M. BROWER

A Reproduced Copy

OF

NASA CR-166,436

Reproduced for NASA

by the

NASA Scientific and Technical Information Facility

NASA CONTRACTOR REPORT 166436

(NASA-CR-166436) FREE WAKE TECHNIQUES FOR ROTOR AERODYNAMIC ANALYSIS. VOLUME 3: VORTEX FILAMENT MODELS (Massachusetts Inst. of Tech.) 69 p HC A04/MF A01 CSCL 01A

N83-19713

Unclas
G3/02 09171

Free Wake Techniques for Rotor Aerodynamic Analysis; Volume No. III - Vortex Filament Models

Michael Brower

CONTRACT NAG2-47
December 1982

NASA

NASA CONTRACTOR REPORT 166436

Free Wake Techniques for Rotor Aerodynamic Analysis; Volume No. III - Vortex Filament Models

Michael Brower
Department of Aeronautics and Astronautics
Massachusetts Institute of Technology
Cambridge, Massachusetts

Prepared for Ames Research Center under
NASA Cooperative Agreement No. NAG2-47

NASA
National Aeronautics and
Space Administration

Ames Research Center
Moffett Field, California 94035

TABLE OF CONTENTS

NOMENCLATURE

C_T	thrust coefficient
R	rotor radius
r	radius to vortex
w_z	induced velocity in wake in z direction
x	horizontal displacement
z	vertical displacement
Γ	circulation
Ω	rotational speed
ε	core size
$\Delta\theta$	defined in Fig. 1-2

SUMMARY

This report is Volume III of a three volume series entitled "Free Wake Techniques for Rotor Aerodynamic Analysis" and covering the following topics:

Volume I (Ref. 3) "Summary of Results and Background Theory" reviews the results obtained to date using both complete and simplified wake models and summarizes the theoretical background on which these models are based.

Volume II (Ref. 2) "Vortex Sheet Models" presents the results of computations using vortex sheets to model the wake and tests the sensitivity of the solutions to various assumptions used in the development of the models. The complete codings are included.

Volume III (present volume) "Vortex Filament Models" discusses results obtained using a vortex filament model, as opposed to sheets, again using various modelling techniques and including the computer codings.

INTRODUCTION

The application of computational fluid dynamics to the calculation of airloads on a hovering helicopter rotor is much more difficult than for an aircraft wing in steady flight. This is mainly because the trailing vortex geometry is more important to the induced velocities on a helicopter blade than on a wing. Whereas a wing in straight flight leaves its wake far behind, a hovering rotor gathers it beneath itself in a spiral that remains close to the blade. Thus the induced velocities must be calculated everywhere in the wake as well as on the blade in order to give the correct bound circulation. The close proximity of concentrated vortices often makes convergence slow and the results extremely sensitive to vortex core size and other parameters.

This report presents the results of airloads calculations for one case of a hovering rotor for which good experimental data are available. The method used allows the trailing wake geometry to evolve freely according to the velocities calculated at discrete control points in the wake. This is known as a free-wake method. A loop calculates in turn the induced velocities, the geometry and the bound circulation, iterating until convergence is reached.

The method is loosely based on an original code written by J.D. Gohard (Ref. 1). That program has been developed by A. Tanuwidjaja (Ref. 2). In addition a simplified free-wake analysis has been done by R.H. Miller (Ref. 3 and 4) in which the wake immediately following the blade is made up of semi-infinite vortex lines and the wake beneath the blade consists of circular vortex rings. In order to reproduce the experimentally observed bound circulation both these programs require assumptions in calculating the wake geometry and core size in the first 180 degrees after the blade, as discussed in Ref. 4. This is a problem with the present method as well, as will be discussed in this report, and should become a focus of further research.

The program presented here is a rewritten version of the Tanuwidjaja-Gohard program. In describing the wake, vortex sheets have been eliminated in favor of concentrated vortex filaments; some other minor changes have been made, and the program has been "cleaned up" to make it less confusing and to remove parts no longer used.

Section 1 of this report presents the free-wake method in detail. Section 2 discusses the results of the program for the test case as different parameters are varied. The appendix contains the computer codes.

I. THE FREE-WAKE MODEL

1. Method

Each helicopter blade is treated as a concentrated line vortex of varying strength along the span. It is divided into discrete segments of constant vorticity where the strength of each segment is found from the angle of attack at its midpoint according to the formula:

$$\Gamma = \pi u \alpha c$$

At the join of any two segments a trailing vortex is formed perpendicular to the blade with a strength equal to the difference in bound circulation on either side. (Figure 1-1). These trailing vortices are responsible for the downwash on the blade which, along with the blade pitch, give the angle of attack.

Since the wake moves down as the rotor turns, the trailing wake forms a helix beneath the rotor plane. Each trailing spiral vortex in the helix is divided into straight vortex segments of equal angular length joined end-to-end. Thus the actual length of a segment depends on its distance from the hub (Figure 1-2). The control points where induced velocities are calculated are on the vortex lines where two segments meet. The location of a particular segment in the wake is found by integrating the velocities up to that point on the vortex line to which it belongs.

The whole wake is divided into three sections, the near, intermediate and far wakes (Figure 1-3). The near wake retains all the individual trailing vortices from the blade to obtain the best resolution in calculating the bound circulation. After some azimuthal angle, usually less than 90 degrees, the intermediate wake rolls up the near wake into three concentrated vortices. The strength of a rolled-up vortex is the sum of the strengths of the near wake vortices contributing to it; its first azimuthal position is the centroid of the last azimuthal positions of the same near wake vortices:

$$\vec{x} = \frac{\Sigma \vec{x}_i \cdot \Gamma_i}{\Sigma \Gamma_i}$$

After several blade revolutions the intermediate wake is replaced by the far wake. The far wake is modeled as three semi-infinite vortex cylinders corresponding to the three vortices in the intermediate wake. (For a discussion of the effects of far wake modeling see Ref. 5, p. 16.) The start of the far wake is one vertical spacing underneath the last azimuthal station of the intermediate wake, and the radius is assumed to stay constant. There are no control points in the far wake and no induced velocities are calculated there.

The induced velocities due to the near and intermediate wakes are found using the Biot-Savart law for straight vortex segments with a correction for a viscous core size. The core allows the induced velocity to go to zero as a vortex line is approached instead of going to infinity. The induced velocity due to each segment in the wake and on the blade is calculated at every control point. These calculations consume the most computer time. A typical wake contains two hundred segments or 40,000 applications of the Biot-Savart relation for every iteration.

The induced velocities due to the far wake are found from an elliptic integral series solution used by Miller (Ref. 3).

The self-induced velocity of a straight vortex segment is zero if a finite core size is used. But the trailing vortices are really locally curved, and a correction to take this curvature into account is needed. This has been derived by Scully (Ref. 5) based on the self-induced velocity of a vortex ring of finite core size found by Lamb (Ref. 6). The formula is:

$$\omega_z = \frac{\Gamma}{4\pi r}\left(8\cdot \ln\left(\frac{r}{\varepsilon}\cdot \tan\left(\frac{\Delta\theta}{4}\right)\right) - .25\right)$$

where $\Delta\theta$ is the angular length of the segments on either side of the control point. It is important to note that this is accurate only in cases where the arc length of the segments is large compared to their core size. This is true in the outer part of the wake where typical arc lengths are 0.1R and core sizes are .01R, but it becomes questionable closer to the root. The exact geometry close to the rotor hub is much less important to the bound circulation, however, and these corrections are small compared to the total induced velocities.

When overall convergence is finally reached, the main program calculates the lift coefficient. This coefficient, the final bound circulation and the wake geometry are compared with the experimental results.

2. Parameters

The important parameters to be varied in the program are the mesh sizes on the blade and in the near and intermediate wakes, the number of azimuthal stations in each wake, the criterion for roll-up in the intermediate wake, the tip loss factor and the viscous core size.

a) Mesh Size

The usual practice is for the bound vortex segments on the blade to be long inboard and short near the tip where the circulation changes rapidly. In all the cases presented here, the distance between adjacent control points on the blade is .02R from the ninety-percent span out to the tip, .05R from the eighty-percent span to the ninety-percent span and .1 or .15R over the rest of the blade. In all, thirteen control points (twelve segments) are used.

The near wake has the same number of trailing vortices as the number of bound segments requires; in this case, thirteen. The angular length of each segment is ten degrees. Usually the wake is carried 70 degrees after the blade. The intermediate wake is made up of twenty-degree segments and is usually carried about 740 degrees, or four blade passages, past the end of the near wake.

b) Roll-Up Criterion

The tip vortex in the intermediate wake is rolled up from the outermost trailing vortex in the near wake to the point of maximum circulation on the blade. The middle vortex is rolled up three stations in from where the tip roll-up ends, or about the 70% span. (As shown in Ref. 2, the solution is not sensitive to the exact definition of this station.) The root vortex is rolled up over the rest of the blade, excluding the innermost vortex of the near wake. It is found that eliminating the root vortex in the near wake tends to smooth out the distribution of bound circulation on the inner half of the blade and makes convergence easier.

c) Tip Loss

The tip loss factor compensates for the reduced lift at the very tip of the blade which is not well accounted for by the lifting line method. It simply sets the bound circulation at the last spanwise station equal to some fraction of the value it would have if the usual lifting line formula were used. This fraction is zero for all cases presented here, that is, the bound circulation of the last segment starting at 98% of the blade is set to zero; this is similar to the treatment of Ref. 7.

d) Viscous Core Size

not for general ... [illegible]

The core size is probably the most important single parameter. It has a very large effect on the velocity induced by vortex segments on nearby points. For example, if the core size of the tip vortex is set equal to .02R, this causes the bound circulation to go up at the last spanwise station because the tip vortex no longer induces enough downwash there to pull it down. It makes no difference to the circulation over the rest of the blade. In this study, a core size of .01R is always used at the tip; it may be larger in the rest of the wake.

It is difficult to determine what core sizes should be used in the program, and few experimental guides are available. In general, the results seem overly dependent on the core. One area that should be studied is how the "proper" core size depends on the mesh size. In the example given above, if the distance between stations were made larger, a larger core size could be used (and vice-versa) and yet the true core size should not change. For a further discussion of the effect of core size on the tip vortex in the near wake see Ref. 4.

II. RESULTS

Table 1 shows the characteristics of the rotor being studied. The experimental data were published in Ref. 8.

a) Fundamental Solution

The "fundamental solution" uses a core size of .01R throughout the wake. It carries the near wake over 70 degrees and the intermediate wake over 740 degrees. The solution is given in Figure 2-1 along with the experimental results; the bound circulation is plotted against the span and the tip vortex position is shown for every 180 degrees.

There are three major problems with the solution. One is that the bound circulation is too low from about the 70% span out to the tip. This gives a lift coefficient (CT) of about -.00448 instead of the experimental value of -.0046. Another problem is that the tip vortex remains too close to the blade in the first 180 degrees. The third problem is that the wake defined by the tip vortex does not contract enough; it even begins to expand after 540 degrees.

b) Wake Contraction

The wake does not contract enough either because the intermediate wake produces too much radial velocity or because the far wake produces too little. By removing the inner vortices in the intermediate and far wakes the wake contraction can be improved (Figure 2-2) but not without sacrificing bound circulation near the root, which yields a much lower lift coefficient. It is interesting to note that the circulation near the tip is unaffected. This implies that wake contraction has little to do with the rest of the problem.

c) Core Size (Near Wake)

The only way to raise the bound circulation near the tip is to increase the viscous core size in the near wake. All three parts of the wake, near, intermediate and far, contribute an important fraction of the downwash on the outer 25% of the blade. The far wake is too distant for any reasonable increase in the core size to have an effect. The core size of the intermediate wake can be important if it is larger than about .04R; this possibility, known as core burst, is discussed below. The near wake vortex segments are close enough for a small change in the core size to have a large effect.

The result of increasing the core size in the near wake (but not, as mentioned in Section 1, at the tip) from .01R to .02R is shown in Figure 2-3. The peak is higher and the tip vortex has accordingly moved down somewhat. The circulation in the "valley" between the 70% and 90% span is still too low.

d) Core Burst

It has been observed that a concentrated tip vortex passing close to a blade undergoes a sudden expansion of its core (Ref. 9). This fact can be exploited to improve the distribution of bound circulation. The intermediate wake tip vortex produces strong upwash on the blade from the 90% span out to the tip, and downwash everywhere else. Thus an increase in core size will at the same time raise the circulation in the valley and lower the circulation at the peak. This is illustrated in Figure 2-4, where the assumed core burst is to .05R and the core size in the near wake is .02R. The wake geometry stays the same.

In order to raise the peak back up again, the core size in the near wake must be increased to .05R. (It quickly reaches the point where a larger near-wake core size makes no difference.) This is shown in Figure 2-5. This last solution is the best obtained so far. The lift coefficient is -.00453, about two percent lower than the experimental value.

e) Tip Vortex Location

The deficiency in circulation between the 70% and 90% span is possibly caused by the tip vortex coming too close to the blade at first encounter. The "natural" location of the tip vortex predicted by the program is closer to the blade than was observed in Ref. 8. It may be that the slow downward motion of the tip vortex is caused by inadequate modeling of the roll-up in the near wake, possibly by not having enough trailing vortices at the tip.

One way to test this is to reduce the near wake azimuth to only ten degrees - essentially rolling up the tip vortex immediately. The results are shown in Figure 2-6 for the same case as Figure 2-5 (core burst to .05R and near wake core size .05R). The circulation and the lift coefficient are slightly larger, and the tip vortex location is a little farther from the blade. The solutions for other combinations of parameters are similar. This suggests that poor modeling of the near wake roll-up could be part of the problem.

In the earlier results from this program using a near wake spanning seventy degrees, the problem of the tip vortex location was much worse. It was discovered that this was because the program had not been allowed to converge properly. The initial approximation to the wake geometry does not roll up the vortices in the near wake. During the first few iterations, the main trailing vortex actually moves up. If the convergence test for the geometry is not tight enough the program will converge before the near wake has had a chance to move back down. It usually takes over 60 iterations to converge within two percent.

CONCLUSIONS

The results of the program so far indicate that further investigation of the method is necessary. With a small core size, the bound circulation is too low on the outer part of the blade while the tip vortex passes too close at first encounter. The two problems are closely related. The slow descent of the tip vortex may be because of near wake modelling deficiency. Assuming a larger core size, corresponding to a burst vortex, results in better agreement with test data. A further area needing research is the relationship between the core size and the mesh size.

REFERENCES

1 J.D. Gohard, "Free Wake Analysis of Wind Turbine Aerodynamics," MIT ASRL TR 184-14, September 1978.

2 A. Tanuwidjaja, "Free Wake Techniques for Rotor Aerodynamic Analysis, Volume II: Vortex Sheet Models," MIT ASRL Report 199-2, December 1982.

3 R.H. Miller, "Simplified Free Wake Analyses for Rotors," FFA (Sweden) TN 1982-7; also MIT ASRL TR 194-3.

4 R.H. Miller, "Free Wake Techniques for Rotor Aerodynamic Analysis, Volume I: Summary of Results and Background Theory," MIT ASRL Report 199-1, December 1982.

5 M. Scully, "Computation of Helicopter Rotor Wake Geometry and Its Influence on Rotor Harmonic Airloads," MIT ASRL TR 178-1, March 1975.

6 H. Lamb, HYDRODYNAMICS, Dover, 1945.

7 W. Johnson, "Comparison of Calculated and Measured Model Rotor Loading and Wake Geometry," NASA TM 81189, April 1980.

8 J.D. Ballard, K.L. Orloff and A. Luebs, "Effect of Tip Shape on Blade Loading Characteristics and Wake Geometry for a Two-Bladed Rotor in Hover," Journal of the American Helicopter Society, Vol. 25, No. 1, pp. 30-35, January 1980.

