APPENDIX A

Visual basic code for Crowley's cone test: D3 scheme

```
**********
CLS
CLEAR

DIM u(35, 35), v(35, 35), c(35, 35), d(35, 35), dx(35, 35), dy(35, 35), xp(35,
&35), yp(35, 35)

h = 1
ll = 1
HA = 35
VA = 35
ta = 1 * 288
ww = 7.2722
s = (2 * 3.141593) / (ta * ww)

'******************************************************
OPEN "c:/matlab/vogx.m" FOR OUTPUT AS #2

'******************************************************
'**** Velocity field

FOR i = 1 TO 35
x = (i - 18) * h
FOR j = 1 TO 35
v(i, j) = -ww * x
NEXT j
NEXT i

FOR j = 1 TO 35
y = (j - 18) * ll
FOR i = 1 TO 35
u(i, j) = ww * y
NEXT i
NEXT j

'******************************************************
'**** Initial cone distribution

FOR i = 1 TO 35
x = (i - 18) * h
FOR j = 1 TO 35
y = (j - 18) * ll
IF (x + 8) ^ 2 + y ^ 2 <= 16 THEN
c(i, j) = -(25) * SQR((x + 8) ^ 2 + y ^ 2) + 100
ELSE c(i, j) = 0
END IF
NEXT j
NEXT i

'******************************************************
'**** Calculation of departure points

FOR i = 3 TO 33
x = (i - 18) * h
FOR j = 3 TO 33
```

```
y = (j - 18) * ll

ux = (u(i + 1, j) - u(i - 1, j)) / (2 * h)
uy = (u(i, j + 1) - u(i, j - 1)) / (2 * ll)
vx = (v(i + 1, j) - v(i - 1, j)) / (2 * h)
vy = (v(i, j + 1) - v(i, j - 1)) / (2 * ll)
uxx = (u(i + 2, j) - 2 * u(i, j) + u(i - 2, j)) / (4 * h ^ 2)
uyy = (u(i, j + 2) - 2 * u(i, j) + u(i, j - 2)) / (4 * ll ^ 2)
uxy = (u(i + 1, j + 1) - u(i - 1, j + 1) - u(i + 1, j - 1)
& + u(i - 1, j - 1)) /(4 * h * ll)
vxx = (v(i + 2, j) - 2 * v(i, j) + v(i - 2, j)) / (4 * h ^ 2)
vyy = (v(i, j + 2) - 2 * v(i, j) + v(i, j - 2)) / (4 * ll ^ 2)
vxy = (v(i + 1, j + 1) - v(i + 1, j - 1) - v(i - 1, j + 1)
& + v(i - 1, j - 1)) / &(4 * h * ll)

dx(i, j) = x - s * u(i, j) + ((s ^ 2) / 2) * (u(i, j) * ux + v(i, j) * uy)
& - ((s ^ 3) / 6) * ((u(i, j) ^ 2) * uxx + u(i, j) * (ux ^ 2) + 2 * u(i, j) *
& v(i, j) * uxy + u(i, j) * vx * uy + v(i, j) * uy * ux + (v(i, j) ^ 2) * (uyy)
& + v(i, j) * &vy * uy)
dy(i, j) = y - s * v(i, j) + ((s ^ 2) / 2) * (u(i, j) * vx + v(i, j) * vy)
& - ((s ^ 3) / 6) * ((u(i, j) ^ 2) * vxx + u(i, j) * ux * vx + 2 * u(i, j) *
& v(i, j) * vxy + u(i, j) * vx * vy + v(i, j) * uy * vx + (v(i, j) ^ 2) * (vyy)
& + & v(i, j) * &((vy) ^ 2))

FOR l = 1 TO 35
xt = (l - 18) * h
yt = (l - 18) * ll
IF dx(i, j) > xt THEN xp(i, j) = l + 1
IF dy(i, j) > yt THEN yp(i, j) = l + 1
NEXT l

NEXT j
NEXT i

'************************************
'**** Bicubic interpolation

FOR t = 1 TO ta
PRINT t

FOR i = 3 TO 33
FOR j = 3 TO 33

d(i, j) = 0
w = 0
lp = 0
som = 0
som1 = 0

'**** Bilinear interpolation may be required

IF xp(i, j) = 34 OR yp(i, j) = 34 THEN lp = 1
IF xp(i, j) = 3 OR yp(i, j) = 3 THEN lp = 1

FOR k = lp + xp(i, j) - 2 TO xp(i, j) + 1 - lp
xk = (k - 18) * h
FOR l = lp + yp(i, j) - 2 TO yp(i, j) + 1 - lp
yl = (l - 18) * ll

somx = 1
```

```
FOR m = lp + xp(i, j) - 2 TO xp(i, j) + 1 - lp
xm = (m - 18) * h
IF m <> k THEN somx = ((dx(i, j) - xm) / (xk - xm)) * somx
NEXT m

somy = 1
FOR m = lp + yp(i, j) - 2 TO yp(i, j) + 1 - lp
