風上差分（1/4）

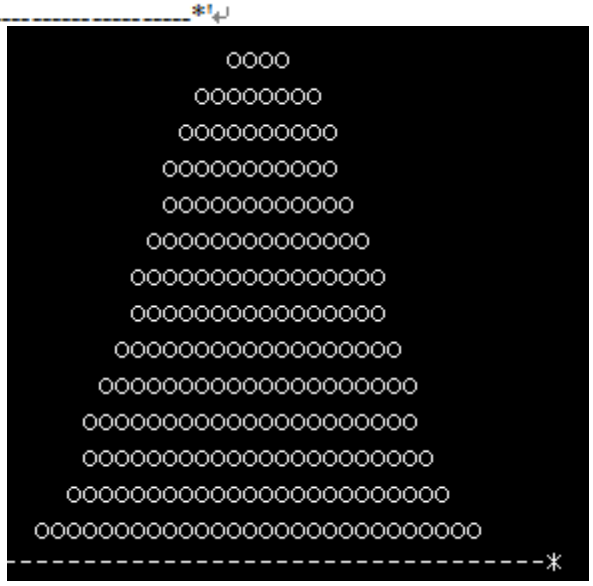
$$u_i^{n+1} = \begin{cases} u_i^n - \dfrac{c\Delta t}{\Delta x}\left(u_i^n - u_{i-1}^n\right) & c \geq 0 \\ u_i^n - \dfrac{c\Delta t}{\Delta x}\left(u_{i+1}^n - u_i^n\right) & c < 0 \end{cases}$$

ソースファイル名：advection_ori.for，実行ファイル名：advection_ori.out

```
$ cp ../common/advection_ori.for .
$ cp advection_ori.for . advection.for
$ ls
$ vi advection.for
$ gfortran -o advection.out advection.for
$ ./advection.out
```

```
      program advdif
      parameter(kx=100)
      real*8 u(kx),un(kx)
      real*8 dt,dx,c,ntime
      integer nfnl,imax
      character a(kx,30)
      dt=0.02; dx=0.025;c= 1.0; nfnl=100
      imax = 70
      ntime=0.

c.. initial condition
      do i=2,imax-1
       un(i)=0
      end do
      do i=5,20
       un(i)=1
      end do
      do i=2,imax-1
       u(i)=un(i)
      end do

      ntime=0

200   do n=1,nfnl
       ntime=ntime+dt
       do inf=1,99999999; end do
```

```
c.. boundary condition
        un(1)=0.; un(imax)=0.
c.. upwind
        do i=2,imax-1
          u(i)=un(i)-c*dt/dx*(un(   )   -un(   ))
        end do

c.. data transfer
        do i=1,imax
          un(i)=u(i)
        end do
c.. graphics
        write(*,*)ntime

        do i=1, imax; do j=1,20
          a(i,j)=' '
        end do; end do

        do i=1,imax
          nb=int(u(i)*15)
          do j=1,nb
            a(i,j)='o'
          end do
        end do

        do j=20,1,-1
          write(*,*)(a(i,j),i=1,imax)
        end do
        write(*,*)'*----------*'

      end do
c
      end
```