9 T.C. Biggers, A. Lee, K.L. Orloff and O.T. Lemmer, "Measurements of Helicopter Rotor Tip Vortices," American Helicopter Society Forum Proceedings, May 1977.

Table 1. ROTOR CHARACTERISTICS

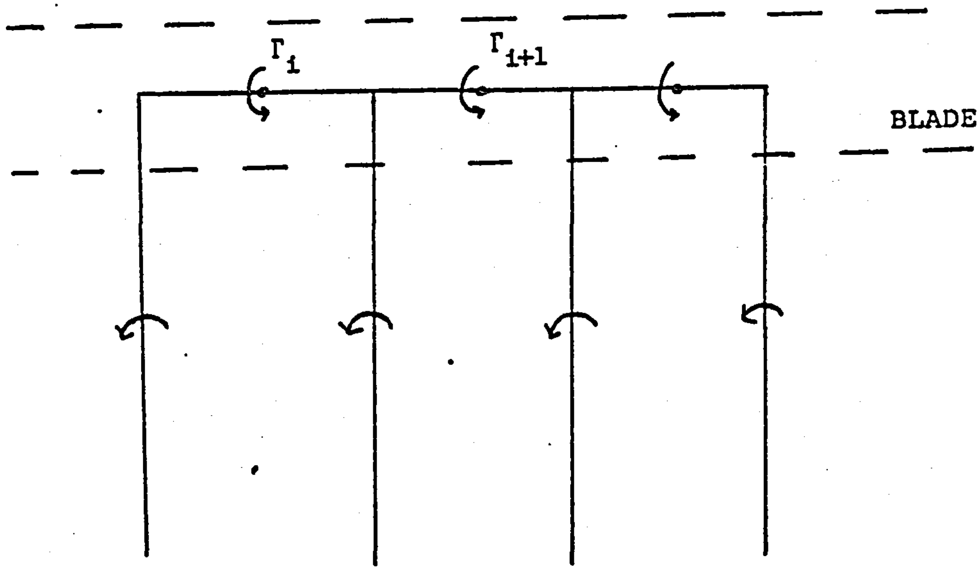
Straight

Blades - 2

Twist - 11°

Collective pitch at 75% Radius 9.8°

Solidity .0464



$\Gamma_+ = \Gamma_{i+1} - \Gamma$

Fig. 1-1 Lifting line bound segments and trailing vortices.

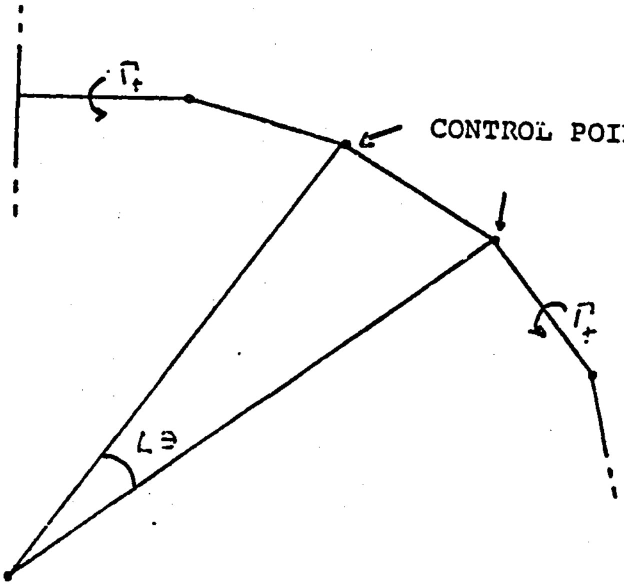


Fig. 1-2 Trailing vortices divided into straight segments.

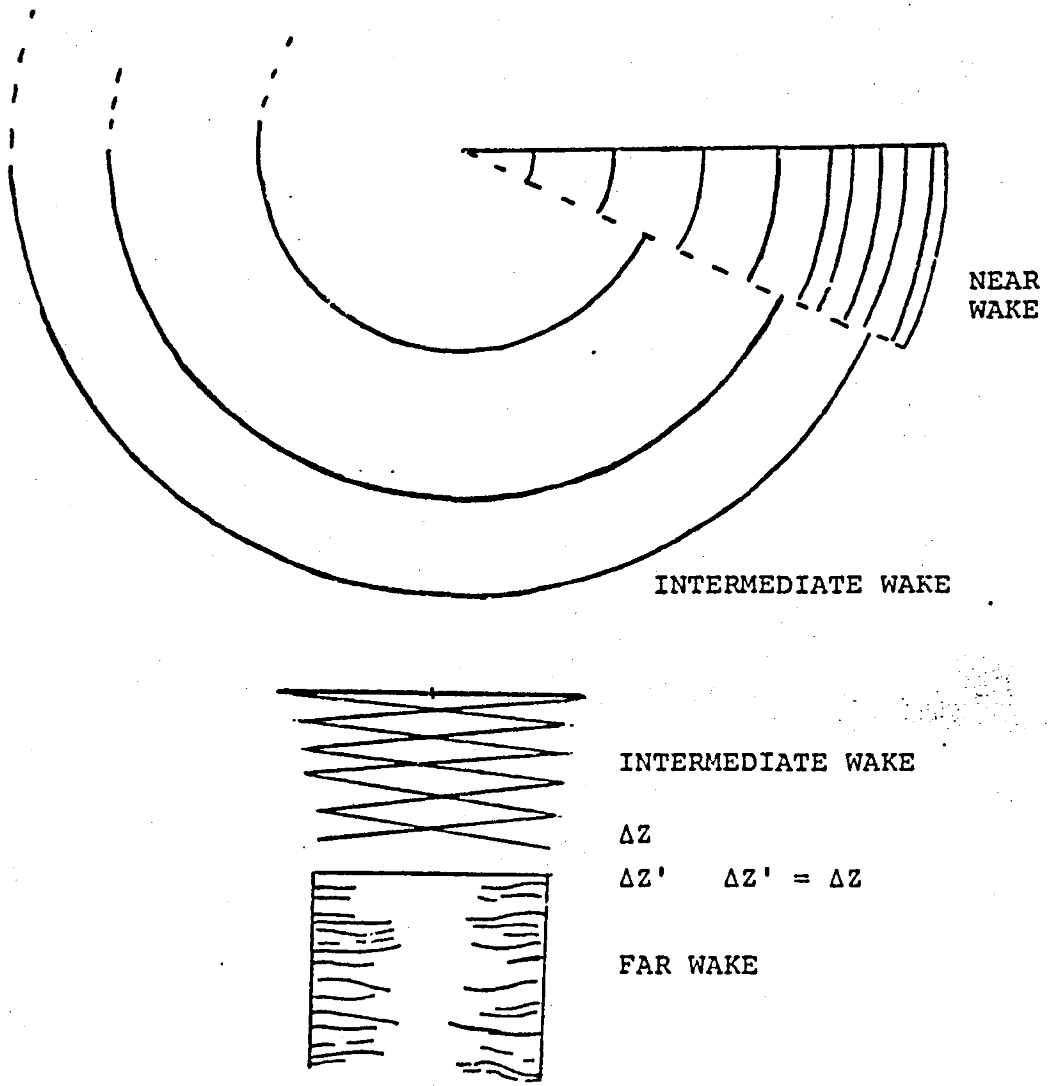


Fig. 1-3   Near, intermediate and far wakes.

Fig. 2-1 Bound circulation distribution with core size of .01R everywhere in wake. C_T = .00452
Near Wake extends to 70°, intermediate wake to 740°

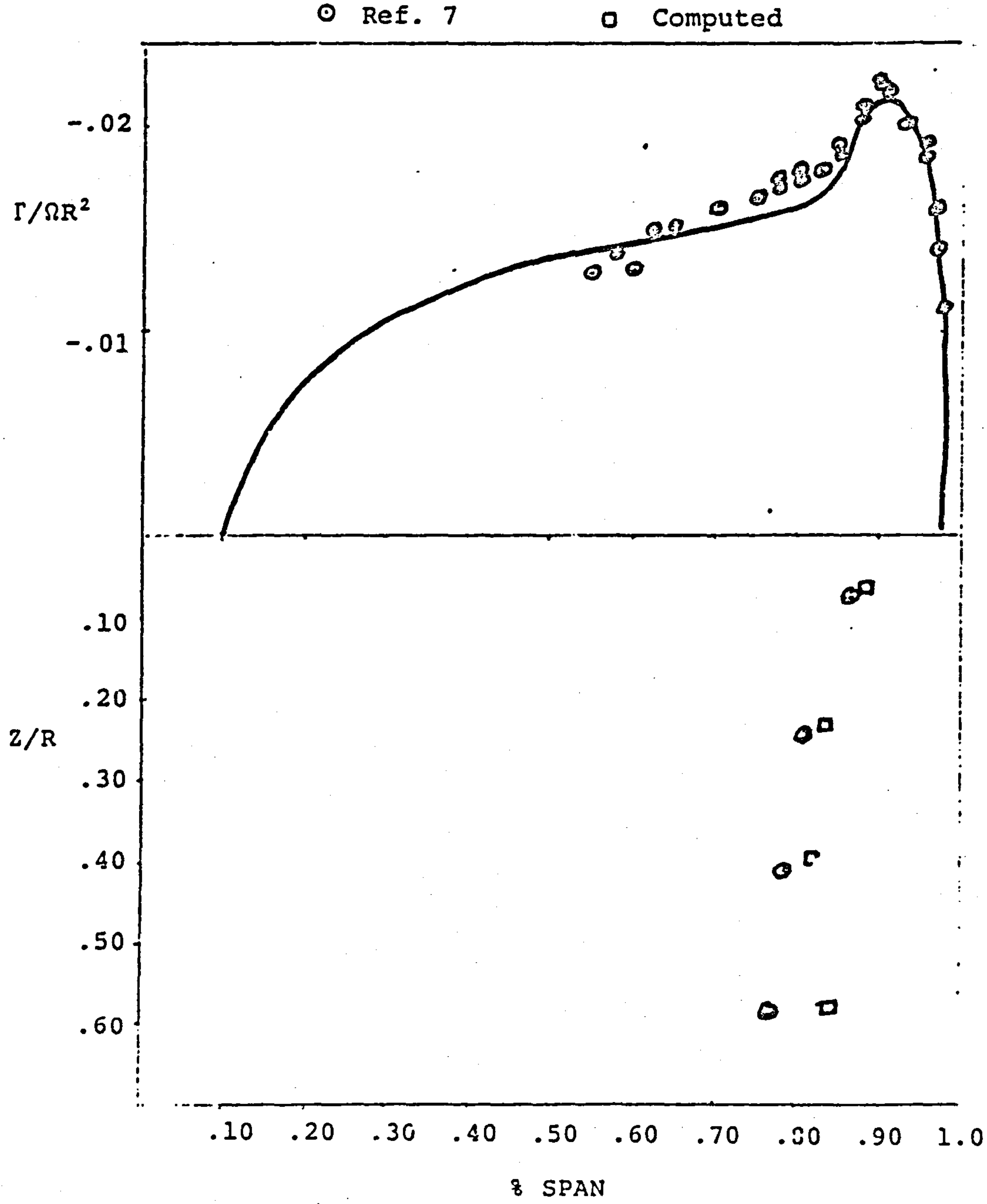


Fig. 2-2 Effect of inner wake on case of Fig. 2-1.

□ Inner wake eliminated.

Δ Far wake, inner wake only eliminated.

x Same as Fig. 2-1.

⊙ Ref. 7

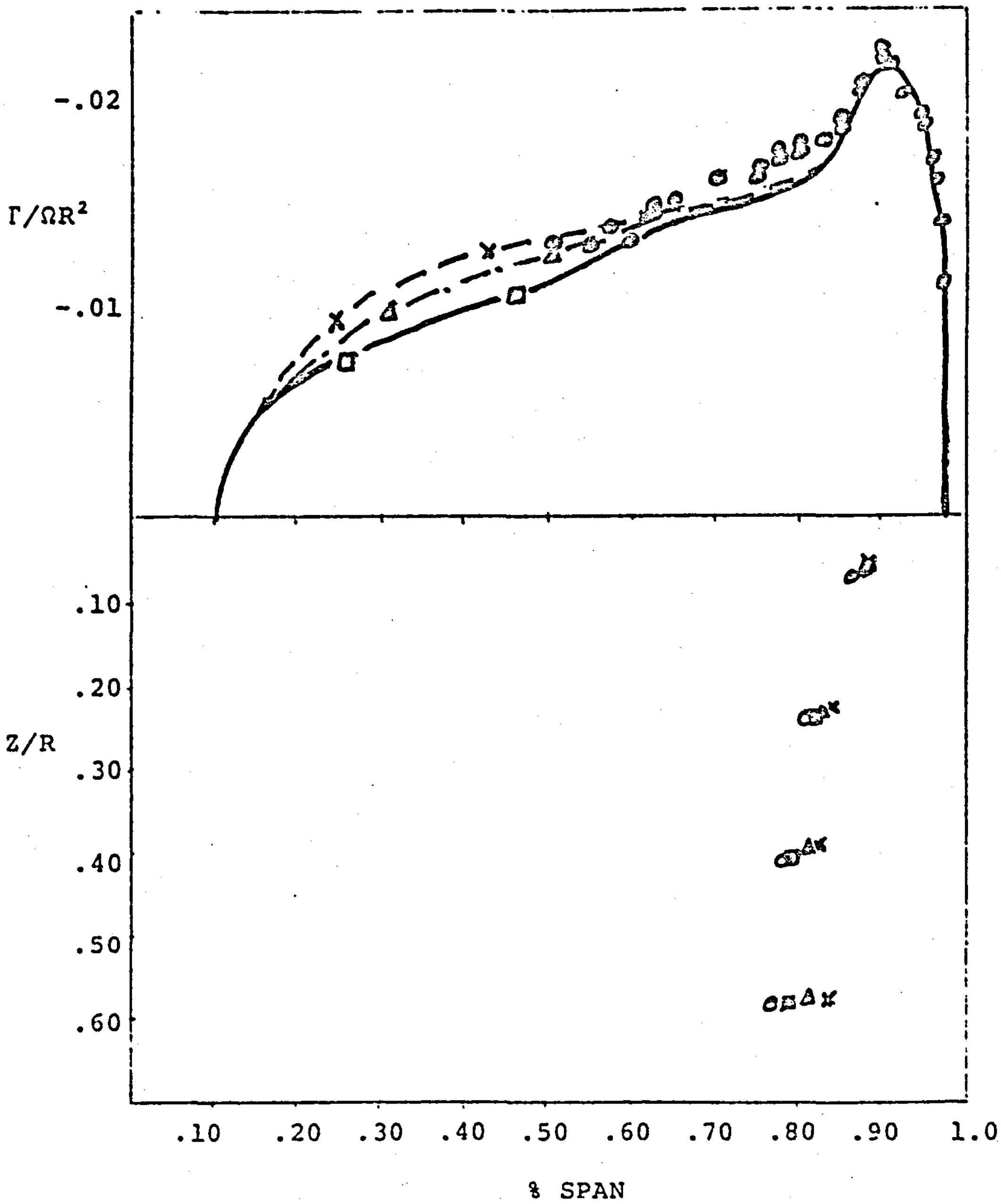


Fig. 2-3 Effect of core size for case of Fig. 2-1.
$C_T = .00452$
Inner near wake core size of .02R
Tip vortex and intermediate wake of .01R
⊙ Ref. 7 ◻ Computed