ym = (m - 18) * ll
IF m <> l THEN somy = ((dy(i, j) - ym) / (yl - ym)) * somy
NEXT m

w = somx * somy
som = som + w * c(k, l)
soml = soml + w

NEXT l
NEXT k

d(i, j) = som

NEXT j
NEXT i


FOR i = 1 TO 35
FOR j = 1 TO 35
c(i, j) = d(i, j)
NEXT j
NEXT i

NEXT t

'*********************************
'**** Writing to Matlab

PRINT #2, "x=[";
FOR i = 1 TO HA - 1
x = (i - 18) * h
PRINT #2, x; ",";
NEXT i
i = HA
x = (i - 18) * h
PRINT #2, x; "];"
PRINT #2, " "

PRINT #2, "y=[";
FOR j = 1 TO VA - 1
y = (j - 18) * ll
PRINT #2, y; ",";
NEXT j
j = VA
y = (j - 18) * ll
PRINT #2, y; "];"
PRINT #2, " "
PRINT #2, "[X,Y]=meshgrid(x,y);"

PRINT #2, " "

PRINT #2, "Z=[";
FOR j = 1 TO VA - 1
FOR i = 1 TO HA
```

```
PRINT #2, c(i, j);
PRINT #2, " ";
NEXT i
PRINT #2, " "
NEXT j

j = VA

FOR i = 1 TO HA - 1
PRINT #2, c(i, j);
PRINT #2, " ";
NEXT i
i = HA
PRINT #2, c(i, j);
PRINT #2, "];"
PRINT #2, " "
PRINT #2, "surf(X,Y,Z)"
CLOSE 2

'*******************************************************
END
'*******************************************************
'*******************************************************
```

APPENDIX B

Fortran code for Smolarkiewicz's deformational flow: D_3 scheme

```
**********
      program smolar

      integer A, L, CI, IER, t, lp, xp(101,101), yp(101,101),ta
      real tt,aa
      real pi, kk, tsi(101,101), c(101,101), ALEV(13),KEELY(10), TR(6)
      real u(101,101), v(101,101), s, dx(101,101), dy(101,101)
      real zt, d(101,101), som, som1, xk, yl, xm, ym, somx, somy, w
      real f(101,101),b(101,101)
      character*(4) LABEL

      A=8
      L=100
      pi=3.141592654
      kk=4*pi/L
      s=0.7

c**** Iterations correspond to the times in the paper of Staniforth et al. (1987)
c     ta=19
c     ta=38
c     ta=57
c     ta=75
c     ta=377
      ta=3768

c**** Calculate grid point values of.....
      do i = 1, 101
      x = (i-1)
      do j = 1, 101
      y = (j-1)
c**** Streamlines
      tsi(i,j) = A*(sin(kk*x))*(cos(kk*y))
c**** Velocity field
      u(i,j) = A*(kk)*(sin(kk*x))*(sin(kk*y))
      v(i,j) = A*(kk)*(cos(kk*x))*(cos(kk*y))
      enddo
      enddo

c**** Calculate the grid point values of initial cone distribution
      do i = 1, 101
      x = (i-1)
      do j = 1, 101
      y = (j-1)
      c(i,j) = 0.0
      if ((x-50)**2 + (y-50)**2 .le. 225.0) c(i,j) =
     & (-1.0/15.0)*(((x-50)**2 + (y-50)**2)**(0.5)) +1
      b(i,j)=c(i,j)
      enddo
      enddo

c**********
c**** Determination of departure points

      do i=2,100
```

```
      x=i-1
      do j=2,100
      y=j-1

      ux = (u(i+1,j)-u(i-1,j))/2.
      uy = (u(i,j+1)-u(i,j-1))/2.
      vx = (v(i+1,j)-v(i-1,j))/2.
      vy = (v(i,j+1)-v(i,j-1))/2.
      uxx = (u(i+2,j)-2.*u(i,j)+u(i-2,j))/4.
      uyy = (u(i,j+2)-2.*u(i,j)+u(i,j-2))/4.
      uxy = (u(i+1,j+1)-u(i-1,j+1)-u(i+1,j-1)+u(i-1,j-1))/4.
      vxx = (v(i+2,j)-2.*v(i,j)+v(i-2,j))/4.
      vyy = (v(i,j+2)-2.*v(i,j)+v(i,j-2))/4.
      vxy = (v(i+1,j+1)-v(i+1,j-1)-v(i-1,j+1)+v(i-1,j-1))/4.

