1　常微分方程式の差分解法

1.1　概要

常微分方程式 (Ordinary Defferential Equation, 以下 ODE) を厳密に解けない場合、コンピューターにより数値的に解く方法がある。それが差分法である。差分法では、時間発展の様子を各時間ステップごとに計算することで表す。そのとき、次のステップへの計算の手法と時間ステップの刻み幅により解は変化する。ここでは、厳密解が既知の下式 ODE について Euler 陽解法、Euler 陰解法、Crank-Nicolson 法の 3 つの手法を用いて数値解を求め、その精度を比較する。

$$\frac{df}{dt} = -\kappa f + \alpha \tag{1}$$

1.1.1　Euler 陽解法

下式の ODE

$$\frac{df(t)}{dt} = S(f(t), t) \tag{2}$$

において、Taylor 展開

$$f(t+\Delta t) = f(t) + \Delta t \frac{df(t)}{dt} + \frac{\Delta t^2}{2!}\frac{d^2 f(t)}{dt^2} + \frac{\Delta t^3}{3!}\frac{d^3 f(t)}{dt^3} + \cdots \tag{3}$$

$$\simeq f(t) + \Delta t \frac{df(t)}{dt} \tag{4}$$

より

$$\frac{df(t)}{dt} = \frac{f(t+\Delta t) - f(t)}{\Delta t} \tag{5}$$

が得られる。これより

$$f(t+\Delta t) = f(t) + S(f(t), t)\Delta t \tag{6}$$

により次の時間ステップの数値解が求められる。

したがって、(1) については

$$f(t+\Delta t) = f(t) + (-\kappa f(t) + \alpha)\Delta t \tag{7}$$

を用いればよい。

1.1.2　Euler 陰解法

時間ステップ更新後に ODE が成立するとし (6) を書き換えて

$$f(t+\Delta t) = f(t) + S(f(t+\Delta t), t+\Delta t)\Delta t \tag{8}$$

により次の時間ステップの数値解が求められる。

(1) に対して適用すると

$$f(t+\Delta t) = f(t) + (-\kappa f(t+\Delta t) + \alpha)\Delta t \tag{9}$$

$$f(t+\Delta t) = \frac{f(t) + \alpha\Delta t}{1 + \kappa\Delta t} \tag{10}$$

1.1.3 Crank-Nicolson 法

時間ステップ更新前後の中間で ODE が成立するとし (6) を書き換えて

$$f(t+\Delta t) = f(t) + S\left(f\left(t+\frac{\Delta t}{2}\right), t+\frac{\Delta t}{2}\right)\Delta t \tag{11}$$

により次の時間ステップの数値解が求められる。

(1) に対して適用すると

$$f(t+\Delta t) = f(t) + \left(-\kappa f\left(t+\frac{\Delta t}{2}\right) + \alpha\right)\Delta t \tag{12}$$

$$= f(t) + \left(-\kappa\frac{f(t+\Delta t)+f(t)}{2} + \alpha\right)\Delta t \tag{13}$$