風上差分＋中心差分（2/4）

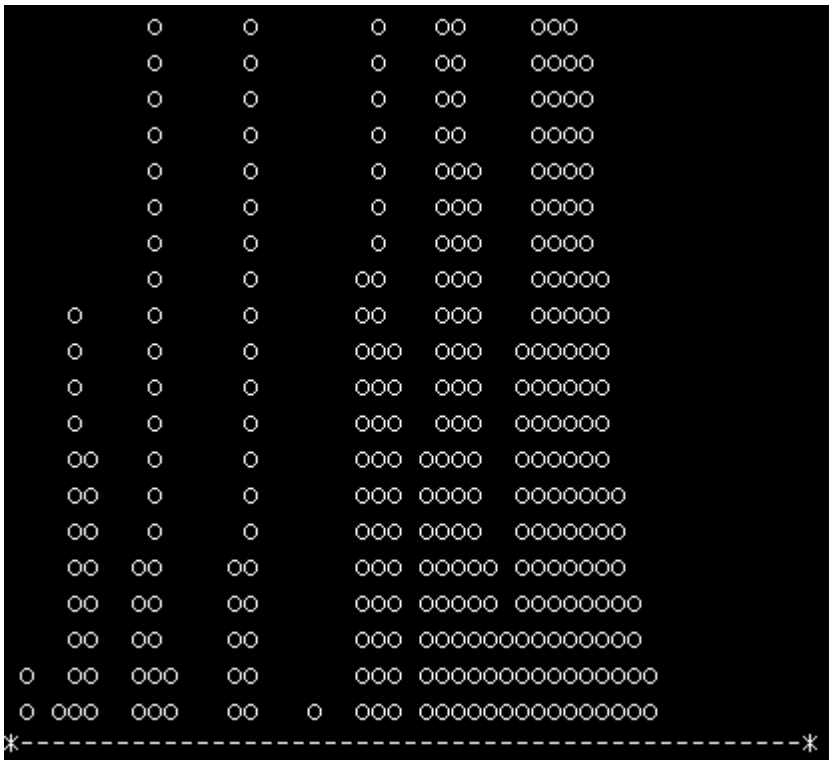
$$u_i^{n+1} = u_i^n - \frac{c\Delta t}{2\Delta x}\left(u_{i+1}^n - u_{i-1}^n\right)$$

ソースファイル名：advection.for，実行ファイル名：advection.out

```
$ vi advection.for
$ gfortran -o advection.out advection.for
$ ./advection.out
```

```
      program advdif

      parameter(kx=100)
      real*8  u(kx),un(kx)
      real*8  dt,dx,c,ntime
      integer nfnl,imax,sflag
      character a(kx,30)

      dt=0.02; dx=0.025;c=  1.0; nfnl=100
      imax  =70
      sflag=  1
c
      write(*,*)' Scheme?'
      write(*,*)'(1) upwind'
      write(*,*)'(2) ftcs'
      read(*,*) sflag
      ntime=0.

c.. initial condition

      do i=2,imax-1
        un(i)=0
      end do
      do i=5,20
        un(i)=1
      end do
      do i=2,imax-1
        u(i)=un(i)
      end do

      ntime=0
```

```
200     do n=1,nfnl
        ntime=ntime+dt
        do inf=1,99999999;end do
c.. boundary condition
        un(1)=0.;un(imax)=0.
c
c.. calculation

        if(sflag.eq.1)then
c.. upwind
        do i=2,imax-1
        u(i)=un(i)-c*dt/dx*(un(   )-un(   ))
        end do
        else if(sflag.eq.2)then
c.. ftcs
        do i=2,imax-1
        u(i)=un(i)-c*dt/dx*0.5*(un(   )-un(   ))
        end do

        endif

c.. data transfer
        do i=1,imax
        un(i)=u(i)
        end do

c.. graphics

      (略)

      end
```



風上差分＋中心差分＋CIP（3/4）

$$u_i^{n+1} = a_i\xi^3 + b_i\xi^2 + g_i^n\xi + u_i^n,\quad g_i^{n+1} = 3a_i\xi^2 + 2b_i\xi + g_i^n$$

$$a_i = \frac{g_i^n + g_{iup}^n}{D^2} + \frac{2\left(u_i^n + u_{iup}^n\right)}{D^3},\quad b_i = \frac{3\left(u_{iup}^n + u_i^n\right)}{D^2} - \frac{2g_i^n + g_{iup}^n}{D}$$

ここで，$g_i^n = \partial u_i^n / \partial x$，$\xi = -c\Delta t$，$iup = i - 1$，$D = -\Delta x$である．

