付 録B　1次元流体プログラム解答例

主プログラム

```
      program roe
      implicit real*8 (a-h,o-z)
      parameter(nt=80)
      parameter(nx=100)
      parameter(itorder=2)
      parameter(gamma=1.40d0)
      dimension q(3,0:nx),ql(3,nx),qr(3,nx)
      dimension flx(3,nx),flxco(3,nx)
      dimension qt(3,0:nx)
      common /flag/imuscl, imusclsuperbee, ico, icosuperbee

C initial condition etc.
      call initial(nx,gamma,rho0,rho1,p0,p1,u0,u1,q)
      t=0.0d0
      tend=14.40d0
      dt=tend/dble(nt)
      dx=1.0d0
      dtdx=dt/dx
      imuscl=0
      imusclsuperbee=0
      ico=3
      icosuperbee=1

      if (itorder.eq.1) then
C 1st order in time
        do it=1,nt
          call copyvariable(nx,q,qt)
          call makeleftright(nx,gamma,q,qt,ql,qr)
          call roescheme(nx,gamma,ql,qr,flx,flxco)
          if (ico.ne.0) then
            call addcoflux(nx,flxco,flx)
          endif
          call advance(nx,dtdx,flx,qt,q)
          t=t+dt
        enddo
      endif
```

(*continue*)

主プログラム

```
37
38       if (itorder.eq.2) then
39  C 2nd order in time
40           do it=1,nt
41  C first step
42             imuscl2=imuscl
43             imuscl=0
44             call copyvariable(nx,q,qt)
45             call makeleftright(nx,gamma,q,qt,ql,qr)
46             call roescheme(nx,gamma,ql,qr,flx,flxco)
47             call advance(nx,dtdx/2.0d0,flx,q,qt)
48             imuscl=imuscl2
49  C second step
50             ico2=ico
51             ico=0
52             call makeleftright(nx,gamma,q,qt,ql,qr)
53             call roescheme(nx,gamma,ql,qr,flx,flxco)
54             if (ico2.ne.0) then
55               call addcoflux(nx,flxco,flx)
56             endif
57             call copyvariable(nx,q,qt)
58             call advance(nx,dtdx,flx,qt,q)
59             ico=ico2
60             t=t+dt
61           enddo
62       endif
63  C output etc.
64       open(10,file='out.d',status='unknown',form='formatted')
65       do i=1,nx
66         write(10,*) dble(i),
67      $    q(1,i),q(2,i)/q(1,i),
68      $    (gamma-1.0d0)*(q(3,i)-q(2,i)**2/q(1,i)/2.0d0)
69       enddo
70       close(10)
71       stop
72       end
```

roeaverage

```
1        subroutine roeaverage(rl,ul,hl,rr,ur,hr,au,ah)
2        implicit real*8 (a-h,o-z)
3        sqrl=dsqrt(rl)
4        sqrr=dsqrt(rr)
5        au=(sqrl*ul+sqrr*ur)/(sqrl+sqrr)
6        ah=(sqrl*hl+sqrr*hr)/(sqrl+sqrr)
7        return
8        end
```

fluxatmesh

```
      subroutine fluxatmesh(rho,u,p,e,f)
      implicit real*8 (a-h,o-z)
      dimension f(3)
      f(1)=rho*u
      f(2)=rho*u**2+p
      f(3)=(e+p)*u
      return
      end
```

fluxabsA

```
      subroutine fluxabsA(rm,sl,sr,rl,ul,el,cl,rr,ur,er,cr,fa)
      implicit real*8 (a-h,o-z)
      dimension rm(3),sr(3,3),sl(3,3),fa(3)
      dimension dq(3)
      dimension absrmmod(3)

      dq(1)=rr-rl
      dq(2)=rr*ur-rl*ul
      dq(3)=er-el
      call fancorrect(ul,ur,cl,cr,rm,absrmmod)
      do i=1,3
        fa(i)=0.0d0
        do j=1,3
          do k=1,3
            fa(i)=fa(i)+sr(i,j)*absrmmod(j)*sl(j,k)*dq(k)
          enddo
        enddo
      enddo
      return
      end

CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      subroutine fancorrect(ul,ur,cl,cr,rm,absrmmod)
      implicit real*8 (a-h,o-z)
      dimension rm(3),absrmmod(3), rml(3),rmr(3)
      rml(1)=ul-cl
      rml(2)=ul
      rml(3)=ul+cl
      rmr(1)=ur-cr
      rmr(2)=ur
```