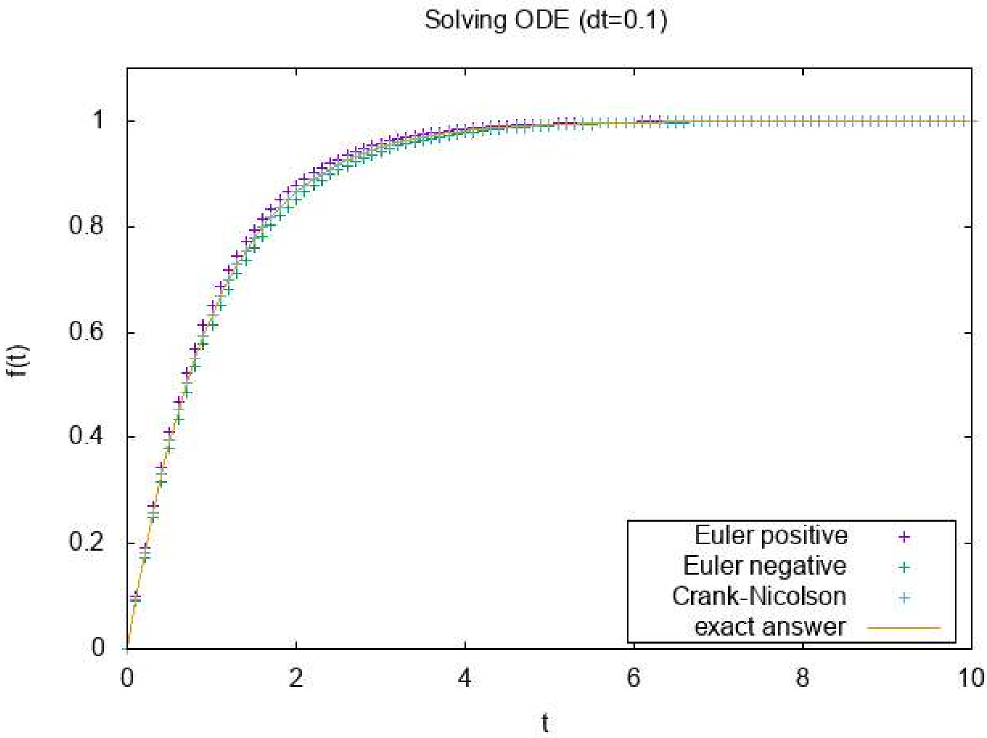
$$f(t+\Delta t) = \frac{(2-\kappa\Delta t)f(t) + 2\alpha\Delta t}{2+\kappa\Delta t} \tag{14}$$