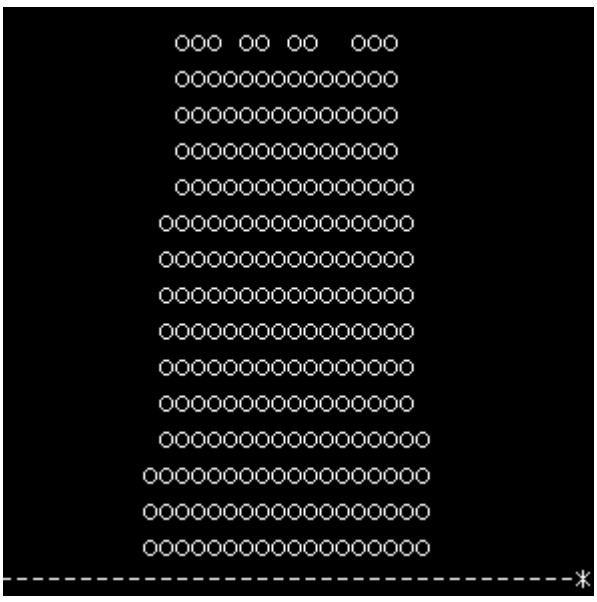
ソースファイル名：advection.for，実行ファイル名：advection.out

```
$ vi advection.for
$ gfortran -o advection.out advection.for
$ ./advection.out
```

```
      program advdif

      parameter(kx=100)
      real*8 u(kx),un(kx)
      real*8 g(kx),gn(kx)
      real*8 dt,dx,c,ntime
      real*8 zeta,ai,bi,d
      integer nfnl,imax,sflag
      character a(kx,30)

      dt=0.02;dx=0.025;c=-1.0;nfnl=100
      imax =70
      sflag= 1
c
      write(*,*)' Scheme?'
      write(*,*)'(1)upwind'
      write(*,*)'(2)ftcs'
      write(*,*)'(3)CIP'
      read(*,*)sflag
      ntime=0.

c..initial condition

      do i=2,imax-1
        un(i)=0
      end do
      do i=5,20
        un(i)=1
```

```
      end do
      do i=2,imax-1
      u(i)=un(i)
      end do

      if(sflag.eq.3)then
      do i=2,imax-1
      gn(i)=0.5*(un(i+1)-un(i-1))/dx
      g(i)=gn(i)
      end do
      endif

      ntime=0

200   do n=1,nfnl
      ntime=ntime+dt
      do inf=1,99999999;end do
c..boundary condition
      if(sflag.eq.3)then
      un(1)=0;un(imax)=0
      gn(1)=0;gn(imax)=0
      else
      un(1)=0.;un(imax)=0.
      endif
c
c..calculation

      if(sflag.eq.1)then
c..upwind
      do i=2,imax-1
      u(i)=un(i)-c*dt/dx*   (un(   )   -un(   ))
      end do
      else if(sflag.eq.2)then
c..ftcs
      do i=2,imax-1
      u(i)=un(i)-c*dt/dx*0.5*(un(   )-un(   ))
      end do
      else if(sflag.eq.3)then
c..CIP
      zeta=-c*dt
```

```
        do i=2,imax-1
        if(c.ge.0) then; iup=i-1; d=-dx; endif
        if(c.lt.0) then; iup=i+1; d= dx; endif
        ai=(gn(i)+gn(iup))/d**2+2.*(un(i)-un(iup))/d**3
        bi=3.*(un(iup)-un(i ))/d**2-(2.*gn(i)+gn(iup))/d
        u(i)=
        g(i)=
        end do
      endif

c..data transfer
      if(sflag.eq.3)then
        do i=1,imax
        gn(i)=g(i)
        un(i)=u(i)
        end do
      else
        do i=1,imax
        un(i)=u(i)
        end do
      endif

c..graphics

      (略)

      end
```



