED TO CONVERGE. PRINT ERROR MESSAGE AND GO
C     TO NEXT DATA CASE.

      WRITE(6,95)
      IF(ISAVE.LT.11) GO TO 3
  233 ISAVE=MOD(ISAVE,10)
      GO TO 134

C     CONVERGENCE OBTAINED. PRINT MESSAGE AND GO TO NEXT DATA CASE.

   34 IF(ITER.LT.ITST) CALL PRINT(PSI(1,1,3))
      IF(ITER.LT.ITST) CALL PRINT(OMEGA(1,1,3))
      ALFCON=ALPHA
      EPSCON=EPS
      WRITE(6,94)
      GO TO 233
  134 IF(ISAVE.LE.0) GO TO 3
      IF(ISAVE.GT.3) GO TO 3
      IF(ISAVE.NE.2) GO TO 135

C     SAVE SOLUTION ON DRUM FILE TO BE WRITTEN TO TAPE.

   35 WRITE(12) ((PSI(I,J,3),I=1,N4P1),J=1,NP1),((OMEGA(I,J,3),I=1,N4P1)
     * ,J=1,NP1)
      ITREC=ITREC+1
      WRITE(6,87) ITREC
      GO TO 3

C     SAVE SOLUTION JUST OBTAINED IN CORE MEMORY.

  135 DO 36 J=1,NP1
      J2=N4P1-J+1
      DO 36 I=J,J2
      PSI(I,J,4)=PSI(I,J,3)
   36 OMEGA(I,J,4)=OMEGA(I,J,3)
      WRITE(6,86)
      ISOL=1
      IF(ISAVE.EQ.3) GO TO 35
      GO TO 3
C     TERMINATION POINT FOR PROGRAM. CONTROL REACHES HERE AFTER ALL DATA
C     HAS BEEN READ AND PROCESSEC.
   40 STOP

C     FORMAT STATEMENTS

   99 FORMAT(11E5.5,4I5)
   98 FORMAT(1H1 4X 'ALPHA =' F10.6,5X 'EPS =' F10.6//34X 'PSI' 8X
     * 'OMEGA'/14X'INITIAL VALUES' F9.0,F12.C/6X 'CONVERGENCE TOLERANCES
     *' 2F11.6/11X 'SMOOTHING FACTORS'F9.5,2F13.5/5X 'OVER-RELAXATION FA
     *CTORS' F9.2,F13.2//)
   97 FORMAT('0INNER ITERATION FAILEC FOR PSI IN OUTER ITERATION' I5)
   96 FORMAT('0INNER ITERATION FAILED FOR OMEGA IN OUTER ITERATION' I5)
   95 FORMAT('0OUTER ITERATION FAILEC TO CONVERGE')
   94 FORMAT('0OUTER ITERATION CONVERGED.'/)
   93 FORMAT('0PSI ANC OMEGA AT OLTER ITERATION' I5,5X 'INNERATICNS REQU
     *IRED =' 2I7/)
   89 FORMAT('0INITIAL VALLES FOR PSI AND OMEGA  TAKEN FROM EARLIER COMP
     *UTATION THIS RUN'/)
   88 FORMAT('0ENC OF FILE REACHEC AT RECORC NO.'I5,3X'STATUS ='3X O12)
   87 FORMAT('0SOLUTION JUST OBTAINED WAS SAVED ON TAPE AS RECORD NO'I5)
   86 FORMAT('0SOLUTION WILL BE SAVEC IN MEMORY FOR USE AS INITIAL VALUE
     *S IN LATER CASES THIS RUN')
   85 FORMAT('0INITIAL VALUES FROM TAPE TO BE USED WHERE NOTED'/)
   84 FORMAT('0INITIAL VALUES TAKEN FROM TAPE'/)

C     THIS ROUTINE PRINTS A FUNCTION IN THE TRAPEZCICAL FORMAT
C     CORRESPONDING TO THE GEOMETRY OF THE PHYSICAL PROBLEM.

      SUBROUTINE PRINT(A)
      DIMENSICN A(N4P1,NP1)
   99 FORMAT(1X  11F10.3)
      N1=1
      N2=N4P1
      K2=MIN0(NP1,11)
      K1=N4P1-K2+1
    2 DO 10 I=N1,N2
      IF(I.GT.K1) GO TO 5
      J2=MIN0(I,K2)
    3 WRITE(6,99) (A(I,J),J=N1,J2)
      GO TO 10
    5 J2=MIN0(N4P1-I+1,K2)
      GO TO 3
   10 CONTINUE
      IF(K2.EG.NP1) RETURN
      N1=12
      N2=N4P1-11
      K1=N3P1
      K2=NP1
      WRITE(6,99)
      GO TO 2
      END

@XQT
  .04 .005    0.    0. .050  .30.0025.0025.9975  1.7  1.0    3    2  150    0
  .15 .005    0.    0. .050  .30.0025.0025.9975  1.7  1.0    2    1  150    1

      FINAL COPYING OF OUTPUT DRUM FILE (12) TO TAPE FILE (10) IF THERE
      WAS DRUM OUTPUT FROM PROGRAM.


@COPY,GMC  12.,10.
@REWIND    10.
@FREE      10.

@FIN
```

**BIBLIOGRAPHIC DATA SHEET**

1. Report No.
WIS-CS-72-T.R. #149

2.

3. Recipient's Accession N

4. Title and Subtitle
A Numerical Approach to Wind-Driven Ocean Circulation

5. Report Date
March, 1972

6.

7. Author(s)

8. Performing Organization Rept. No.

9. Performing Organization Name and Address

10. Project/Task/Work Unit No.

11. Contract/Grant No.

12. Sponsoring Organization Name and Address
Computer Sciences Department
University of Wisconsin
1210 West Dayton Street
Madison, Wisconsin 53706

13. Type of Report & Period Covered

14.

15. Supplementary Notes

16. Abstracts

A numerical method is developed for a widely studied, wind-driven ocean circulation model. Examples of flow patterns of the northern Pacific, which include large non-linear effects, are given.

17. Key Words and Document Analysis. 17a. Descriptors

Fluid Dynamics
Ocean-Circulation
Navier-Stokes Equations
Partial-Differential Equations
Numerical Analysis

17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group

18. Availability Statement
Available to Public.

19. Security Class (This Report)
UNCLASSIFIED

20. Security Class (This Page
UNCLASSIFIED

21. No. of Pages
33

22. Price