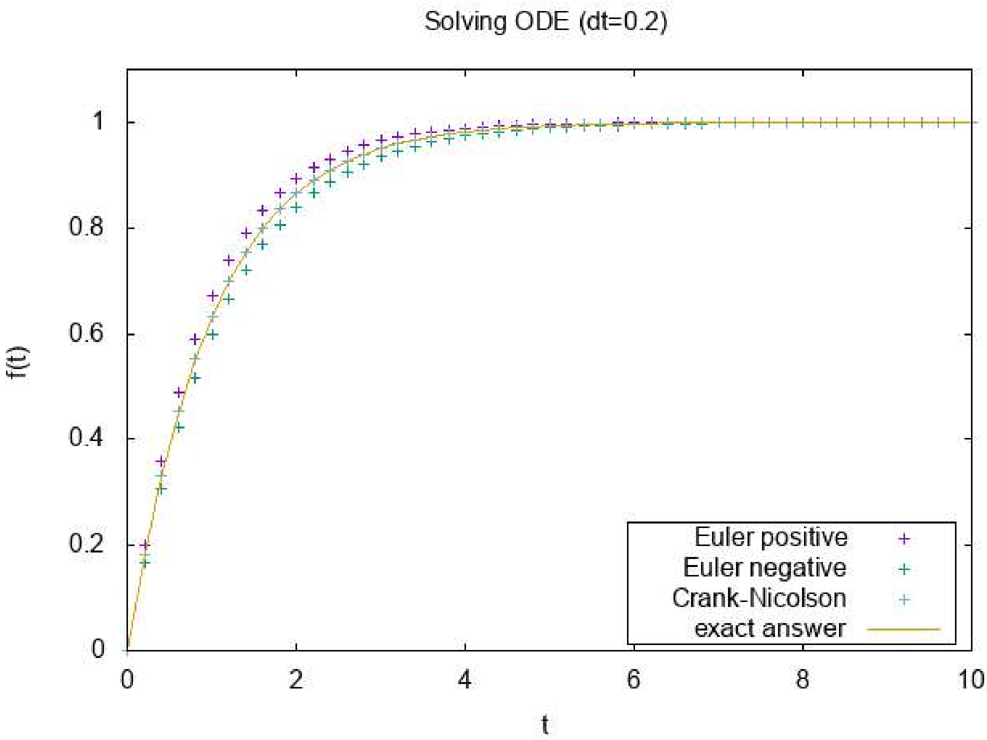
1.2 プログラム

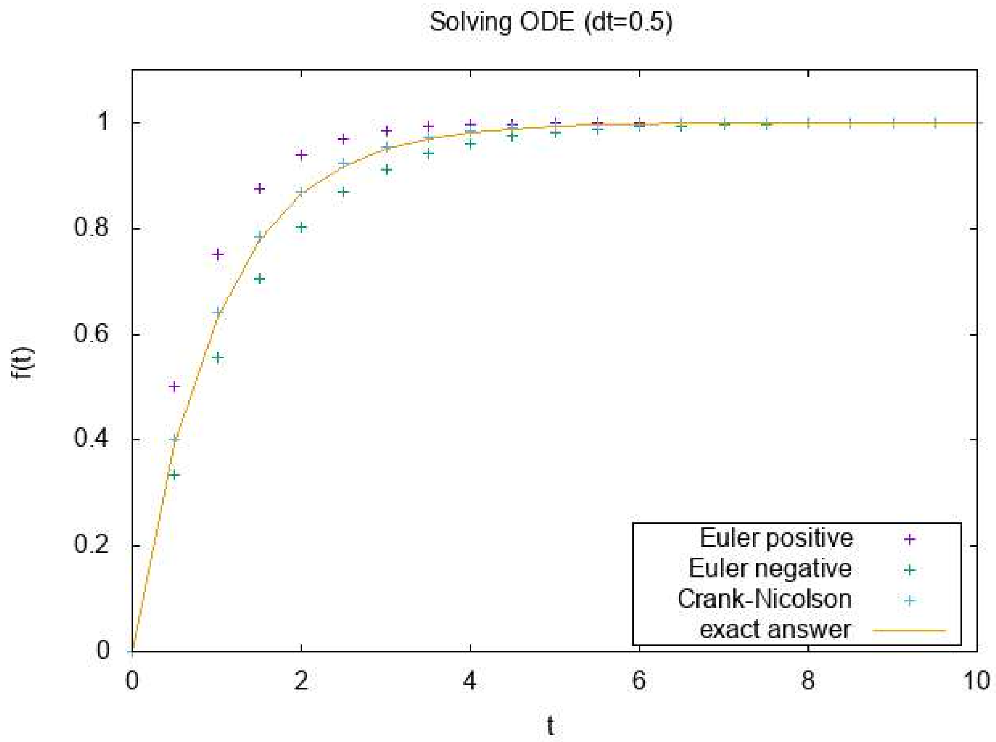
```fortran
program ODE

    implicit none
    integer :: n
    double precision :: time, dt
    double precision :: ffpositive,ffnegative,ffcrank
    double precision :: Rpositive,Rnegative,Rcrank
    double precision :: fftheory
    double precision :: alpha, kappa

    ! ***** 初期設定、定数の設定 *****
    time=0.d0
    alpha=1.d0
    kappa=1.d0

    ffpositive=0.d0
    ffnegative=0.d0
    ffcrank=0.d0
    Rpositive=0.d0
    Rnegative=0.d0
    Rcrank=0.d0
    fftheory=alpha/kappa*(1.-exp(-kappa*time))

    dt=5.d-1  ! 時間刻み幅の設定

    open(100, file='1211/result0.5.dat', status='replace')
    write(100,*) time,ffpositive,ffnegative,ffcrank,fftheory         ! 初期値の書き込み

    do n=1,100000

        ! ***** 次の時間ステップの計算 *****

        time=time+dt                                                 ! 現在時刻の更新
        ffpositive=ffpositive+alpha*dt-kappa*ffpositive*dt           ! Euler陽解法
        ffnegative=(ffnegative+alpha*dt)/(1+kappa*dt)                ! Euler陰解法
        ffcrank=(2*ffcrank+2*alpha*dt-kappa*ffcrank*dt)/(2+kappa*dt) ! 2次精度Crank-Nicolson
        fftheory=alpha/kappa*(1.-exp(-kappa*time))                   ! 解析解

        write(100,*) time,ffpositive,ffnegative,ffcrank,fftheory     ! 計算値の書き込み

        ! ***** 誤差の計算 *****
        if(n*dt==2.) then                                            ! t=2のみ抽出
            Rpositive=abs(ffpositive-fftheory)
            Rnegative=abs(ffnegative-fftheory)
            Rcrank=abs(ffcrank-fftheory)

            open(50, file='1211/R0.5.dat', status='replace')
            write(50,*) dt,Rpositive,Rnegative,Rcrank                ! 誤差の書き込み
            close(50)
        end if

        ! ***** ループを抜け出す *****
        if(time>=10.d0) exit                                         ! t=10で計算終了
    end do

    close(100)

end program ODE
```