風上差分＋中心差分＋CIP＋Lax-Wendroff（4/4）

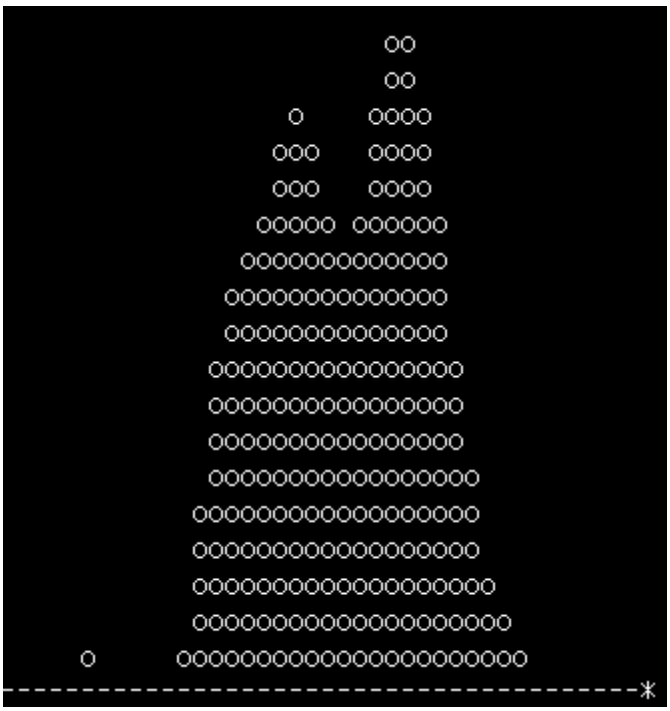
$$u_i^{n+1} = u_i^n - \frac{c\Delta t}{2\Delta x}\left(u_{i+1}^n - u_{i-1}^n\right) + \frac{(c\Delta t)^2}{2\Delta x^2}\left(u_{i+1}^n - 2u_i^n + u_{i-1}^n\right)$$

ソースファイル名：advection.for，実行ファイル名：advection.out

```
$ vi advection.for
$ gfortran -o advection.out advection.for
$ ./advection.out
```

```
      program advdif

      parameter(kx=100)
      real*8  u(kx), un(kx)
      real*8  g(kx), gn(kx)
      real*8  dt, dx, c, ntime
      real*8  zeta, ai, bi, d
      integer nfnl, imax, sflag
      character a(kx,30)

      dt=0.02; dx=0.025; c=  1.0; nfnl=100
      imax  =70
      sflag=  1
c
      write(*,*)' Scheme?'
      write(*,*)'(1) upwind'
      write(*,*)'(2) ftcs'
      write(*,*)'(3) CIP'
      write(*,*)'(4) Lax-Wendroff'
      read(*,*) sflag
      ntime=0.

      （略）

c.. calculation

      if(sflag.eq.1)then
c.. upwind
        do i=2,imax-1
          u(i)=un(i)-c*dt/dx*  (un(   )  -un(   ))
        end do
      else if(sflag.eq.2)then
```

```
c..ftcs
        do i=2,imax-1
        u(i)=un(i)-c*dt/dx*0.5*(un(   )-un(   ))
        end do
        else if(sflag.eq.3)then
c..CIP
        zeta=-c*dt

        do i=2,imax-1
        if(c.ge.0)then; iup=i-1; d=-dx; endif
        if(c.lt.0)then; iup=i+1; d= dx; endif
        ai=(gn(i)+gn(iup))/d**2+2.*(un(i)-un(iup))/d**3
        bi=3.*(un(iup)-un(i ))/d**2-(2.*gn(i)+gn(iup))/d
        u(i)=
        g(i)=
        end do
        else if(sflag.eq.4)then
c..Lax-Wendroff
        do i=2,imax-1
        u(i)=un(i)-c*dt/dx*0.5*(un(   )-un(   ))
      +      *(un(   )-2.*un(   )+un(   ))
        end do
        endif

c..data transfer
        if(sflag.eq.3)then
        do i=1,imax
        gn(i)=g(i)
        un(i)=u(i)
        end do
        else
        do i=1,imax
        un(i)=u(i)
        end do
        endif

c..graphics
      （略）

      end
```