      dx(i,j)=x-s*u(i,j)+((s**2)/2.)*(u(i,j)*ux+
     &v(i,j)*uy)-((s**3)/6.)*((u(i,j)**2)*uxx+u(i,j)
     &*(ux**2.)+2.*u(i,j)*v(i,j)*uxy+u(i,j)*vx*uy+
     &v(i,j)*uy*ux+(v(i,j)**2)*(uyy)+v(i,j)*vy*uy)

      dy(i,j)=y-s*v(i,j)+((s**2)/2.)*(u(i,j)*vx+
     &v(i,j)*vy)-((s**3)/6.)*((u(i,j)**2)*vxx+u(i,j)
     &*ux*vx+2.*u(i,j)*v(i,j)*vxy+u(i,j)*vx*vy+
     &v(i,j)*uy*vx+(v(i,j)**2)*(vyy)+v(i,j)*((vy)**2))

      do l=1,101
      xt=(l-1)
      yt=(l-1)
      if (dx(i,j).gt.xt) xp(i,j)=l+1
      if (dy(i,j).gt.yt) yp(i,j)=l+1
      enddo

      enddo
      enddo

c*********************************************************
c*********************************************************
c****  Bicubic interpolation
      do t=1,ta

      do i=2,100
      do j=2,100

      d(i,j)=0
      w=0
      lp=0
      som=0
      som1=0

      if (xp(i,j).eq.101) lp=1
      if (yp(i,j).eq.101) lp=1
      if (xp(i,j).eq.2) lp=1
      if (yp(i,j).eq.2) lp=1

      do k=lp+xp(i,j)-2,xp(i,j)+1-lp
      xk=k-1
      do l=lp+yp(i,j)-2,yp(i,j)+1-lp
      yl=l-1

      somx=1
      do m=lp+xp(i,j)-2,xp(i,j)+1-lp
```

```
      xm=m-1
      if(m.ne.k) somx=((dx(i,j)-xm)/(xk-xm))*somx
      enddo

      somy=1
      do m=lp+yp(i,j)-2,yp(i,j)+1-lp
      ym=m-1
      if (m.ne.l) somy=((dy(i,j)-ym)/(yl-ym))*somy
      enddo

      w=somx*somy
      som=som+w*c(k,l)
      soml=soml+w

      enddo
      enddo

      d(i,j)=som

      enddo
      enddo

      do i=1,101
      do j=1,101
      c(i,j)=d(i,j)
      enddo
      enddo

      enddo
c*************************************************************
c****  Conservation properties
      con1=0
      con2=0
      con3=0
      cona=0
      conb=0
      conc=0

      do i=1,101
      do j=1,101
      con1=con1+c(i,j)
      con2=con2+(c(i,j))**2
      con3=con3+abs(c(i,j))
      cona=cona+b(i,j)
      conb=conb+(b(i,j))**2
      conc=conc+abs(b(i,j))
      enddo
      enddo

      con1=con1/cona
      con2=con2/conb
      con3=con3/conc
      print*, con1,con2,con3

c*************************************************************
c*************************************************************
c****  Writing out for MATLAB
      open(1,file='/frae/WEERLIG/MAT/matd36.m',form='formatted'
     &,recl=100000)
      write(1,20) c
 20   format (101f20.10)
```

```
c****  Writing statistics
       open(2,file='/frae/WEERLIG/STA/stad36')
       write(2,30) con1
       write(2,30) con2
       write(2,30) con3
 30    format (f20.10)
c*********************************************************
c****  Interactive display using PGPLOT

c****  Open plot
       IF (PGOPEN('?') .LT. 1) STOP

c****  Set contours  of streamlines to be plotted
       do i = 1,13
       ALEV(I) = -6 + (i-1)
       enddo

c****  Set contours of cone to be plotted
       KEELY(1) = 0.02
       KEELY(2) = 0.1
       KEELY(3) = 0.2
       KEELY(4) = 0.3
       KEELY(5) = 0.4
       KEELY(6) = 0.5
       KEELY(7) = 0.6
       KEELY(8) = 0.7
       KEELY(9) = 0.8
       KEELY(10) = 0.9

c****  Set axis
       CALL PGENV(0.,100.,0.,100.,1,1)
       TR(1) = -1.0
       TR(2) = 1.0
       TR(3) = 0.0
       TR(4) = -1.0
       TR(5) = 0.0
       TR(6) = 1.0

c****  Label axis, title
       CALL PGLAB('x', 'y', 'Smolarkiewicz`s Test  D3 scheme  t=T')

c****  Set colour and plot contours of streamlines
       CALL PGSCI(3)
       CALL PGCONT(tsi,101,101,1,101,1,101,ALEV,-7,TR)
       CALL PGSCI(4)
       CALL PGCONT(tsi,101,101,1,101,1,101,ALEV(8),-6,TR)

c****  Set colour and plot contours of cone
       CALL PGSCI(1)
       CALL PGCONT(b,101,101,1,101,1,101,KEELY,10,TR)

c****  Set colour and plot contours of solution
       CALL PGSCI(2)
       CALL PGCONT(c,101,101,1,101,1,101,KEELY,10,TR)

c**** CLOSE PLOT
       CALL PGCLOS

       end
```