1.3 結果

時間発展を $0 \leq t \leq 10$ の範囲で描画した。時間刻み幅 $\Delta t = 0.1, 0.2, 0.5$ の結果を示す。



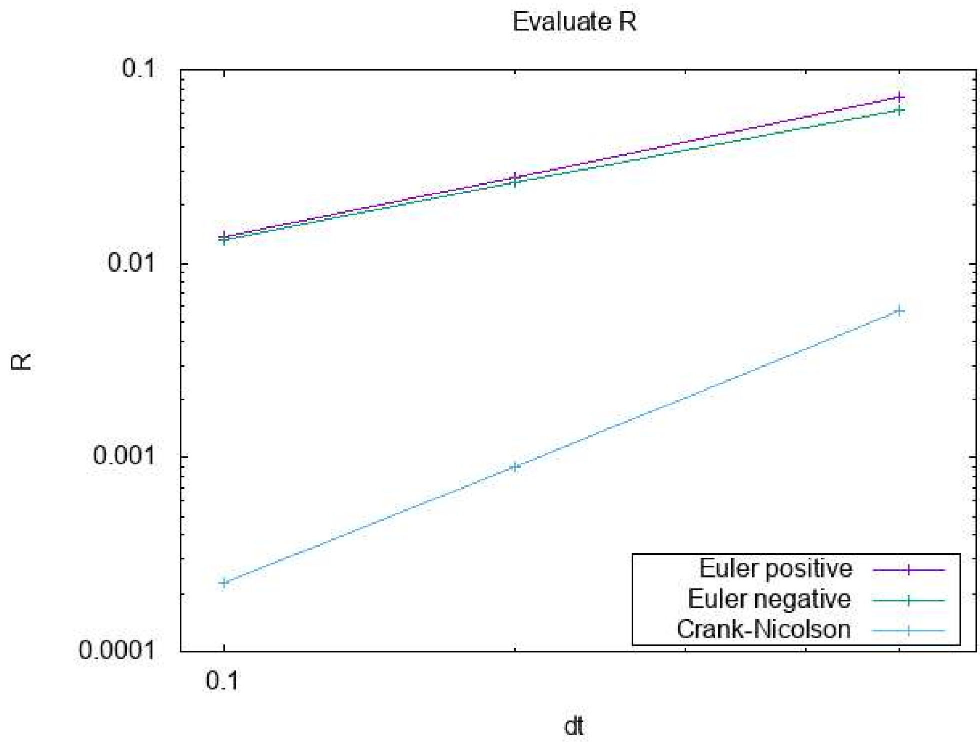




また、誤差を比較する。$t = 2$ での厳密解との差の絶対値をまとめた。

	Euler 陽解法	Euler 陰解法	Crank-Nicolson 法
$R_{0.1}$	1.376×10^{-2}	1.331×10^{-2}	2.257×10^{-4}
$R_{0.2}$	2.796×10^{-2}	2.617×10^{-2}	9.047×10^{-4}
$R_{0.5}$	7.284×10^{-2}	6.220×10^{-2}	5.735×10^{-3}
$R_{0.5}/R_{0.1}$	5.294	4.673	25.41

また、$\Delta t - R$ の関係を両対数グラフで示す。



1.4 考察

各手法における結果から、Euler 陽解法では正の誤差、陰解法では負の誤差、Crank-Nicolson 法では正の誤差が確認できた。

各手法における誤差の要因について考える。Taylor 展開について

$$f(t+\Delta t) = f(t) + \Delta t \frac{df(t)}{dt} + \frac{\Delta t^2}{2!}\frac{d^2 f(t)}{dt^2} + \frac{\Delta t^3}{3!}\frac{d^3 f(t)}{dt^3} + \cdots \tag{15}$$

$$\frac{df(t)}{dt} = \frac{f(t+\Delta t) - f(t)}{\Delta t} - \frac{\Delta t}{2!}\frac{d^2 f(t)}{dt^2} + o(\Delta t) \tag{16}$$

Euler 陽解法の式より

$$f(t+\Delta t) = f(t) + S(f(t), t)\Delta t \tag{17}$$

$$\frac{f(t+\Delta t) - f(t)}{\Delta t} = S(f(t), t) \tag{18}$$

したがって、

$$\frac{df(t)}{dt} = S(f(t), t) - \frac{\Delta t}{2!}\frac{d^2 f(t)}{dt^2} + o(\Delta t) \tag{19}$$