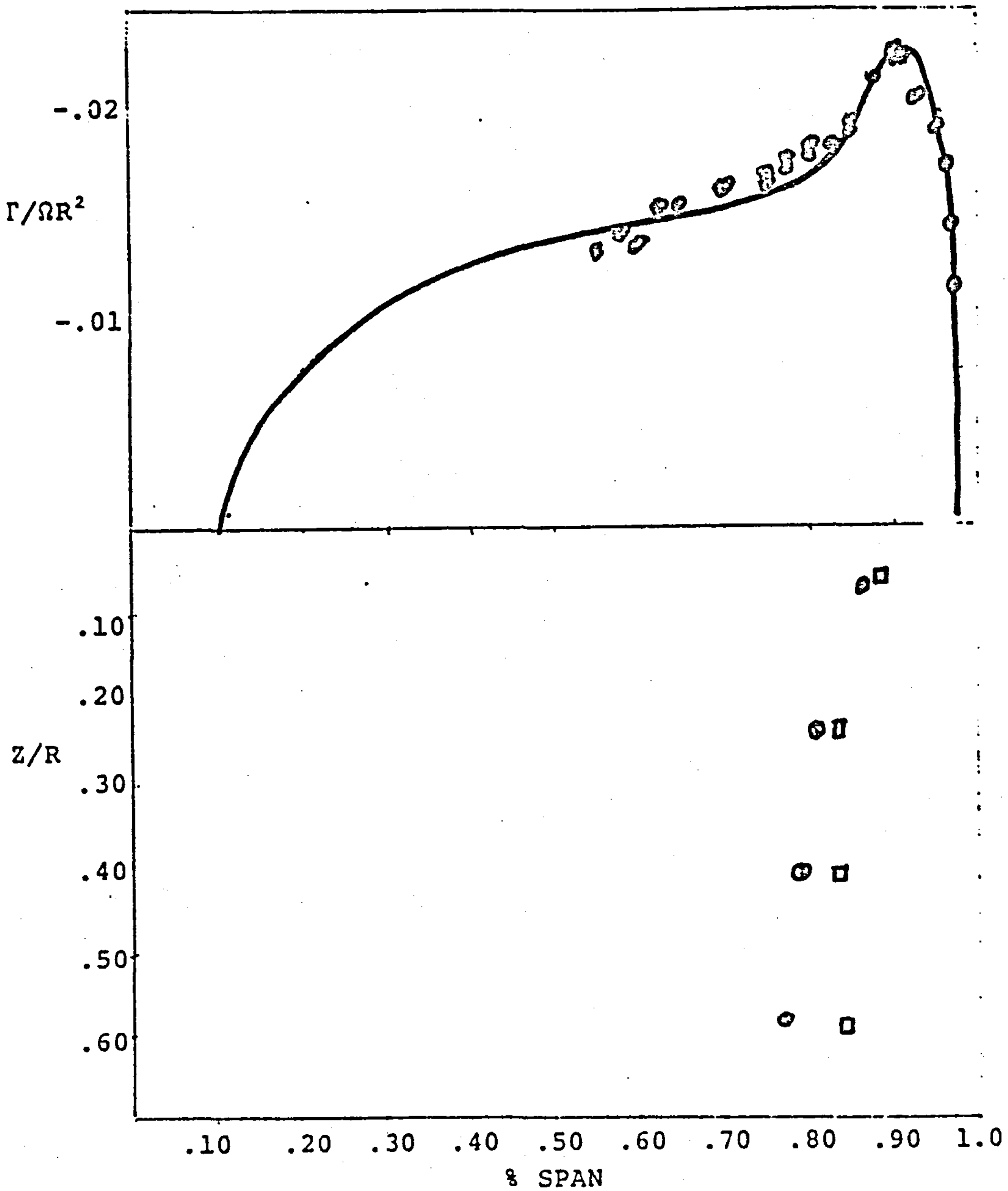


Fig. 2-4 Same as Fig. 2-3 but with "burst" tip vortex.
Core size of .05R in intermediate wake. C_T = .00451
⊙ Ref. 7 ◻ Computed

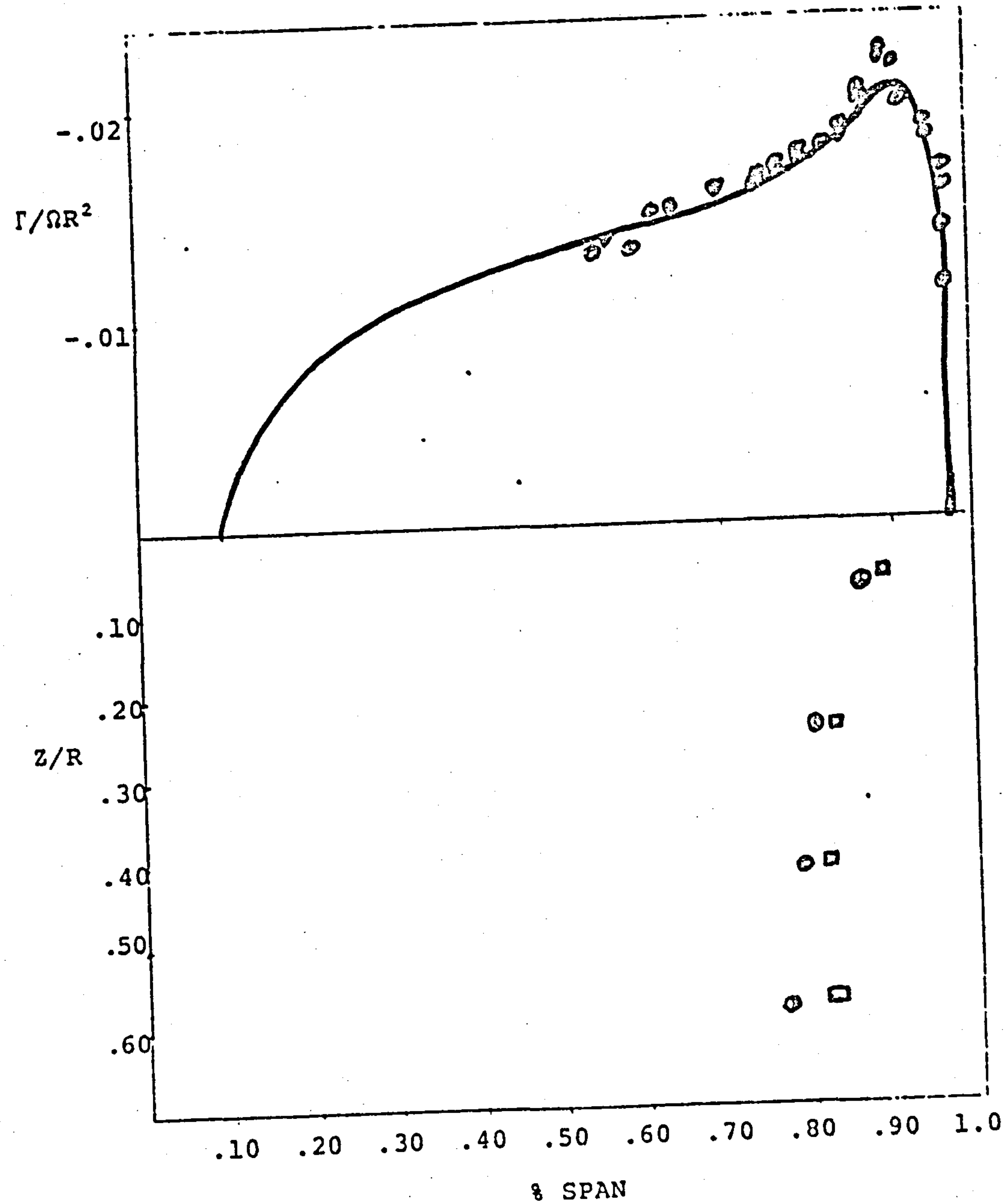
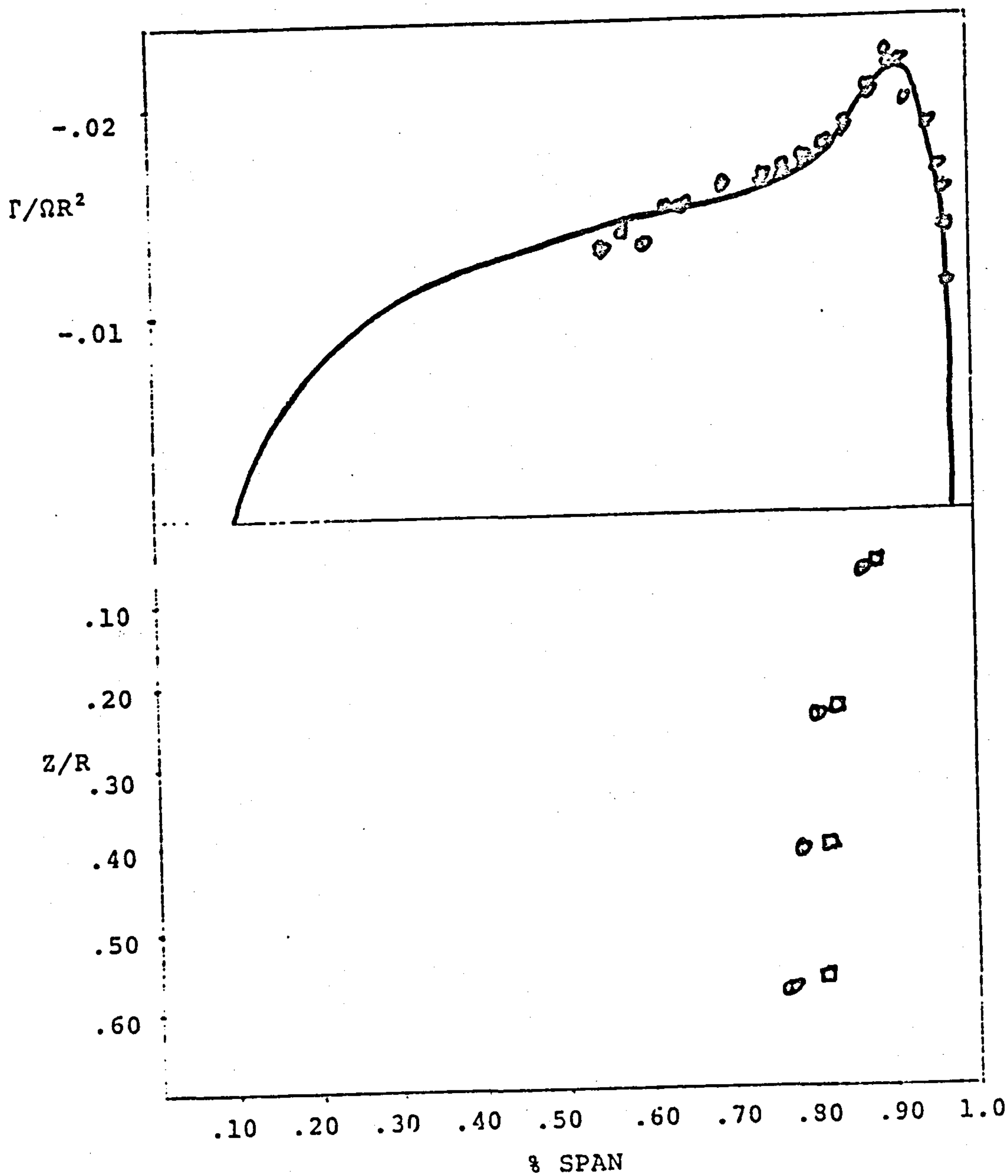


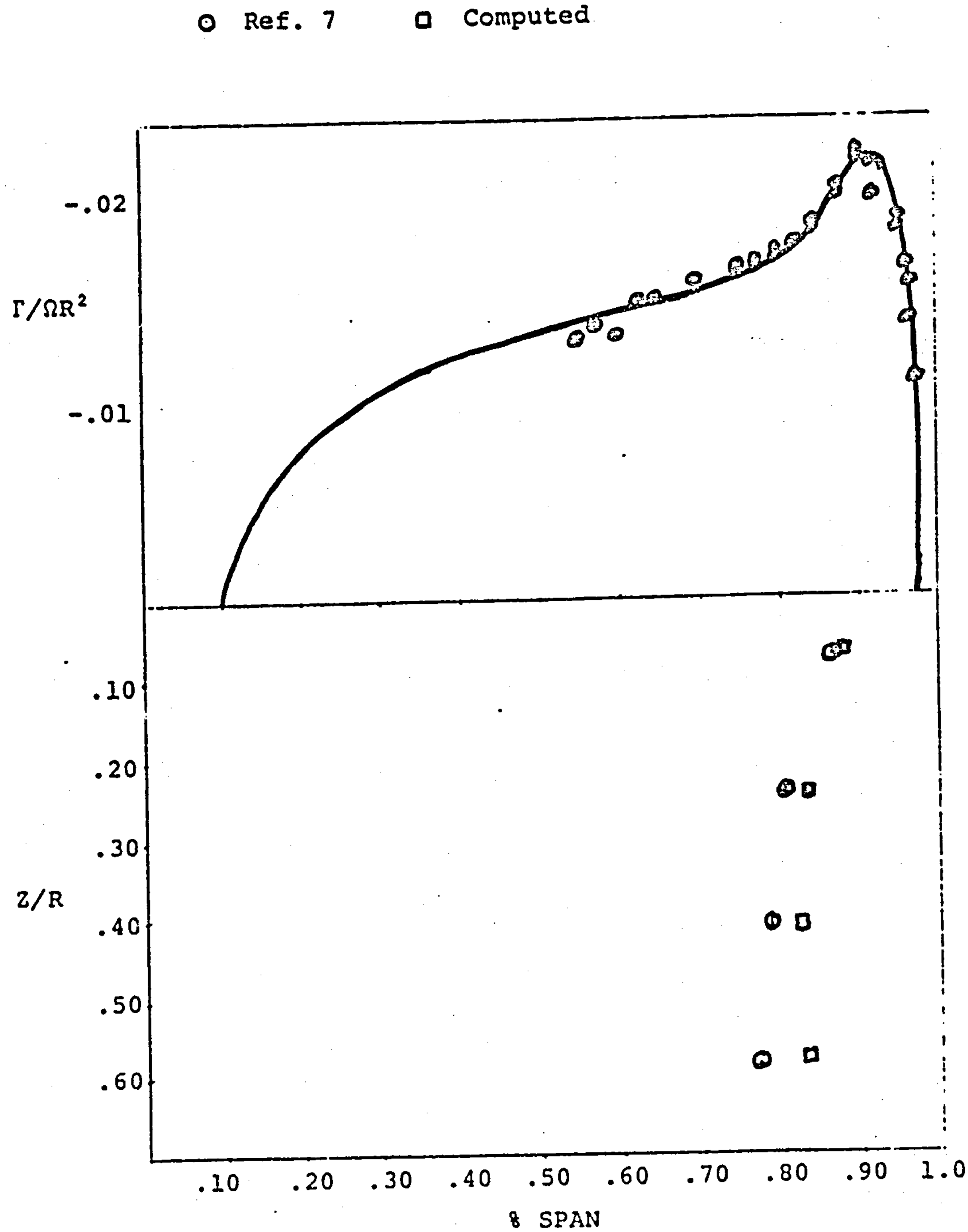
Fig. 2-5 Same as Fig. 2-4 but with "burst" tip and inner intermediate wake. C_T = -.00453

⊙ Ref. 7 □ Computed



ORIGINAL PAGE IS
OF POOR QUALITY

Fig. 2-6 Same as Fig. 2-5 but near wake extending to 10° only, intermediate wake to 800°. $C_T = .00454$



APPENDIX

LIST OF VARIABLES

Following is a list of important variables in the FreeWake program.