(continue)

fluxabsA

```
31      rmr(3)=ur+cr
32      do j=1,3
33        if (rml(j).lt.0.0d0.and.0.0d0.lt.rmr(j)) then
34          absrmmod(j)=dabs(rm(j))+(rmr(j)-rml(j))/2.0d0
35        else
36          absrmmod(j)=dabs(rm(j))
37        endif
38      enddo
39      return
40      end
```

muscl

```
 1      subroutine muscl(nx,gamma,q,qt,ql,qr)
 2      implicit real*8 (a-h,o-z)
 3      dimension q(3,0:nx),qt(3,0:nx),ql(3,nx),qr(3,nx)
 4      dimension rm(3),sr(3,3),sl(3,3)
 5      dimension dqm(3),dqp(3)
 6      dimension alpm(3),alpp(3),alpl(3),alpr(3)
 7      common /flag/imuscl, imusclsuperbee, ico, icosuperbee
 8
 9      fminmod(a,b)=dsign(1.0d0,a)*
10     $  max(0.0d0,min(dabs(a),dsign(1.0d0,a)*b))
11      fsuperbee(a,b,w)=dsign(1.0d0,a)*
12     $  max(0.0d0,min(w*dsign(1.0d0,a)*b,dabs(a)),
13     $  min(dsign(1.0d0,a)*b,w*dabs(a)))
14
15      gam1=gamma-1.0d0
16      isuperbee=imusclsuperbee
17      xi=1.0d0/3.0d0
18      if (isuperbee.eq.1) then
19        wcomp=2.0d0
20      else
21        wcomp=(3.0d0-xi)/(1.0d0-xi)
22      endif
23      do i=1,nx-1
24        do j=1,3
25          dqm(j)=q(j,i)-q(j,i-1)
26          dqp(j)=q(j,i+1)-q(j,i)
27        enddo
28        call variables(nx+1,i+1,q,gamma,r0,u0,e0,p0,h0)
29        call makematrix(gam1,u0,h0,sl,sr,rm)
30        do j=1,3
31          alpm(j)=0.0d0
32          alpp(j)=0.0d0
33          do k=1,3
```

(*continue*)

muscl

```
34            alpm(j)=alpm(j)+sl(j,k)*dqm(k)
35            alpp(j)=alpp(j)+sl(j,k)*dqp(k)
36          enddo
37          if (isuperbee.eq.0) then
38            alpr(j)=fminmod(alpp(j),wcomp*alpm(j))
39            alpl(j)=fminmod(alpm(j),wcomp*alpp(j))
40          else
41            alpr(j)=fsuperbee(alpp(j),alpm(j),wcomp)
42            alpl(j)=fsuperbee(alpm(j),alpp(j),wcomp)
43          endif
44        enddo
45        do j=1,3
46          qr(j,i)=qt(j,i)
47          ql(j,i+1)=qt(j,i)
48          do k=1,3
49            qr(j,i)=qr(j,i)
50     $        -sr(j,k)
51     $        *((1.0d0-xi)*alpr(k)+(1.0d0+xi)*alpl(k))/4.0d0
52            ql(j,i+1)=ql(j,i+1)
53     $        +sr(j,k)
54     $        *((1.0d0-xi)*alpl(k)+(1.0d0+xi)*alpr(k))/4.0d0
55          enddo
56        enddo
57      enddo
58C boundary :  ql(1), qr(nx)
59      do j=1,3
60        ql(j,1)=qt(j,0)
61        qr(j,nx)=qt(j,nx)
62      enddo
63      return
64      end
```