となり、$-\Delta t$ に比例した誤差が生じていることが確認できる。

同様に、陰解法については

$$\frac{df(t+\Delta t)}{dt} = S(f(t+\Delta t), t+\Delta t) + \frac{\Delta t}{2!}\frac{d^2 f(t+\Delta t)}{dt^2} + o(\Delta t) \tag{20}$$

となり、Δt に比例した誤差が生じていることが確認できる。

Crank-Nicolson 法については

$$\frac{d}{dt} f\left(t + \frac{\Delta t}{2}\right) = S\left(f\left(t + \frac{\Delta t}{2}\right), t + \frac{\Delta t}{2}\right) - \frac{\Delta t^2}{24}\frac{d^3}{dt^3} f\left(t + \frac{\Delta t}{2}\right) + o(\Delta t^2) \tag{21}$$

となり、$-\Delta t^2$ に比例した誤差が生じていることが確認できる。

3 つの手法の相対誤差を表した両対数グラフの傾きからも、Euler 法が 1 次精度で Crank-Nicolson 法が 2 次精度であることが確認できる。そのため、dt を小さくするほど Crank-Nicolson 法の精度が高まることが考えられる。[1]

[1]逆に時間ステップを大きくとる計算の場合は、Euler 法を用いた法がよい結果が得られる可能性がある。

2　拡散方程式

2.1　概要

気体や液体中での物質の拡散を表す式として、拡散方程式がある。今回は 1 次元拡散方程式について扱う。
1 次元拡散方程式

$$\frac{\partial}{\partial t}f(t,x) = \nu\frac{\partial^2}{\partial x^2}f(t,x) \tag{22}$$

について、時間微分を Euler 陽解法を用いて表せば

$$\frac{f(t+\Delta t,x)-f(t,x)}{\Delta t} = \nu\frac{\partial^2}{\partial x^2}f(t,x) \tag{23}$$

空間微分については、中心 2 階差分を用いる。Taylor 展開式より

$$f(t,x+\Delta x) = f(t,x) + \Delta x\left(\frac{\partial f(t,x)}{\partial x}\right) + \frac{\Delta x^2}{2!}\left(\frac{\partial^2 f(t,x)}{\partial x^2}\right) + \frac{\Delta x^3}{3!}\left(\frac{\partial^3 f(t,x)}{\partial x^3}\right) + \cdots \tag{24}$$

$$f(t,x-\Delta x) = f(t,x) - \Delta x\left(\frac{\partial f(t,x)}{\partial x}\right) + \frac{\Delta x^2}{2!}\left(\frac{\partial^2 f(t,x)}{\partial x^2}\right) - \frac{\Delta x^3}{3!}\left(\frac{\partial^3 f(t,x)}{\partial x^3}\right) + \cdots \tag{25}$$

$$\tag{26}$$

足し合わせて

$$\begin{aligned} f(t,x+\Delta x) + f(t,x-\Delta x) &= 2f(t,x) + 2\cdot\frac{\Delta x^2}{2!}\left(\frac{\partial^2 f(t,x)}{\partial x^2}\right) + \cdots && (27)\\ &\simeq 2f(t,x) + \Delta x^2\left(\frac{\partial^2 f(t,x)}{\partial x^2}\right) && (28)\end{aligned}$$

よって

$$\frac{\partial^2 f(t,x)}{\partial x^2} = \frac{f(t,x+\Delta x) - 2f(t,x) + f(t,x-\Delta x)}{\Delta x^2} \tag{29}$$

が導かれる。与式に代入して

$$\frac{f(t+\Delta t,x)-f(t,x)}{\Delta t} = \nu\frac{f(t,x+\Delta x) - 2f(t,x) + f(t,x-\Delta x)}{\Delta x^2} \tag{30}$$

より時刻 $t+\Delta t$ の値が求められる。
今回は 1 次元拡散方程式

$$\frac{\partial f}{\partial t} = \nu\frac{\partial^2 f}{\partial x^2} \tag{31}$$