alp	angle of attack, alp=flamda-thetac
alphas	angle of attack at stall
blades	real number of blades
cd	(cd0,cdk) drag coefficients, cd=cd0 + cdk*alp*alp
coeff	(coeff1) tip loss coefficient
dpsii	angular length of intermediate wake segments
dpsin	angular length of near wake segments
etanv	spanwise location of endpoints of bound segments
etaiv	initial spanwise location of intermediate wake trailing vortices
eta1 eta2 eta3	spanwise location of intermediate wake trailers
facgam	relaxation factor for LOOP1
facgeom	relaxation factor for INTGR
facvel	relaxation factor for LOOP2
flamda	inflow angle due to downwash on blade
fmu	net inflow -- used only for wind turbine cases

fps1	core size for near wake
fps2	core size for tip vortex in near wake
fps3	core size of intermediate wake (core burst)
gamc	bound circulation
gamt	trailing vortex strengths in near wake
gamti	rolled-up vortex strengths in intermediate wake
gamtip	same as gamti(3)
int	parameter for intermediate output: int=1 prints lift coefficient and wake geometry for every fifth iteration
itest	itest=0 if all subroutines have converged
iwrite	parameter for output: iwrite=0 gives only final results, iwrite=1 gives results for every iteration, iwrite=2 gives induced velocities from each part of wake (near, intermediate and far) for each iteration
knnvr	initial value of nnvr
knivr	initial value of nivr
lim1	maximum number of iterations for LOOP1
lim2	maximum number of iterations for main loop
lp	power coefficient, or drag integrated over span
lt	lift coefficient, or lift integrated over span
nblds	(nblds1) integer number of blades
ncase	number of different cases in file to be run
niter	current iteration number of main loop
niva	number of azimuthal stations in intermediate wake
nivr	number of spanwise stations in intermediate wake

nnva	number of azimuthal stations in near wake
nnvr	number of spanwise stations in near wake
sigma	solidity of rotor
theta	blade pitch at endpoints of bound segments
thetac	blade pitch at midpoints of bound segments
wxiv wyiv wziv	x,y,z components of induced velocities in intermediate wake
wxnv wynv wznv	x,y,z components of induced velocities in near wake
xiv yiv ziv	x,y,z coordinates of control points in intermediate wake
xnv ynv znv	x,y,z coordinates of control points in near wake

At the end of the computer code a sample output is given (for Figure 2-5) showing the lift coefficient, bound circulation, induced velocity on the blade and wake geometry.

The bound circulation (GAMC) and induced velocities (WXC, WYC, WZC) are given at the centers of the bound vortex segments whereas the trailing vortices (which give the wake geometry) are located at the ends.

The integer headings in the wake geometry refer to the azimuthal station number. The first line under each heading is the azimuthal angle (PSI), the second is the radius (R) and the third is the vertical distance (Z) from the rotor plane. The first azimuthal station is at the blade, so here PSI=0., Z=0. and R gives the spanwise positions of the ends of the bound vortex segments.

ORIGINAL PAGE IS
OF POOR QUALITY

Computer Codes

```
c    *  FREE-WAKE ANALYSIS OF A WIND TURBINE AERODYNAMICS
c    *  -------------------------------------------------
c    *
        common/parm/iwr,ird,lims,lim1,lim2;niter,iwrite
        common/geom/nblds1,nblds,sigma,fmu,etan(15),knnvr,etai(6),knivr,
         &itwist,thetad(15),theta(15),thetac(14),theta0,thet0d,alphas,
         &cd0,cdk,dpsind,dpsin,dpsiid,dpsii,coeff,coeff1,c,s,blades,
         &nnvr,nnva,nnvr1,nnva1,nncr,nnca,etanv(15),etanc(16),
         &nivr,niva,nivr1,niva1,nicr,nica,etaiv(6),etaic(7),
        &ntva,ntva1,ntca,fps1,fps2
        common/gamma/gamc(16),gamt(15),gamti(3),gamtip,eta1,eta2,eta3,
         &k1,k2,k3,facgam
        common/posit/xnv(15,18),ynv(15,18),znv(15,18),xiv(6,50),yiv(6,50),
         &ziv(6,50)
        common/veloc/wxnv(15,18),wynv(15,18),wznv(15,18),wxiv(6,50),
          &wyiv(6,50),wziv(6,50),facgeom
        common/initial/wxc(14),wyc(14),wzc(14)
        common/vindo/ifar,nfar,x1,x2,y1,y2,z1,z2,x,y,z,facvel,gm,ux,uy,uz
        common/self/iself
        common/loopsave/wxv(16),wyv(16),wzv(16)
        common/special/itgr
        common/extraspec/fps3
        namelist/fparm/iwr,ird,lims,lim1,lim2,iwrite,nfar,ifar,gam,wx,wy,
         &wz,facgam,facvel,facgeom,ncase,iself,int,itgr
        real lip,lp,lt
        data pi/3.14159/
        iwr=6
        ird=5
        iself=0
        coeff=0.
        nblds1=2
        alphas=1.
        coeff1=.5
        lims=50
        lim1=50
        lim2=15
        nfar=90
        ifar=0
        int=0
        read(ird,fparm)
        do 901 ncas=1,ncase
        call input1(ncas)
        if(fps3.ne.fps2) write(iwr,53) fps3
53      format(" CORE BURST TO ",f5.2)
        if(ifar.eq.1) write(iwr,54)
54      format(" ELLIPTIC INTEGRAL FAR WAKE ")
        write(iwr,57) coeff1
57      format(" TIP COEFFICIENT ",f5.2)
        if(itgr.eq.1)write(iwr,58)
38      format(" ROLLED UP TRAILERS MEET AT POINT")
        do 10 i=1,nnvr
        et=etanv(i)
        wxc(i)=wx*et*et
        wyc(i)=0.
        wzc(i)=et*wz/.85
        gamc(i+1)=gam*et/.91
        if(et.gt..91) wzc(i)=(wz*et/.85)*(1.-et)/.15
        if(et.gt..91) gamc(i+1)=(gam*et/.91)*(1.-et)/.09
10      continue
        wzc(nnvr)=wzc(nnvr-1)
```

```
      gamc(nnvr)=0.
      gamc(1)=0.
      gamc(nncr)=0.
      call compa
c   *
c   * MAIN LOOP
c   * ---------
c   *
      kwrite=iwrite
      do 1 niter=1,lim2
      itest=0
      facgeom=.5
      facvel=.3
      if(niter.le.2) facgeom=1.
      if(kwrite.eq.0) iwrite=0
      fiter=niter/5.
      fniter=float(niter/5)
      if(fniter.eq.fiter.and.kwrite.eq.0) iwrite=1
      if(niter.eq.1.and.kwrite.eq.0) iwrite=1
      call loop2(ktest)
      itest=ktest
      call intgr(ktest)
      itest=itest+ktest
      call loop1(ktest)
      itest=itest+ktest
      if(itest.eq.0) goto 2
      if(int.eq.1.and.iwrite.gt.0) goto 1003
1004  continue
1     continue
2     continue
c   *
      write(iwr,100) itest
      write(iwr,108)
      write(iwr,120) (gamc(i),i=1,nnvr)
      write(iwr,120) (wxv(i),i=1,nnvr)
      write(iwr,120) (wyv(i),i=1,nnvr)
      write(iwr,120) (wzv(i),i=1,nnvr)
      write(iwr,107)
      do 150 j=1,nnva
      write(iwr,109) j
      write(iwr,120) (xnv(i,j),i=1,nnvr)
      write(iwr,120) (ynv(i,j),i=1,nnvr)
      write(iwr,120) (znv(i,j),i=1,nnvr)
150   continue
      do 160 j=1,niva
      write(iwr,109) j
      write(iwr,120) (xiv(i,j),i=1,nivr)
      write(iwr,120) (yiv(i,j),i=1,nivr)
      write(iwr,120) (ziv(i,j),i=1,nivr)
160   continue
1003  continue
      lt=0.
      lp=0.
      do 148 i=1,nnvr-1
      et=.5*(etanv(i)+etanv(i+1))
      wz=wzv(i)+fmu
      wy=wyv(i)+et
      flamda=atan(wz/wy)
      u=sqrt(wz**2+wy**2)
      alp=flamda-thetac(i)
```

```
        cd=cd0+cdk*alp**2
        if(abs(alp).gt.abs(alphas)) cd=2.*cd0+cdk*alp**2
        lip=u*gamc(i+1)
        tlip=lip*cos(flamda)
        flip=-lip*sin(flamda)
        dp=u*u*pi*(sigma/blades)*cd*.5
        fdp=dp*cos(flamda)
        tdp=dp*sin(flamda)
        tp=tlip+tdp
        fp=flip+fdp
        lt=lt+tp*(etanv(i+1)-etanv(i))
        lp=lp+fp*(etanv(i+1)**2-etanv(i)**2)
148     continue
        lt=lt*blades/pi
        lp=lp*blades/2./pi
        write(iwr,106) lt,lp
        if(int.eq.1.and.itest.ne.0.and.niter.lt.lim2) goto 1004
        iwrite=2
        call loop2(ktest)
901     continue
100     format(//,"  MAIN FINAL RESULTS -- CONVERGENCE:",i4)
109     format(/,i4)
108     format(/,"  GAMC,WXV,WYV,WZV:")
106     format(/,"  LT=",f10.6,"  LP=",f10.6)
107     format(//,"  WAKE GEOMETRY:")
120     format(2(9f10.6,/))
999     continue
        stop
        end
```