風上差分＋中心差分＋CIP＋Lax-Wendroff+河村・桑原（Extra）

3 次精度の風上差分 ＝ 4 次中心差分 ＋ 4 階微分項

※1 次の風上差分 ＝ 2 次中心差分 ＋2 階微分項

$$u_i^{n+1} = u_i^n - \frac{c\Delta t}{12\Delta x}\left(-u_{i+2}^n + 8u_{i+1}^n - 8u_{i-1}^n + u_{i-2}^n\right) - \alpha\frac{|c|\Delta t}{\Delta x}\left(u_{i+2}^n - 4u_{i+1}^n + 6u_i^n - 4u_{i-1}^n + u_{i-2}^n\right)$$

ソースファイル名：advection.for，実行ファイル名：advection.out

```
$ vi    advection.for
$ gfortran   -o     advection.out    advection.for
$ ./ advection.out
```

```
      program advdif

      parameter(kx = 100)
      real*8    u(kx), un(kx)
      real*8    g(kx), gn(kx)
      real*8    dt, dx, c, ntime
      real*8    zeta, ai, bi, d, alpha
      integer nfnl, imax, sflag
      character a(kx,30)

      dt = 0.02; dx = 0.025;c =    1.0; nfnl = 100
      dx = 0.1; alpha = 0.2
      imax    = 70
      sflag =    1
c
      write(*,*)' Scheme?'
      write(*,*)'(1) upwind'
      write(*,*)'(2) ftcs'
      write(*,*)'(3) CIP'
      write(*,*)'(4) Lax-Wendroff '
      write(*,*)'(5) Kawamura-Kuwabara'
      read(*,*) sflag
      ntime = 0.

     (略)

200     do n = 1, nfnl
          ntime = ntime + dt
          do inf = 1,99999999; end do
c.. boundary condition
```

```
          if(sflag .eq. 3) then
            un(1)= 0; un(imax) = 0
            gn(1)= 0; gn(imax) = 0
          else if(sflag .eq. 5) then
            un(1)= 0.; un(imax   ) = 0.
            un(2)= 0.; un(imax-1) = 0.
          else
            un(1)= 0.; un(imax) = 0.
          endif
c
c.. calculation

          if(sflag .eq. 1) then
c.. upwind
            do i = 2, imax - 1
              u(i) = un(i) - c*dt/dx*     (un(    )   - un(    ))
            end do
          else if(sflag .eq. 2) then
c.. ftcs
            do i = 2, imax - 1
              u(i) = un(i) - c*dt/dx*0.5*(un(    ) - un(    ))
            end do
          else if(sflag .eq. 3) then
c.. CIP
            zeta = -c*dt

            do i = 2, imax - 1
              if(c .ge. 0) then; iup = i - 1; d = -dx; endif
              if(c .lt. 0) then; iup = i + 1; d =   dx; endif
              ai = (gn(i) + gn(iup))/d**2 + 2.*(un(i) - un(iup))/d**3
              bi = 3.*(un(iup) - un(i   ))/d**2 - (2.*gn(i) + gn(iup))/d
              u(i) =
              g(i) =
            end do
          else if(sflag .eq. 4) then
c.. Lax-Wendroff
            do i = 2, imax - 1
              u(i) = un(i) - c*dt/dx*0.5*(un(    ) - un(    ))
     .        +                    *(un(    ) – 2.*un(    ) + un(    ))
            end do
```

```
      else if(sflag .eq. 5) then
c.. Kawamura-Kuwabara
        do i = 3, imax - 2
          u(i) = un(i)
     .          - c*dt/dx*(- un(   ) + 8.*un(   )
     .                     + un(   ) - 8.*un(   ))/12.
     .    - alpha*c*dt/dx*( un(   ) - 4.*un(   ) + 6.*un(   )
     .                     + un(   ) - 4.*un(   ))
        end do
      end if

      (略)

      end
```