coscheme

```
      subroutine coscheme(nx,i,ql,qr,sl,sr,rm,flxco)
      implicit real*8 (a-h,o-z)
      dimension ql(3,nx),qr(3,nx),flxco(3,nx)
      dimension rm(3),sr(3,3),sl(3,3)
      dimension dqm(3),dq(3),dqp(3),alpm(3),alp(3),alpp(3)
      common /flag/imuscl, imusclsuperbee, ico, icosuperbee

      fminmod(a,b)=dsign(1.0d0,a)*
     $  max(0.0d0,min(dabs(a),dsign(1.0d0,a)*b))
      fsuperbee(a,b,w)=dsign(1.0d0,a)*
     $  max(0.0d0,min(w*dsign(1.0d0,a)*b,dabs(a)),
     $  min(dsign(1.0d0,a)*b,w*dabs(a)))

      isuperbee=icosuperbee
      xi=1.0d0/3.0d0
      if (isuperbee.eq.1) then
        wcomp=2.0d0
      else
        wcomp=(3.0d0-xi)/(1.0d0-xi)
      endif
      if (i.eq.1) then
        im=1
      else
        im=i-1
      endif
      if (i.eq.nx) then
        ip=nx
      else
        ip=i+1
      endif
      do j=1,3
        dqm(j)=qr(j,im)-ql(j,im)
        dq(j)=qr(j,i)-ql(j,i)
        dqp(j)=qr(j,ip)-ql(j,ip)
      enddo
      do j=1,3
        alpm(j)=0.0d0
        alp(j)=0.0d0
        alpp(j)=0.0d0
        do k=1,3
          alpm(j)=alpm(j)+sl(j,k)*dqm(k)
          alp(j)= alp(j) +sl(j,k)*dq(k)
          alpp(j)=alpp(j)+sl(j,k)*dqp(k)
        enddo
      enddo
      do k=1,3
        flxco(k,i)=0.0d0
        do j=1,3
          if (isuperbee.eq.0) then
            dsigmr=fminmod(alpp(j)*(rm(j)-dabs(rm(j))),
```

(*continue*)

coscheme

```
51      $        wcomp*alp(j)*(rm(j)-dabs(rm(j))))/2.0d0
52              dsigm=fminmod(alp(j)*(rm(j)-dabs(rm(j))),
53      $        wcomp*alpp(j)*(rm(j)-dabs(rm(j))))/2.0d0
54              dsigp=fminmod(alp(j)*(rm(j)+dabs(rm(j))),
55      $        wcomp*alpm(j)*(rm(j)+dabs(rm(j))))/2.0d0
56              dsigpl=fminmod(alpm(j)*(rm(j)+dabs(rm(j))),
57      $        wcomp*alp(j)*(rm(j)+dabs(rm(j))))/2.0d0
58            else
59              dsigmr=fsuperbee(alpp(j)*(rm(j)-dabs(rm(j))),
60      $        alp(j)*(rm(j)-dabs(rm(j))),wcomp)/2.0d0
61              dsigm=fsuperbee(alp(j)*(rm(j)-dabs(rm(j))),
62      $        alpp(j)*(rm(j)-dabs(rm(j))),wcomp)/2.0d0
63              dsigp=fsuperbee(alp(j)*(rm(j)+dabs(rm(j))),
64      $        alpm(j)*(rm(j)+dabs(rm(j))),wcomp)/2.0d0
65              dsigpl=fsuperbee(alpm(j)*(rm(j)+dabs(rm(j))),
66      $           alp(j)*(rm(j)+dabs(rm(j))),wcomp)/2.0d0
67            endif
68            flxco(k,i)=flxco(k,i)-
69      $      ((1.0d0-xi)*dsigmr
70      $      +(1.0d0+xi)*dsigm
71      $      -(1.0d0+xi)*dsigp
72      $      -(1.0d0-xi)*dsigpl)*sr(k,j)/4.0d0
73          enddo
74        enddo
75        return
76        end
```