ORIGINAL PAGE IS
OF POOR QUALITY

```
      subroutine input1(ncas)
      common/parm/iwr,ird,lims,lim1,lim2,niter,iwrite
      common/geom/nblds1,nblds,sigma,fmu,etan(15),knnvr,etai(6),knivr,
     &itwist,thetad(15),theta(15),thetac(14),theta0,thet0d,alphas,
     &cd0,cdk,dpsind,dpsin,dpsiid,dpsii,coeff,coeff1,c,s,blades,
     &nnvr,nnva,nnvr1,nnva1,nncr,nnca,etanv(15),etanc(16),
     &nivr,niva,nivr1,niva1,nicr,nica,etaiv(6),etaic(7),
     &ntva,ntva1,ntca,fps1,fps2
      common/extraspec/fps3
      equivalence (nnvr,nncr1),(nivr,nicr1),(nnva,nnca1)
      equivalence (niva,nica1),(nnvr1,nncr2),(nivr1,nicr2)
      namelist/case/nblds,sigma,knnvr,nnva,dpsind,etan,knivr,
     &niva,dpsiid,etai,fmu,thetad,alphas,cd0,cdk,
     &fps1,fps2,coeff1,fps3
      data conv,twopi/.017453293,6.283185308/
c   *
      read(ird,case,err=999,end=888)
      dpsin=dpsind*conv
      dpsii=dpsiid*conv
      do 11 i=1,knnvr
      theta(i)=thetad(i)*conv
11    continue
      blades=float(nblds)
      c=cos(twopi/blades)
      s=sin(twopi/blades)
77    continue
c   *
      nnvr=knnvr
      nivr=knivr
      nncr=nnvr+1
      nica=niva+1
      nicr=nivr+1
      niva1=niva-1
      nnvr1=nnvr-1
      nivr1=nivr-1
      do 65 i=1,nnvr
      etanv(i)=etan(i)
65    continue
      do 66 i=1,nivr
      etaiv(i)=etai(i)
66    continue
      do 12 i=2,nnvr
      etanc(i)=(etanv(i)+etanv(i-1))*.5
12    continue
      do 13 i=2,nivr
      etaic(i)=(etaiv(i)+etaiv(i-1))*.5
13    continue
      etanc(1)=1.5*etanv(1)-.5*etanv(2)
      etaic(1)=1.5*etaiv(1)-.5*etaiv(2)
      etanc(nncr)=1.5*etanv(nnvr)-.5*etanv(nnvr-1)
      etaic(nicr)=1.5*etaiv(nivr)-.5*etaiv(nivr-1)
c   *
      nnva1=nnva-1
      nnca=nnva+1
      do 32 i=1,nncr2
      thetac(i)=theta(i)+(theta(i+1)-theta(i))*
     &(etanc(i+1)-etanv(i))/(etanv(i+1)-etanv(i))
32    continue
      if(nblds.lt.1.or.nblds.gt.8) goto 60
      if(nnvr.gt.15.or.nnvr.lt.2) goto 60
```

```
          if(nivr.gt.6.or.nivr.lt.2) goto 60
          if(nnva.gt.18.or.nnva.lt.2) goto 60
          if(niva.gt.50.or.niva.lt.2) goto 60
          if(etanv(1).lt.0.) goto 60
          if(etanv(1).ne.etaiv(1)) goto 60
          if(etanv(nnvr).ne.1..or.etaiv(nivr).ne.1.) goto 60
          do 80 i=2,nnvr
          if(etanv(i).le.etanv(i-1)) goto 60
80        continue
          do 81 i=2,nivr
          if(etaiv(i).le.etaiv(i-1)) goto 60
81        continue
          if(fmu.lt.0.) go to 60
          write(iwr,150)ncas,nblds,sigma,fmu
150       format("1*** INPUT *** CASE # :",i3,/,
          &" NUMBER OF BLADES",i2,/," SIGMA",f12.5,/,
          &" MU",f12.5)
          write(iwr,162)(thetad(i),i=1,knnvr)
162       format(" PITCH ANGLE DISTRIBUTION(DG):",5f10.5,/,5x,6f10.5,
          &/,5x,4f10.5)
          write(iwr,153)alphas,cd0,cdk
153       format(" STALL ANGLE",f12.5,/,"   CD=",f5.4,"+",f5.3,"*ALPHA**2")
          write(iwr,154)lims,lim1,lim2,fps1,fps2
154       format(" MAX. NUMBER OF ITERATIONS FOR SEMRI,LOOP1,LOOP2:",3i4,/,
          &" CORE SIZE FOR NEAR WAKE  INBOARD: ",f6.3,"  TIP: ",f6.3)
          write(iwr,155)knnvr,nnva,(etan(i),i=1,knnvr)
155       format(" NEAR WAKE DEFINITION:(",i2,",",i2,"),",4f10.5,
          &/,5x,6f10.5,/,4f10.5)
          write(iwr,156)knivr,niva,(etai(i),i=1,knivr)
156       format(" INT. WAKE DEFINITION:(",i2,",",i2,"),",4f10.5,
          &/,5x,6f10.5,/,5x,4f10.5)
          return
999       write(iwr,200)
200       format(" *** INPUT *** ERROR ON (IRD) ")
          stop
c   *
888       write(iwr,215)
215       format("  INPUT:  END OF FILE ON (IRD)")
          stop
102       format(8f10.5)
122       format(3i2)
c   *
60        continue
          write(iwr,181)
181       format(" *** INPUT *** DATA INVALID OR OUT OF RANGE ")
          write(iwr,case)
          stop
c   *
          end
```

```
      subroutine compa
      common/parm/iwr,ird,lims,lim1,lim2,niter,iwrite
      common/geom/nblds1,nblds,sigma,fmu,etan(15),knnvr,etai(6),kr
     &itwist,thetad(15),theta(15),thetac(14),theta0,thet0d,alpha,
     &cd0,cdk,dpsind,dpsin,dpsiid,dpsii,coeff,coeffi,c,s,cladem,
     &nnvr,nnva,nnvr1,nnva1,nncr,nnca,etanv(15),etanc(16),
     &nivr,niva,nivr1,niva1,nicr,nica,etaiv(6),etaic(7),
     &ntva,ntva1,ntca,fps1,fps2
      common/posit/xnv(15,18),ynv(15,18),znv(15,18),xiv(6,50),
     &yiv(6,50),ziv(6,50)
      common/initial/wxc(14),wyc(14),wzc(14)
      dimension wrn(15),wri(6),wzn(15),wzi(5)
      equivalence (nnvr,nncr1),(nivr,nicr1),(nnva,nnca1)
      equivalence (niva,nica1),(nnvr1,nncr2),(nivr1,nicr2)
      data pi/3.14159/
c    *
      do 21 i=1,nnvr
      xnv(i,1)=etanv(i)
      ynv(i,1)=0.
      znv(i,1)=0.
      phi=0.
      r=etanv(i)
      do 21 j=1,nnva-1
      phi1=phi+dpsin
      r1=r+dpsin*wxc(i)
      xnv(i,j+1)=r1*cos(phi1)
      ynv(i,j+1)=r1*sin(phi1)
      znv(i,j+1)=znv(i,j)+dpsin*wzc(i)
      phi=phi1
      r=r1
21    continue
      if(iwrite.lt.2) goto 103
      write(iwr,500)
500   format(//,". INITIAL WAKE GEOMETRY:")
      do 91 j=1,nnva
      write(iwr,505) j
      write(iwr,510) (xnv(i,j),i=1,nnvr)
      write(iwr,510) (ynv(i,j),i=1,nnvr)
      write(iwr,510) (znv(i,j),i=1,nnvr)
91    continue
103   continue
505   format(/,i4)
510   format(1x,2(5x,9f10.6,/))
c    *
c    *  NODES AND VELOCITIES FOR THE INTERMEDIATE WAKE
c    *
      do 30 i=1,nivr
      et=etaiv(i)
      do 31 i2=2,nnvr
      if(et.lt.etanv(i2)) goto 32
31    continue
32    continue
      a=(etanv(i2)-et)/(etanv(i2)-etanv(i2-1))
      wzi(i)=a*wzc(i2-1)+(1.-a)*wzc(i2)
      wri(i)=a*wxc(i2-1)+(1.-a)*wxc(i2)
      xiv(i,1)=a*xnv(i2-1,nnva)+(1.-a)*xnv(i2,nnva)
      yiv(i,1)=a*ynv(i2-1,nnva)+(1.-a)*ynv(i2,nnva)
      ziv(i,1)=a*znv(i2-1,nnva)+(1.-a)*znv(i2,nnva)
      psi=dpsin*(nnva-1)
      r=sqrt(xiv(i,1)**2+yiv(i,1)**2)
```

```
      do 22 j=1,niva-1
      psi1=psi+dpsi1
      wrr=wr1(i)
      if(psi1.gt.pi) wrr=wrr/2.
      if(psi1.gt.2.*pi) wrr=wrr/2.
      if(psi1.gt.3.*pi) wrr=wrr/2.
      if(psi1.gt.4.*pi) wrr=wrr/2.
      r1=r+dpsi1*wrr
      xiv(i,j+1)=r1*cos(psi1)
      yiv(i,j+1)=r1*sin(psi1)
      wz=wzi(i)
      if(psi1.gt.pi.and.i.eq.nivr) wz=2.*wz
      ziv(i,j+1)=ziv(i,j)+wz*dpsi1
      r=r1
      psi=psi1
22    continue
30    continue
      if(iwrite.lt.2) goto 104
      do 92 j=1,niva
      write(iwr,505) j
      write(iwr,510) (xiv(i,j),i=1,nivr)
      write(iwr,510) (yiv(i,j),i=1,nivr)
      write(iwr,510) (ziv(i,j),i=1,nivr)
92    continue
104   continue
      return
      end
```

```
      subroutine loop2(ktest)
c
c     This subroutine calculates the induced velocities at all
c     the points in the near and intermediate wake.  Only trailing
c     segment elements are used.  The wake is rolled up into three
c     trailing vortices starting at the intermediate wake.  Only the
c     ends of the segments are considered -- the positions of the
c     centers are never used or even calculated.
c
      real k2
      common/parm/iwr,ird,lims,limt,lim2,niter,iwrite
      common/geom/nbldsi,nblds,sigma,fmu,etan(15),knnvr,etai(6),knivr,
     &itwist,thetad(15),theta(15),thetac(14),theta0,thet0d,alphas,
     &cd0,cdk,dpsind,dpsin,dpsiid,dpsii,coeff,coeffi,c,s,blades,
     &nnvr,nnva,nnvri,nnvai,nncr,nnca,etanv(15),etanc(16),nivr,
     &niva,nivri,nivai,nicr,nica,etaiv(6),etaic(7),
     &ntva,ntvai,ntca,fps1,fps2
      common/gamma/gamc(16),gamt(15),gamti(3),gamtip,eta(3),
     &j1,kk(2),facgam
      common/veloc/wxnv(15,18),wynv(15,18),wznv(15,18),wxiv(6,50),wyiv(6,50),
     &wziv(6,50),facgeom
      common/vindo/ifar,nfar,x1,x2,y1,y2,z1,z2,x,y,z,facvel,gm,ux,uy,uz
      common/posit/xnv(15,18),ynv(15,18),znv(15,18),xiv(6,50),yiv(6,50),
     &ziv(6,50)
      common/save/wxnvs(15,18),wynvs(15,18),wznvs(15,18),wxivs(6,50),
     &wyivs(6,50),wzivs(6,50)
      common/self/iself
         common/extraspec/fps3
      data twopi,fpi/6.28318,.079577/
c     TRAILING VORTEX STRENGTHS CALCULATED BY SUBROUTINE ROLLUP
c
      call rollup(2)
      if(iwrite.lt.1)goto 5
      write(iwr,100) niter
      write(iwr,123) (gamc(i),i=1,nnvr)
      write(iwr,123) (gamt(i),i=1,nnvr)
      write(iwr,123) (gamti(i),i=1,nivr)
      write(iwr,110) (eta(i),i=1,3)
5     continue
c
c     INITIALIZE ARRAYS FOR NEW INDUCED VELOCITIES
c
      do 10 i=1,nnvr
      do 10 j=1,nnva
      wxnv(i,j)=0.
      wynv(i,j)=0.
      wznv(i,j)=0.
10    continue
      do 12 i=1,nivr
      do 12 j=1,niva
      wxiv(i,j)=0.
      wyiv(i,j)=0.
      wziv(i,j)=0.
12    continue
c
c     VELOCITIES INDUCED BY BLADE BOUND CIRCULATION
c
      xi=etanv(i)
      yi=0.
      zi=0.
```

```
      y2=0.
      z2=0.
      do 20 i=1,nnvr-1
      x2=etanv(i+1)
      gm=-gamc(i+1)
      n1=1
      n2=i+1
      call vindn(n1,n2,2)
      call vindi(n1,n2)
      x1=x2
20    continue
      if(iwrite.ne.2) goto 22
      do 21 j=1,nnva
      write(iwr,119) j
      write(iwr,120) (wxnv(k,j),k=1,nnvr)
      write(iwr,120) (wynv(k,j),k=1,nnvr)
      write(iwr,120) (wznv(k,j),k=1,nnvr)
21    continue
      do 23 j=1,niva
      write(iwr,119) j
      write(iwr,120) (wxiv(i,j),i=1,nivr)
      write(iwr,120) (wyiv(i,j),i=1,nivr)
      write(iwr,120) (wziv(i,j),i=1,nivr)
23    continue
22    continue
c
c     VELOCITIES INDUCED BY NEAR WAKE
c
      dps2=fps2
      do 30 i=1,nnvr
      fps2=fps1
         if(i.eq.nnvr-1) fps2=dps2
      do 30 j=1,nnva-1
      x1=xnv(i,j)
      y1=ynv(i,j)
      z1=znv(i,j)
      x2=xnv(i,j+1)
      y2=ynv(i,j+1)
      z2=znv(i,j+1)
      gm=gamt(i)
      n1=(j-1)*nnvr+1
      n2=n1+nnvr
      call vindn(n1,n2,2)
      if(j+1.eq.nnva) n2=0
      call vindi(n1,n2)
30    continue
      fps2=dps2
      if(iwrite.ne.2) goto 32
      do 31 j=1,nnva
      write(iwr,119) j
      write(iwr,120) (wxnv(k,j),k=1,nnvr)
      write(iwr,120) (wynv(k,j),k=1,nnvr)
      write(iwr,120) (wznv(k,j),k=1,nnvr)
31    continue
      do 33 j=1,niva
      write(iwr,119) j
      write(iwr,120) (wxiv(i,j),i=1,nivr)
      write(iwr,120) (wyiv(i,j),i=1,nivr)
      write(iwr,120) (wziv(i,j),i=1,nivr)
33    continue
```

```
32    continue
c
c     INTERMEDIATE WAKE
c
      phib=twopi/blades
      do 40 i=1,nivr
          if(i.eq.1) ksave=niva
          if(i.eq.2) ksave=niva
          if(i.eq.3) ksave=niva
      do 40 j=1,ksave-1
      x1=xiv(i,j)
      y1=yiv(i,j)
      z1=ziv(i,j)
      x2=xiv(i,j+1)
      y2=yiv(i,j+1)
      z2=ziv(i,j+1)
      gm=gamti(i)
      dps2=fps2
      phii=float(nnva-1)*dpsin + float(j-1)*dpsii
      if(phii.gt.phib-dpsin) fps2=fps3
      n1=(j-1)*nivr + nnva*nnvr + i
      n2=n1+nivr
      if(j.eq.1) n1=0
      call vindn(n1,n2,2)
      call vindi(n1,n2)
      fps2=dps2
40    continue
      if(iwrite.ne.2) goto 42
      do 41 j=1,nnva
      write(iwr,119) j
      write(iwr,120) (wxnv(i,j),i=1,nnvr)
      write(iwr,120) (wynv(i,j),i=1,nnvr)
      write(iwr,120) (wznv(i,j),i=1,nnvr)
41    continue
      do 43 j=1,niva
      write(iwr,119) j
      write(iwr,120) (wxiv(i,j),i=1,nivr)
      write(iwr,120) (wyiv(i,j),i=1,nivr)
      write(iwr,120) (wziv(i,j),i=1,nivr)
43    continue
42    continue
c
c     FAR WAKE
c
      if(ifar.eq.1) goto 45
      dz=ziv(nivr,niva)-ziv(nivr,niva-1)
      gm=gamtip
      x1=xiv(nivr,niva)
      y1=yiv(nivr,niva)
      z1=ziv(nivr,niva)
      phii=atan2(y1,x1)
      r=sqrt(x1*x1 + y1*y1)
      do 50 j=1,nfar
      phii=phii+dpsii
      x2=r*cos(phii)
      y2=r*sin(phii)
      z2=z1+dz
      n1=j + nnva*nnvr + niva*nivr
      n2=n1+1
      call vindn(n1,n2,2)
```

```
      call vindi(n1,n2)
      x1=x2
      y1=y2
      z1=z2
50    continue
      goto 49
45    continue
      fkapa=phib/dpsi1
      kapa=ifix(fkapa + .5)
      do 300 k1=1,nivr
      x1=xiv(k1,niva)
      y1=yiv(k1,niva)
      z1=ziv(k1,niva)+(ziv(k1,niva)-ziv(k1,niva-kapa))
      dgam=gamt1(k1)/(ziv(k1,niva)-ziv(k1,niva-kapa))
      r=(sqrt(x1*x1 + y1*y1))
      do 46 i=1,nnvr
      do 46 j=1,nnva
      n=1
      x=xnv(i,j)
      y=ynv(i,j)
      z=znv(i,j)
      z3=abs(z1-z)
      phi=atan2(y,x)
      eta3=sqrt(x*x + y*y)
      goto 200
210   continue
      wxnv(i,j)=wxnv(i,j) + wr*cos(phi)
      wynv(i,j)=wynv(i,j) + wr*sin(phi)
      wznv(i,j)=wznv(i,j) + wz
46    continue
      do 220 i=1,nivr
          if(i.eq.1) ksave=niva
          if(i.eq.2) ksave=niva
          if(i.eq.3) ksave=niva
      do 220 j=1,ksave
      n=2
      x=xiv(i,j)
      y=yiv(i,j)
      z=ziv(i,j)
      z3=abs(z1-z)
      phi=atan2(y,x)
      eta3=sqrt(x*x + y*y)
      goto 200
230   continue
      wxiv(i,j)=wxiv(i,j) + wr*cos(phi)
      wyiv(i,j)=wyiv(i,j) + wr*sin(phi)
      wziv(i,j)=wziv(i,j) + wz
220   continue
      if(iwrite.ne.2) goto 52
      do 51 j=1,nnva
      write(iwr,119) j
      write(iwr,120) (wxnv(i,j),i=1,nnvr)
      write(iwr,120) (wynv(i,j),i=1,nnvr)
      write(iwr,120) (wznv(i,j),i=1,nnvr)
51    continue
      do 53 j=1,niva
      write(iwr,119) j
      write(iwr,120) (wxiv(i,j),i=1,nivr)
      write(iwr,120) (wyiv(i,j),i=1,nivr)
      write(iwr,120) (wziv(i,j),i=1,nivr)
```

```
53    continue
52    continue
300   continue
      goto 49
200   continue
      k2=4.*r*eta3/((r+eta3)**2 + z3**2)
      if(k2.eq.1.) goto 47
      f=alog(4./sqrt(1.-k2))
      e=1.+.5*(f-.5)*(1.-k2) + .1875*(f-1.08333)*(1-k2)*(1-k2)
      capk=f+.25*(f-1.)*(1.-k2) + .14*(f-1.16666)*(1-k2)*(1-k2)
      goto 48
47    e=1.
      capk=10.
48    wr=-fpi*dgam*2.*sqrt(r/eta3/k2)*(capk*(2.-k2)-2.*e)
      psi=dpsin
      wz=0.
      do 250 k=1,2*kapa
      x3=r*r + eta3*eta3 + z3*z3 - 2.*r*eta3*cos(psi)
      wtemp=dpsii*r*(r-eta3*cos(psi))/(eta3*eta3+r*r-2.*r*eta3*cos(psi))
      wtemp=wtemp*(1.-(z3/sqrt(x3)))
      wz=wtemp + wz
      psi=psi+dpsii
250   continue
      wz=fpi*dgam*wz
      if(n.eq.1) goto 210
      if(n.eq.2) goto 230
49    continue
c
c     SELF-INDUCED VELOCITIES USING SCULLY APPROXIMATION
c
      if(iself.eq.0) goto 57
      if(iwrite.eq.2) write(iwr,150)
      do 54 i=1,nnvr
      do 54 j=1,nnva
      r=sqrt(xnv(i,j)**2+ynv(i,j)**2)
      dwz=fpi*gamt(i)*(alog(8.*r*tan(dpsin/4.)/fps2)-.25)/r
      wznv(i,j)=wznv(i,j)+dwz
54    continue
      do 55 i=1,nivr
          if(i.eq.1.or.i.eq.2) ksave=niva
          if(i.eq.3) ksave=niva
      do 55 j=1,ksave
      r=sqrt(xiv(i,j)**2+yiv(i,j)**2)
      dwz=fpi*gamti(i)*(alog(8.*r*tan(dpsii/4.)/fps2)-.25)/r
      wziv(i,j)=wziv(i,j)+dwz
55    continue
57    continue
c
c     CONVERGENCE TEST AND NEXT APPROXIMATION
c
      ktest=1
      if(niter.eq.1)goto 69
      savevel=facvel
      do 67 i=1,nnvr
      do 67 j=1,nnva
      wxnv(i,j)=facvel*wxnv(i,j) + (1.-facvel)*wxnvs(i,j)
      wynv(i,j)=facvel*wynv(i,j) + (1.-facvel)*wynvs(i,j)
      wznv(i,j)=facvel*wznv(i,j) + (1.-facvel)*wznvs(i,j)
67    continue
      do 68 i=1,nivr
```

```
      do 68 j=1,niva
      wxiv(i,j)=facvel*wxiv(i,j) + (1.-facvel)*wxivs(i,j)
      wyiv(i,j)=facvel*wyiv(i,j) + (1.-facvel)*wyivs(i,j)
      wziv(i,j)=facvel*wziv(i,j) + (1.-facvel)*wzivs(i,j)
68    continue
          wconv=0.
          do 1000 i=1,nnvr
          wconv=wconv + abs(wznv(i,2))
1000      continue
          wconv=wconv/float(nnvr)
      facvel=savevel
      ktest=0
      do 60 i=1,nnvr
      do 60 j=1,nnva
      if(abs(wznv(i,j)-wznvs(i,j)).gt.(.02*abs(wconv))) ktest=1
      if(ktest.eq.1) goto 66
60    continue
      do 65 i=1,nivr
      do 65 j=1,niva
      if(abs(wziv(i,j)-wzivs(i,j)).gt.(.02*abs(wconv))) ktest=1
      if(ktest.eq.1) goto 66
65    continue
66    continue
69    continue
      do 203 i=1,nnvr
      do 203 j=1,nnva
      wxnvs(i,j)=wxnv(i,j)
      wynvs(i,j)=wynv(i,j)
      wznvs(i,j)=wznv(i,j)
203   continue
      do 204 i=1,nivr
      do 204 j=1,niva
      wxivs(i,j)=wxiv(i,j)
      wyivs(i,j)=wyiv(i,j)
      wzivs(i,j)=wziv(i,j)
204   continue
      if(iwrite.lt.1)return
      write(iwr,125)ktest,facvel
      write(iwr,130)
      do 80 j=1,nnva
      write(iwr,119) j
      write(iwr,120) (wxnv(i,j),i=1,nnvr)
      write(iwr,120) (wynv(i,j),i=1,nnvr)
      write(iwr,120) (wznv(i,j),i=1,nnvr)
80    continue
      write(iwr,140)
      do 90 j=1,niva
      write(iwr,119) j
      write(iwr,120) (wxiv(i,j),i=1,nivr)
      write(iwr,120) (wyiv(i,j),i=1,nivr)
      write(iwr,120) (wziv(i,j),i=1,nivr)
90    continue
      return
100   format(/," LOOP2:  ITERATION ",I4,/," GAMC, GAMT, GAMTI, GAMTIP:")
123   format(2(9f10.6,/))
110   format(/," POSITIONS OF ROLLED-UP VORTICES:",3f10.6)
119   format(/,i4)
120   format(2(9f10.6,/))
121   format(/,i4,5x,f10.6)
125   format(/," LOOP2 CONVERGENCE ",i4," RELAXATION ",F5.2)
```

```
130   format(/," NEAR WAKE INDUCED VELOCITIES:")
140   format(/," INTERMEDIATE WAKE INDUCED VELOCITIES:")
150   format(/," SCULLY CONTRIRUTION FOR EACH TRAILER: ")
      end
```

```
      subroutine rollup(ncall)
c
c     This subroutine calculates:
c          a) the strengths of the trailing vorticies in the
c             near, intermediate and far wakes;
c          b) in addition, the spanwise positions of the rolled up
c             intermediate wake trailing vorticies, interpolating
c             an entire new intermediate wake (ncall=1) or just the
c             first azimuthal position (ncall=3).
c
      common/gamma/gamc(16),gamt(15),gamti(3),gamtip,etai,eta2,eta3,
     &j1,j2,k3,facgam
      common/parm/iwr,ird,lims,limi,lim2,niter,iwrite
      common/geom/nbldsi,nblds,sigma,fmu,etan(15),knnvr,etai(6),knivr,
     &itwist,thetad(15),theta(15),thetac(14),theta0,thet0d,alphas,
     &cd0,cdk,dpsind,dpsin,dpsiid,dpsii,coeff,coeffi,c,s,blades,
     &nnvr,nnva,nnvri,nnvai,nncr,nnca,etanv(15),etanc(16),
     &nivr,niva,nivri,nivai,nicr,nica,etaiv(6),etaic(7),
     &ntva,ntvai,ntca,fps1,fps2
      common/posit/xnv(15,18),ynv(15,18),znv(15,18),xiv(6,50),yiv(6,50),
     &ziv(6,50)
c
c     CLEAR THE ARRAYS OF TRAILING VORTEX STRENGTHS
c
      if(ncall.eq.3) goto 75
      do 10 i=1,nnvr
      gamt(i)=gamc(i+1)-gamc(i)
10    continue
      do 15 i=1,3
      gamti(i)=0.
15    continue
c
c     FIND WHERE GAMC IS MAXIMUM NEAR THE TIP (GAMT CHANGES SIGN)
c     AND ROLL UP FROM THERE OUT TO TIP
c
      gmax=0.
      do 20 i=nncr,2,-1
      if(abs(gamc(i)).gt.gmax) gmax=abs(gamc(i))
      if(gmax.eq.abs(gamc(i))) j1=i
20    continue
30    continue
      if(j1.gt.nnvr) write(iwr,200)
200   format(//," Rollup:  You screwed something up.")
      do 40 i=j1,nnvr
      gamti(3)=gamti(3)+gamt(i)
40    continue
c
c     FIND WHERE GAMT IS MINIMUM INBOARD AND ROLL UP ON EITHER SIDE
c     OF MINIMUM INTO TWO MORE TRAILING VORTICIES
c
c         j2=2
      j2=j1-3
c         if(nnvr.eq.15) j2=j1-4
      do 60 i=j2,j1-1
      gamti(2)=gamti(2)+gamt(i)
60    continue
      do 70 i=2,j2-1
      gamti(1)=gamti(1)+gamt(i)
70    continue
      gamtip=gamti(3)
```

```
      if(ncall.eq.1) return
c
c     FIND CENTROID OF EACH SECTION OF BLADE ROLLED UP
c
      eta1=0.
      eta2=0.
      eta3=0.
      do 80 i=1,j2-1
      eta1=eta1 + gamt(i)*etanv(i)
80    continue
      eta1=eta1/gamti(1)
      do 90 i=j2,j1-1
      eta2=eta2 + gamt(i)*etanv(i)
90    continue
      eta2=eta2/gamti(2)
      do 100 i=j1,nnvr
      eta3=eta3 + gamt(i)*etanv(i)
100   continue
      eta3=eta3/gamti(3)
      if(niter.gt.1) return
c
c     CALCULATE INTERPOLATION FACTORS EITHER FOR NEAR WAKE (NCALL=3)
c     OR FOR INTERMEDIATE WAKE (NITER=1) AND INTERPOLATE NEW POSITIONS
c
      do 110 i=1,nivr
      if(eta1.ge.etaiv(i)) i1=i
      if(eta2.ge.etaiv(i)) i2=i
      if(eta3.gt.etaiv(i)) i3=i
110   continue
      terp1=(eta1-etaiv(i1))/(etaiv(i1+1)-etaiv(i1))
      terp2=(eta2-etaiv(i2))/(etaiv(i2+1)-etaiv(i2))
      terp3=(eta3-etaiv(i3))/(etaiv(i3+1)-etaiv(i3))
      do 140 j=1,niva
      xiv1=xiv(i1,j) + terp1*(xiv(i1+1,j)-xiv(i1,j))
      xiv2=xiv(i2,j) + terp2*(xiv(i2+1,j)-xiv(i2,j))
      xiv3=xiv(i3,j) + terp3*(xiv(i3+1,j)-xiv(i3,j))
      yiv1=yiv(i1,j) + terp1*(yiv(i1+1,j)-yiv(i1,j))
      yiv2=yiv(i2,j) + terp2*(yiv(i2+1,j)-yiv(i2,j))
      yiv3=yiv(i3,j) + terp3*(yiv(i3+1,j)-yiv(i3,j))
      ziv1=ziv(i1,j) + terp1*(ziv(i1+1,j)-ziv(i1,j))
      ziv2=ziv(i2,j) + terp2*(ziv(i2+1,j)-ziv(i2,j))
      ziv3=ziv(i3,j) + terp3*(ziv(i3+1,j)-ziv(i3,j))
      xiv(1,j)=xiv1
      xiv(2,j)=xiv2
      xiv(3,j)=xiv3
      yiv(1,j)=yiv1
      yiv(2,j)=yiv2
      yiv(3,j)=yiv3
      ziv(1,j)=ziv1
      ziv(2,j)=ziv2
      ziv(3,j)=ziv3
140   continue
      nivr=3
      nicr=nivr+1
      nivr1=nivr-1
      return
75    continue
         x=0.
         y=0.
         z=0.
```

```
      do 155 i=2,j2-1
         x=xnv(i,nnva)*gamt(i)+x
         y=ynv(i,nnva)*gamt(i)+y
         z=znv(i,nnva)*gamt(i)+z
155      continue
         xiv(1,1)=x/gamti(1)
         yiv(1,1)=y/gamti(1)
         ziv(1,1)=z/gamti(1)
         x=0.
         y=0.
         z=0.
         do 156 i=j2,j1-1
         x=xnv(i,nnva)*gamt(i)+x
         y=ynv(i,nnva)*gamt(i)+y
         z=znv(i,nnva)*gamt(i)+z
156      continue
         xiv(2,1)=x/gamti(2)
         yiv(2,1)=y/gamti(2)
         ziv(2,1)=z/gamti(2)
         x=0.
         y=0.
         z=0.
         do 157 i=j1,nnvr
         x=xnv(i,nnva)*gamt(i)+x
         y=ynv(i,nnva)*gamt(i)+y
         z=znv(i,nnva)*gamt(i)+z
157      continue
         xiv(3,1)=x/gamtip
         yiv(3,1)=y/gamtip
         ziv(3,1)=z/gamtip
      return
      end
```

```
      subroutine vind
c
c     This routine calculates the induced velocities on the
c     blade (VINDB), the near wake (VINDN) and the intermediate
c     wake (VINDI), due to a segment element of endpoint coordinates
c     x1,y1,z1 and x2,y2,z2 and of strength gm.  VINDB only calculates
c     the influence coefficients cw(i,l,n), which give the velocity
c     induced at spanwise position i by a trailing vortex of strength gm=1
c     located at spanwise position l (summed over all azimuthal positions)
c     in wake n (near=1, intermediate=2, far=3).  These are used in LOOP1.
c
c     VELOCITIES INDUCED ON BLADE
c
      common/parm/iwr,ird,lims,lim1,lim2,niter,iwrite
      common/geom/nbldsi,nblds,sigma,fmu,etan(15),knnvr,etai(6),knivr,
     &itwist,thetad(15),theta(15),thetac(14),theta0,thet0d,alphas,
     &cd0,cdk,dpsind,dpsin,dpsiid,dpsii,coeff,coeff1,c,s,blades,
     &nnvr,nnva,nnvr1,nnva1,nncr,nnca,etanv(15),etanc(16),
     &nivr,niva,nivr1,niva1,nicr,nica,etaiv(6),etaic(7),
     &ntva,ntva1,ntca,fps1,fps2
      common/vindo/ifar,nfar,x1,x2,y1,y2,z1,z2,x,y,z,facvel,gm,ux,uy,uz
      common/posit/xnv(15,18),ynv(15,18),znv(15,18),xiv(6,50),yiv(6,50),
     &ziv(6,50)
      common/veloc/wxnv(15,18),wynv(15,18),wznv(15,18),wxiv(6,50),
     &wyiv(6,50),wziv(6,50),facgeom
      common/coef/cwx(15,15,3),cwy(15,15,3),cwz(15,15,3)
      common/gamma/gamc(16),gamt(15),gamti(3),gamtip,eta(3),
     &ji,kk(2),facgam
      entry vindb(n1,n2,l,n)
      if(gm.ne.1) write(iwr,300)
300   format(/," VINDB:  What are you doing?  gm is not equal to one.")
      iskip=0
      do 10 i=1,nnvr-1
      x=(etanv(i)+etanv(i+1))/2.
      y=0.
      z=0.
      cc=1.
      ss=0.
c      n3=1
c      if(n1.eq.n3.or.n2.eq.n3) iskip=1
      do 10 j=1,nblds
      if(iskip.eq.1) goto 11
      call wxyz(fps2)
      cwx(i,l,n)=cwx(i,l,n) + (ux*cc + uy*ss)
      cwy(i,l,n)=cwy(i,l,n) + (uy*cc - ux*ss)
      cwz(i,l,n)=cwz(i,l,n) + uz
11    continue
      iskip=0
      sav=x
      x=x*c - y*s
      y=y*c + sav*s
      sav=cc
      cc=cc*c - ss*s
      ss=ss*c + sav*s
10    continue
      return
c
c     VELOCITIES INDUCED ON NEAR WAKE
c
      entry vindn(n1,n2,loop)
```

```
      iskip=0
      nf=nnva
      if(n1.eq.0) nf=nnva-1
      if(loop.eq.1) nf=1
      do 20 i=1,nnvr
      do 20 j=1,nf
      x=xnv(i,j)
      y=ynv(i,j)
      z=znv(i,j)
      cc=1.
      ss=0.
      n3=(j-1)*nnvr + 1
      if(n1.eq.n3.or.n2.eq.n3) iskip=1
      do 20 k=1,nblds
      if(iskip.eq.1) goto 21
      call wxyz(fps2)
      wxnv(i,j)=wxnv(i,j) + (ux*cc + uy*ss)
      wynv(i,j)=wynv(i,j) + (uy*cc - ux*ss)
      wznv(i,j)=wznv(i,j) + uz
21    continue
      iskip=0
      sav=x
      x=x*c - y*s
      y=y*c + sav*s
      sav=cc
      cc=cc*c - ss*s
      ss=ss*c + sav*s
20    continue
      if(loop.eq.1) return
      if(n1.ne.0) return
      iskip=0
      do 26 i=1,nnvr
      x=xnv(i,nnva)
      y=ynv(i,nnva)
      z=znv(i,nnva)
      cc=1.
      ss=0.
      r=sqrt(x*x + y*y)
      r1=sqrt(x1*x1 + y1*y1)
      if(abs(r-r1).lt..025) iskip=1
      do 26 k=1,nblds
      if(iskip.eq.1) goto 28
      call wxyz(fps2)
      wxnv(i,nnva)=wxnv(i,nnva) + (ux*cc + uy*ss)
      wynv(i,nnva)=wynv(i,nnva) + (uy*cc - ux*ss)
      wznv(i,nnva)=wznv(i,nnva) + uz
28    continue
      iskip=0
      sav=x
      x=x*c - y*s
      y=y*c + sav*s
      sav=cc
      cc=cc*c - ss*s
      ss=ss*c + sav*s
26    continue
      return
c
c     VELOCITIES INDUCED ON INTERMEDIATE WAKE
c
      entry vindi(n1,n2)
```

```
      iskip=0
      nf=1
      if(n2.eq.0) nf=2
      do 30 i=1,nivr
          if(i.eq.1) ksave=niva
          if(i.eq.2) ksave=niva
          if(i.eq.3) ksave=niva
      do 30 j=nf,ksave
      x=xiv(i,j)
      y=yiv(i,j)
      z=ziv(i,j)
      cc=1.
      ss=0.
      n3=(j-1)*nivr + nnva*nnvr + i
      if(n1.eq.n3.or.n2.eq.n3) iskip=1
      do 30 k=1,nblds
      if(iskip.eq.1) goto 31
      call wxyz(fps2)
      wxiv(i,j)=wxiv(i,j) + (ux*cc + uy*ss)
      wyiv(i,j)=wyiv(i,j) + (uy*cc - ux*ss)
      wziv(i,j)=wziv(i,j) + uz
31    continue
      iskip=0
      sav=x
      x=x*c - y*s
      y=y*c + sav*s
      sav=cc
      cc=cc*c - ss*s
      ss=ss*c + sav*s
30    continue
      if(n2.ne.0) return
      iskip=0
      do 35 i=1,nivr
      x=xiv(i,1)
      y=yiv(i,1)
      z=ziv(i,1`
      cc=1.
      ss=0.
      r=sqrt(x*x + y*y)
      r2=sqrt(x2*x2 + y2*y2)
      if(abs(r-r2).lt..025) iskip=1
      do 35 k=1,nblds
      if(iskip.eq.1) goto 36
      call wxyz(fps2)
      wxiv(i,1)=wxiv(i,1) + ux*cc + uy*ss
      wyiv(i,1)=wyiv(i,1) + uy*cc - ux*cc
      wziv(i,1)=wziv(i,1) + uz
36    continue
      iskip=0
      sav=x
      x=x*c - y*s
      y=y*c + sav*s
      sav=cc
      cc=cc*c - ss*s
      ss=ss*c + sav*s
35    continue
      return
      end
```

ORIGINAL PAGE IS
OF POOR QUALITY


```
      subroutine wxyz(fps2)
c
c     This routine calculates the induced velocity due to any segment
c     of endpoints x1,y1,z1 and x2,y2,z2 and of strength gm on any
c     point x,y,z.  It is only called by VIND.
c
      real*4 i0
      common/vindo/ifar,nfar,x1,x2,y1,y2,z1,z2,x,y,z,facvel,gm,ux,uy,uz
      data fpi/.079577/
      xct=.5*(x1+x2)
      yct=.5*(y1+y2)
      zct=.5*(z1+z2)
      dxx=x-xct
      dyy=y-yct
      dzz=z-zct
          r12=sqrt((x1-x)**2+(y1-y)**2+(z1-z)**2)
          r22=sqrt((x2-x)**2+(y2-y)**2+(z2-z)**2)
          if(r12.le..1e-5.or.r22.le..1e-5) goto 100
      dsx=.5*(x2-x1)
      dsy=.5*(y2-y1)
      dsz=.5*(z2-z1)
      ds2=dsx**2 +dsy**2 +dsz**2
      fva=fps2*fps2*ds2
      rmax2=400.*ds2
      r02=dxx**2 + dyy**2 + dzz**2
      r03=r02*sqrt(r02)
      dsmx=dsz*dyy - dsy*dzz
      dsmy=dsx*dzz - dsz*dxx
      dsmz=dsy*dxx - dsx*dyy
      dsm2=dsmx**2 + dsmy**2 + dsmz**2
      if(dsm2.le..1e-15) goto 100
      fvds=fva/dsm2
      i0=1.
      if(r02.gt.rmax2) goto 120
      a=-(dxx*dsx + dyy*dsy + dzz*dsz)/r02
      alpha2=ds2/r02
      alpaa=alpha2 - a*a
      if(abs(alpaa).le..1e-15) goto 110
      sq1a=sqrt(abs(1. + 2.*a + alpha2))
      sq2a=sqrt(abs(1. - 2.*a + alpha2))
      if(sq1a.lt..001) sq1a=.001
      if(sq1a.lt..001) sq1a=.001
      i0=(alpha2 + a)/sq1a + (alpha2-a)/sq2a
      i0=i0/(2.*alpaa)
      goto 120
110   continue
      i0=1./((1.-a*a)**2)
120   fact=-i0*fpi*gm*2./r03
      fact=fact/(1.+fvds)
      ux=fact*dsmx
      uy=fact*dsmy
      uz=fact*dsmz
      return
100   continue
      ux=0.
      uy=0.
      uz=0.
      return
      end
```

```
      subroutine intgr(ktest)
c
c     This routine integrates the wake geometry using the induced
c     velocities returned by LOOP2.  Tangential and radial velocities
c     are not used; the wake is integrated directly from wxnv, wynv
c     and wznv.   The velocity of a wake node in travelling from
c     position i,j to i,j+1 is assumed to be the average of the
c     velocities at i,j and i,j+1.
c
      common/parm/iwr,ird,lims,lim1,lim2,niter,iwrite
      common/geom/nbldsi,nblds,sigma,fmu,etan(15),knnvr,etai(6),knivr,
     &itwist,thetad(15),theta(15),thetac(14),theta0,thet0d,alphas,
     &cd0,cdk,dpsind,dpsin,dpsiid,dpsii,coeff,coeffi,c,s,blades,
     &nnvr,nnva,nnvri,nnvai,nncr,nnca,etanv(15),etanc(16),
     &nivr,niva,nivri,nivai,nicr,nica,etaiv(6),etaic(7),
     &ntva,ntvai,ntca,fpsi,fps2
      common/posit/xnv(15,18),ynv(15,18),znv(15,18),xiv(6,50),
     &yiv(6,50),ziv(6,50)
      common/veloc/wxnv(15,18),wynv(15,18),wznv(15,18),wxiv(6,50),
     &wyiv(6,50),wziv(6,50),facgeom
      common/gamma/gamc(16),gamt(15),gamti(3),gamtip,etz(3),k1,k2,k3,facgam
      common/intgrsave/xnvs(15,18),ynvs(15,18),znvs(15,18),xivs(6,50),
     &yivs(6,50),zivs(6,50)
          common/special/itgr
          ktest=1
          if(niter.eq.1) goto 203
      do 10 i=1,nnvr
      do 10 j=1,nnva
      xnv(i,j)=0.
      ynv(i,j)=0.
      znv(i,j)=0.
10    continue
      do 20 i=1,nnvr
      xnv(i,1)=etanv(i)
      ynv(i,1)=0.
      znv(i,1)=0.
      phi=0.
      do 20 j=1,nnva-1
      phi1=phi+dpsin
      wr=(wxnv(i,j)*cos(phi)+wxnv(i,j+1)*cos(phi1)+wynv(i,j)*sin(phi)+
     &wynv(i,j+1)*sin(phi1))/2.
      wt=(-wxnv(i,j)*sin(phi)-wxnv(i,j+1)*sin(phi1)+wynv(i,j)*cos(phi)+
     &wynv(i,j+1)*cos(phi1))/2.
      r=sqrt(xnv(i,j)**2 + ynv(i,j)**2)
      r1=r+dpsin*wr
      phi1=phi+dpsin*(2.*wt/(r+r1))+dpsin
      xnv(i,j+1)=r1*cos(phi1)
      ynv(i,j+1)=r1*sin(phi1)
      znv(i,j+1)=znv(i,j) + dpsin*(wznv(i,j)+wznv(i,j+1))/2.
      phi=phi1
20    continue
      do 30 i=1,nivr
      do 30 j=1,niva
      xiv(i,j)=0.
      yiv(i,j)=0.
      ziv(i,j)=0.
30    continue
      ncall=3
      call rollup(ncall)
          if(itgr.ne.1.or.nnva.gt.2) goto 1033
```

```
          do 1030 i=k1,nnvr
          xnv(i,nnva)=xiv(nivr,1)
          ynv(i,nnva)=yiv(nivr,1)
          znv(i,nnva)=ziv(nivr,1)
1030      continue
1033      continue
c         ziv(nivr,1)=.025
      do 40 i=1,nivr
      phi=atan2(yiv(i,1),xiv(i,1))
      do 40 j=1,niva-1
      phi1=phi+dpsi1
      wr=(wxiv(i,j)*cos(phi)+wxiv(i,j+1)*cos(phi1)+wyiv(i,j)*sin(phi)+
     &wyiv(i,j+1)*sin(phi1))/2.
      wt=(-wxiv(i,j)*sin(phi)-wxiv(i,j+1)*sin(phi1)+wyiv(i,j)*cos(phi)+
     &wyiv(i,j)*cos(phi1))/2.
      r=sqrt(xiv(i,j)**2 + yiv(i,j)**2)
      r1=r+dpsi1*wr
      phi1=phi+dpsi1+dpsi1*(2.*wt/(r+r1))
      xiv(i,j+1)=r1*cos(phi1)
      yiv(i,j+1)=r1*sin(phi1)
      ziv(i,j+1)=ziv(i,j) + dpsi1*(wziv(i,j)+wziv(i,j+1))/2.
      phi=phi1
40    continue
      savegeom=facgeom
      do 41 i=1,nnvr
      do 41 j=1,nnva
      xnv(i,j)=xnv(i,j)*facgeom + xnvs(i,j)*(1.-facgeom)
      ynv(i,j)=ynv(i,j)*facgeom + ynvs(i,j)*(1.-facgeom)
      znv(i,j)=znv(i,j)*facgeom + znvs(i,j)*(1.-facgeom)
41    continue
          facgeom=savegeom
      do 42 i=1,nivr
      do 42 j=1,niva
      xiv(i,j)=xiv(i,j)*facgeom + xivs(i,j)*(1.-facgeom)
      yiv(i,j)=yiv(i,j)*facgeom + yivs(i,j)*(1.-facgeom)
      ziv(i,j)=ziv(i,j)*facgeom + zivs(i,j)*(1.-facgeom)
42    continue
      facgeom=savegeom
          ktest=0
      do 43 j=1,nnva
          zconv=0.
          do 1000 ii=1,nnvr-1
          zconv=zconv+abs(znv(ii,j))
1000      continue
          zconv=zconv/nnvr
      do 43 i=1,nnvr-1
      if(abs(znv(i,j)-znvs(i,j)).gt.(.02*zconv)) ktest=1
      if(ktest.eq.1) goto 53
43    continue
      do 44 i=1,nivr
      do 44 j=1,niva
      if(abs(ziv(i,j)-zivs(i,j)).gt..02*abs(ziv(i,j))) ktest=1
      if(ktest.eq.1) goto 53
44    continue
53    continue
203   continue
      do 202 i=1,nnvr
      do 202 j=1,nnva
      xnvs(i,j)=xnv(i,j)
      ynvs(i,j)=ynv(i,j)
```

```
      znvs(i,j)=znv(i,j)
202   continue
      do 204 i=1,nivr
          if(i.eq.1.or.i.eq.2) ksave=niva
          if(i.eq.3) ksave=niva
      do 204 j=1,ksave
      xivs(i,j)=xiv(i,j)
      yivs(i,j)=yiv(i,j)
      zivs(i,j)=ziv(i,j)
204   continue
      if(iwrite.lt.1) goto 50
      write(iwr,100) ktest,facgeom
      do 45 j=1,nnva
      write(iwr,109) j
      write(iwr,110) (xnv(i,j),i=1,nnvr)
      write(iwr,110) (ynv(i,j),i=1,nnvr)
      write(iwr,110) (znv(i,j),i=1,nnvr)
45    continue
      do 46 j=1,niva
      write(iwr,109) j
      write(iwr,110) (xiv(i,j),i=1,nivr)
      write(iwr,110) (yiv(i,j),i=1,nivr)
      write(iwr,110) (ziv(i,j),i=1,nivr)
46    continue
50    return
100   format(//," INTGR CONVERGENCE ",i4," RELAXATION ",F10.5)
109   format(/,i4)
110   format(2(9f10.6,/))
      end
```

```
      subroutine loopi(ktest)
c
c     This routine calculates a new distribution of bound circulation
c     gamc(i) consistent with the current wake geometry as given by
c     cw(i,l,n), by iteration.
c
c     INFLUENCE COEFFICIENTS CW(i,1,1)
c
      common/veloc/wxnv(15,18),wynv(15,18),wznv(15,18),wxiv(6,50),
     &wyiv(6,50),wziv(6,50),facgeom
      common/parm/iwr,ird,lims,lim1,lim2,niter,iwrite
      common/geom/nblds1,nblds,sigma,fmu,etan(15),knnvr,etai(6),knivr,
     &itwist,thetad(15),theta(15),thetac(14),theta0,thet0d,alphas,
     &cd0,cdk,dpsind,dpsin,dpsiid,dpsii,coeff,coeff1,c,s,blades,
     &nnvr,nnva,nnvr1,nnva1,nncr,nnca,etanv(15),etanc(16),
     &nivr,niva,nivr1,niva1,nicr,nica,etaiv(6),etaic(7),
     &ntva,ntva1,ntca,fps1,fps2
      common/gamma/gamc(16),gamt(15),gamti(3),gamtip,eta1,eta2,eta3,
     &j1,kk(2),facgam
      common/vindo/ifar,nfar,x1,x2,y1,y2,z1,z2,x,y,z,facvel,gm,ux,uy,uz
      common/posit/xnv(15,18),ynv(15,18),znv(15,18),xiv(6,50),
     &yiv(6,50),ziv(6,50)
      common/coef/cwx(15,15,3),cwy(15,15,3),cwz(15,15,3)
      common/self/iself
          common/extraspac/fps3
      dimension gamcs(16)
      common/loopsave/wxv(16),wyv(16),wzv(16)
      data pi,twopi,fpi/3.14159,6.28318,.079577/
      real k2
      do 5 i=1,nnvr
      do 5 j=1,nnvr
      do 5 n=1,3
      cwx(i,j,n)=0.
      cwy(i,j,n)=0.
      cwz(i,j,n)=0.
5     continue
      dps2=fps2
      do 10 i=1,nnvr
          fps2=fps1
          if(i.eq.nnvr-1) fps2=dps2
      do 11 j=1,nnva-1
      x1=xnv(i,j)
      y1=ynv(i,j)
      z1=znv(i,j)
      x2=xnv(i,j+1)
      y2=ynv(i,j+1)
      z2=znv(i,j+1)
      n1=(j-1)*nnvr + i
      n2=n1+nnvr
      gm=1.
      call vindb(n1,n2,i,1)
11    continue
10    continue
      fps2=dps2
      pnib=twopi/blades
      do 20 i=1,nivr
          if(i.eq.1) ksave=niva
          if(i.eq.2) ksave=niva
          if(i.eq.3) ksave=niva
      do 21 j=1,ksave-1
```

```
      x1=xiv(i,j)
      y1=yiv(i,j)
      z1=ziv(i,j)
      x2=xiv(i,j+1)
      y2=yiv(i,j+1)
      z2=ziv(i,j+1)
      n1=(j-1)*nivr + nnvr*nnva + i
      n2=n1+nivr
      dps2=fps2
      phi1=float(nnva-1)*dpsin + float(j-1)*dpsi1
      if(phi1.gt.phib-dpsin) fps2=fps3
      gm=1.
      call vindb(n1,n2,1,2)
      fps2=dps2
21    continue
20    continue
      n=3
      if(ifar.eq.1) goto 401
      dz=ziv(nivr,niva)-ziv(nivr,niva-1)
      x1=xiv(nivr,niva)
      y1=yiv(nivr,niva)
      z1=ziv(nivr,niva)
      psi=atan2(y1,x1)
      r=sqrt(x1*x1 + y1*y1)
      do 30 j=1,nfar
      psi=psi+dpsi1
      x2=r*cos(psi)
      y2=r*sin(psi)
      z2=z1+dz
      n1=j + nnva*nnvr + niva*nivr
      n2=n1+1
      gm=1.
      call vindb(n1,n2,1,3)
      x1=x2
      y1=y2
      z1=z2
30    continue
      goto 301
401   continue
      fkapa=phib/dpsi1
      kapa=ifix(fkapa + .5)
      do 505 ki=1,nivr
      x1=xiv(ki,niva)
      y1=yiv(ki,niva)
      z1=ziv(ki,niva)+(ziv(ki,niva)-ziv(ki,niva-kapa))
      dgam=1./(ziv(ki,niva)-ziv(ki,niva-kapa))
      r=(sqrt(x1*x1+y1*y1))
      do 201 i=1,nnvr-1
      x=(etanv(i)+etanv(i+1))/2.
      y=0.
      z=0.
      z3=abs(z1-z)
      eta3=sqrt(x*x+y*y)
      k2=4.*r*eta3/((r+eta3)**2+z3**2)
      if(k2.eq.1.) goto 202
      f=alog(4./sqrt(1.-k2))
      e=1.+.5*(f-.5)*(1.-k2) + .1875*(f-1.08333)*(1.-k2)*(1.-k2)
      capk=f+.25*(f-1.)*(1.-k2)+.14*(f-1.13666)*(1.-k2)*(1.-k2)
      goto 203
202   e=1.
```

```
      capk=10.
203   continue
      wr=-fpi*dgam*2.*sqrt(r/eta3/k2)*(capk*(2.-k2)-2.*e)
      psi=dpsin
      wz=0.
      do 204 k=1,2*kapa
      x3=r*r + eta3*eta3 + z3*z3 - 2.*r*eta3*cos(psi)
      wtemp=dpsii*r*(r-eta3*cos(psi))/(eta3*eta3+r*r-2.*r*eta3*cos(psi))
      wtemp=wtemp*(1.-(z3/sqrt(x3)))
      wz=wz+wtemp
      psi=psi+dpsii
204   continue
      wz=wz*fpi*dgam
      cwx(i,ki,n)=cwx(i,ki,n)+wr
      cwz(i,ki,n)=cwz(i,ki,n)+wz
201   continue
505   continue
c
c     Loop on gamc(i).  First clear array of blade induced velocities.
c     Then calculate new wxv,wyv,wzv from influence coefficients.
c     Then new gamc.  Back to beginning until converged.
c
301   continue
      if(iwrite.eq.2) write(iwr,304)
         ktemp=0
      do 200 k=1,limi
      savegam=facgam
1000     continue
      do 35 i=1,nnvr
      wxv(i)=0.
      wyv(i)=0.
      wzv(i)=0.
35    continue
      do 36 j=1,nnvr
      do 37 i=1,nnvr-1
      wxv(i)=wxv(i) + cwx(i,j,1)*gamt(j)
      wyv(i)=wyv(i) + cwy(i,j,1)*gamt(j)
      wzv(i)=wzv(i) + cwz(i,j,1)*gamt(j)
37    continue
c        if(ktemp.eq.1) write(iwr,310) (wzv(k),k=1,nnvr)
36    continue
      do 40 j=1,nivr
      do 41 i=1,nnvr-1
      wxv(i)=wxv(i) + cwx(i,j,2)*gamti(j)
      wyv(i)=wyv(i) + cwy(i,j,2)*gamti(j)
      wzv(i)=wzv(i) + cwz(i,j,2)*gamti(j)
41    continue
c        if(ktemp.eq.1) write(iwr,310) (wzv(k),k=1,nnvr)
40    continue
      do 50 j=1,nivr
      do 51 i=1,nnvr-1
      wxv(i)=wxv(i) + cwx(i,j,3)*gamti(j)
      wyv(i)=wyv(i) + cwy(i,j,3)*gamti(j)
      wzv(i)=wzv(i) + cwz(i,j,3)*gamti(j)
51    continue
c        if(ktemp.eq.1) write(iwr,310) (wzv(k),k=1,nnvr)
50    continue
c        if(ktemp.eq.1) goto 1001
c      if(iself.eq.0) goto 55
c       do 56 i=1,nnvr
```

```
c       r=etanv(i)
c       dwz=fpi*gamt(i)*(alog(8.*r*tan(dpsin/4.)/fps2)-.25)/r
c       wzv(i)=wzv(i)+dwz
c 56     continue
55    continue
      gamc(1)=0.
      gamc(nncr)=0.
      do 70 i=1,nnvr-1
      wz=(wzv(i)+fmu)
      wy=wyv(i) + (etanv(i)+etanv(i+1))/2.
      tlam=wz/wy
      flamda=atan(tlam)
      u=sqrt(wz**2 + wy**2)
      alp=flamda - thetac(i)
      if(alp.gt.alphas) alp=alphas
      gamc(i+1)=pi*pi*sigma*u*alp/blades
70    continue
      gamc(nnvr)=gamc(nnvr)*coeff1
c
c     CONVERGENCE TEST
c
      ktest=1
      if(k.eq.1) goto 94
      ktest=0
      do 80 i=1,nncr
      if(abs(gamc(i)-gamcs(i)).gt:.01*abs(gamc(i))) ktest=1
      if(ktest.eq.1) goto 81
80    continue
81    continue
      do 90 i=1,nncr
      gamc(i)=gamc(i)*facgam + gamcs(i)*(1.-facgam)
90    continue
94    continue
c          if(ktest.eq.0) ktemp=1
c          if(ktest.eq.0) goto 1000
1001       continue
      if(iwrite.ne.2.and.ktest.ne.0) goto 52
      if(iwrite.lt.1) goto 52
      write(iwr,311) k
      write(iwr,310) (gamc(i),i=1,nnvr)
      write(iwr,310) (wxv(i),i=1,nnvr)
      write(iwr,310) (wyv(i),i=1,nnvr)
      write(iwr,310) (wzv(i),i=1,nnvr)
52    continue
      do 91 i=1,nncr
      gamcs(i)=gamc(i)
91    continue
      call rollup(1)
      if(ktest.eq.0) goto 205
200   continue
205   continue
      facgam=savegam
      if(ktest.eq.1)write(iwr,300)
      if(ktest.eq.0.and.iwrite.gt.0) write(iwr,305) k
311   format(/,i4)
310   format(2(9f10.6,/))
304   format(//,"  LOOP1 RESULTS",/)
300   format(//,"  LOOP1:  NO CONVERGENCE")
305   format(//,"  LOOP1:  CONVERGED IN ",I4," ITERATIONS")
400   return
```

SAMPLE OUTPUT

```
*** INPUT *** CASE # :  1
NUMBER OF BLADES 2
SIGMA     0.04640
MU     0.00000
PITCH ANGLE DISTRIBUTION(DG):  17.50000  15.30000  14.20000  12.55000  10.90000
       9.25000   8.70000   8.15000   7.93000   7.71000   7.49000
       7.27000   7.05000
STALL ANGLE     0.20000
 CD=.0140+0.500*ALPHA**2
MAX. NUMBER OF ITERATIONS FOR LOOP1 AND LOOP2:  20 200
CORE SIZE FOR NEAR WAKE  INBOARD:  0.050  TIP:  0.010
NEAR WAKE DEFINITION:(13, 8),   0.10000   0.25000   0.35000   0.50000
       0.65000   0.80000   0.85000   0.90000   0.92000   0.94000
  0.96000   0.98000   1.00000
INT. WAKE DEFINITION:( 6,38),   0.10000   0.25000   0.60000   0.85000
       0.95000   1.00000
CORE BURST TO  0.05
TIP COEFFICIENT  0.00
```

MAIN RESULTS -- CONVERGENCE: 0

CT= -0.004533 CP= 0.000337

GAMC,WXC,WYC,WZC:

-0.00628	-0.01043	-0.01269	-0.01391	-0.01533	-0.01725	-0.01989	-0.02106	-0.02119	-0.01932	-0.01443	0.00000	GAMC
0.00125	0.00365	0.00491	-0.01000	-0.02830	-0.05013	-0.05587	-0.04742	-0.04134	-0.03603	-0.03172	-0.02829	WXC
-0.00037	-0.00123	-0.00233	-0.00287	-0.00276	-0.00262	-0.00225	-0.00140	-0.00080	-0.00034	-0.00005	0.00085	WYC
0.02283	0.03163	0.04364	0.05684	0.06046	0.05373	0.04163	0.03568	0.03439	0.04172	0.06219	-0.05524	WZC

WAKE GEOMETRY:

1													
0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	PSI
0.10000	0.25000	0.35000	0.50000	0.65000	0.80000	0.85000	0.90000	0.92000	0.94000	0.96000	0.98000	1.00000	R
0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	0.00000	Z
2													
9.803	9.889	9.891	9.913	9.941	9.958	9.960	9.977	9.982	9.991	10.007	10.006	10.003	
0.09999	0.25054	0.35094	0.49951	0.64669	0.79266	0.84057	0.89125	0.91248	0.93420	0.95956	0.97439	0.99061	
0.00398	0.00661	0.00868	0.01123	0.01212	0.01194	0.01158	0.00948	0.00984	0.01046	0.01271	0.00098	-0.00981	
3													
19.464	19.739	19.755	19.812	19.872	19.911	19.911	19.949	19.961	19.983	20.035	20.011	19.997	
0.09991	0.25101	0.35182	0.49899	0.64341	0.78559	0.83145	0.88260	0.90572	0.93072	0.96796	0.96817	0.97260	
0.00890	0.01438	0.01865	0.02361	0.02537	0.02513	0.02550	0.02170	0.02260	0.02373	0.02143	0.00302	-0.01650	
4													
29.273	29.653	29.663	29.731	29.814	29.871	29.867	29.927	29.948	29.988	30.076	30.017	29.969	
0.09982	0.25149	0.35270	0.49851	0.64019	0.77881	0.82272	0.87459	0.89975	0.92961	0.97575	0.96176	0.95108	
0.01270	0.02043	0.02699	0.03476	0.03765	0.03761	0.03912	0.03332	0.03429	0.03497	0.01631	0.00562	-0.01114	
5													
39.186	39.602	39.594	39.661	39.762	39.836	39.824	39.908	39.942	40.003	40.101	40.023	39.943	
0.09978	0.25200	0.35358	0.49807	0.63705	0.77238	0.81441	0.86691	0.89450	0.93051	0.97308	0.95581	0.93606	
0.01594	0.02575	0.03470	0.04549	0.04967	0.04991	0.05295	0.04472	0.04520	0.04422	0.00390	0.00881	0.00558	
6													
49.141	49.570	49.541	49.597	49.714	49.805	49.782	49.891	49.939	50.024	50.097	50.030	49.939	
0.09979	0.25255	0.35448	0.49767	0.63400	0.76630	0.80654	0.85943	0.88981	0.93306	0.95841	0.95051	0.93368	
0.01886	0.03062	0.04200	0.05606	0.06161	0.06215	0.06704	0.05605	0.05539	0.05127	-0.00503	0.01202	0.02442	
7													
59.121	59.554	59.502	59.536	59.668	59.777	59.740	59.874	59.940	60.052	60.070	60.039	59.963	
0.09985	0.25315	0.35539	0.49729	0.63104	0.76060	0.79910	0.85216	0.88557	0.93681	0.93792	0.94548	0.94085	
0.02154	0.03511	0.04879	0.06685	0.07359	0.07442	0.08147	0.06735	0.06491	0.05599	-0.00296	0.01480	0.03558	
8													
69.115	69.551	69.481	69.468	69.623	69.754	69.702	69.861	69.952	70.032	70.059	70.046	69.973	
0.09994	0.25380	0.35636	0.49682	0.62819	0.75526	0.79222	0.84529	0.88202	0.93856	0.92211	0.94027	0.94701	
0.02403	0.03921	0.05477	0.07787	0.08572	0.08650	0.09614	0.07904	0.07392	0.05847	0.00620	0.01730	0.03807	
1													
69.602	69.760	70.050											
0.43726	0.81049	0.93590											
0.06098	0.09036	0.01840											

2	89.520	89.694	90.056
	0.43903	0.79760	0.92485
	0.07532	0.11895	0.02570
3	109.444	109.660	110.072
	0.44082	0.78718	0.91416
	0.08936	0.14845	0.03339
4	129.350	129.658	130.114
	0.44268	0.77875	0.90374
	0.10304	0.17848	0.04153
5	149.209	149.663	150.187
	0.44459	0.77222	0.89352
	0.11636	0.20866	0.04987
6	168.963	169.621	170.281
	0.44639	0.76754	0.88370
	0.12953	0.23887	0.05741
7	188.393	189.413	190.262
	0.44710	0.76425	0.87492
	0.14491	0.27038	0.07239
8	207.821	209.187	210.184
	0.44672	0.76190	0.86721
	0.16268	0.30292	0.09449
9	227.568	229.043	230.179
	0.44618	0.76019	0.86001
	0.18045	0.33470	0.11490
10	247.418	248.940	250.166
	0.44528	0.75872	0.85346
	0.19785	0.36537	0.13454
11	267.288	268.870	270.157
	0.44368	0.75773	0.84785
	0.21500	0.39489	0.15368
12	287.189	288.825	290.155
	0.44184	0.75765	0.84305
	0.23179	0.42339	0.17245
13	307.104	308.756	310.158
	0.44023	0.75803	0.83872
	0.24813	0.45126	0.19083

PSI=180: Z(TIP)=0.061
R(TIP)=0.879

ORIGINAL PAGE IS
OF POOR QUALITY

14		
326.999	328.674	330.166
0.43888	0.75821	0.83469
0.26419	0.47907	0.20873
15		
346.825	348.578	350.160
0.43761	0.75814	0.83099
0.28040	0.50738	0.22610

PSI=360: Z(TIP)=0.235
R(TIP)=0.829

16		
366.533	368.455	370.044
0.43603	0.75795	0.82756
0.29741	0.53676	0.24464
17		
386.215	388.318	389.926
0.43400	0.75788	0.82438
0.31532	0.56732	0.26463
18		
405.983	408.187	409.886
0.43182	0.75829	0.82141
0.33352	0.59868	0.28437
19		
425.811	428.069	429.863
0.42968	0.75944	0.81866
0.35160	0.63029	0.30354
20		
445.667	447.978	449.857
0.42755	0.76100	0.81659
0.36936	0.65926	0.32219
21		
465.548	467.898	469.864
0.42548	0.76164	0.81547
0.38668	0.68215	0.34040
22		
485.450	487.845	489.853
0.42354	0.76085	0.81484
0.40365	0.70163	0.35824
23		
505.350	507.803	509.830
0.42174	0.75906	0.81418
0.42048	0.72158	0.37579
24		
525.219	527.780	529.791
0.41996	0.75623	0.81342
0.43749	0.74248	0.39341

PSI=540: Z(TIP)=0.402
R(TIP)=0.813

25		
545.040	547.780	549.713
0.41803	0.75233	0.81253
0.45494	0.76469	0.41172

26

564.840	567.791	569.637
0.41592	0.74754	0.81142
0.47290	0.78845	0.43077

27

584.655	587.789	589.598
0.41378	0.74225	0.81007
0.49115	0.81364	0.44999

28

604.490	607.760	609.574
0.41166	0.73691	0.80851
0.50941	0.83879	0.46907

29

624.333	627.704	629.571
0.40924	0.73095	0.80808
0.52732	0.86195	0.48780

30

644.182	647.640	649.597
0.40613	0.72367	0.81007
0.54453	0.88270	0.50595

31

664.087	667.653	669.560
0.40263	0.71605	0.81327
0.56120	0.90171	0.52358

32

684.031	687.708	689.498
0.39916	0.70918	0.81628
0.57767	0.91969	0.54096

33

703.982	707.775	709.421
0.39579	0.70310	0.81902
0.59418	0.93690	0.55837

PSI=720: Z(TIP)=0.567
R(TIP)=0.820

34

723.915	727.828	729.329
0.39247	0.69773	0.82147
0.61089	0.95347	0.57609

35

743.825	747.850	749.246
0.38922	0.69300	0.82365
0.62784	0.96945	0.59412

36

763.719	767.848	769.194
0.38614	0.68882	0.82553
0.64494	0.98481	0.61215

37

783.607	787.826	789.160
0.38334	0.68503	0.82708
0.66204	0.99948	0.62993