を $\nu = 1$ で解く。
ただし、端部については隣の値が得られないため、別途条件を与える必要がある。例えば、Neumann 条件を与えた場合

$$\frac{\partial f}{\partial x} = 0 \tag{32}$$

ならば

$$\frac{f(t,x+\Delta x)-f(t,x)}{\Delta x} = 0 \tag{33}$$

より

$$f(t,x+\Delta x) = f(t,x) \tag{34}$$

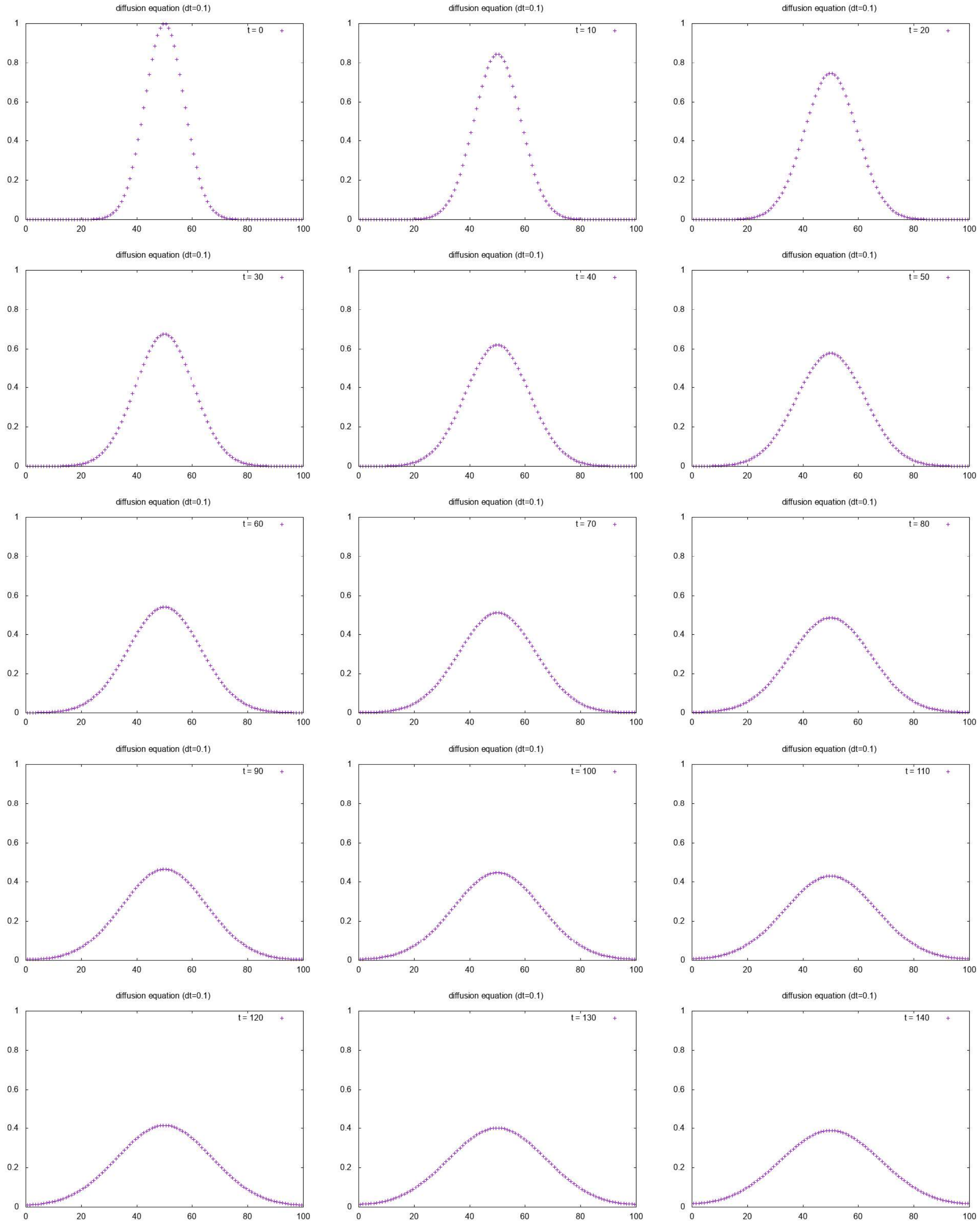
となる。

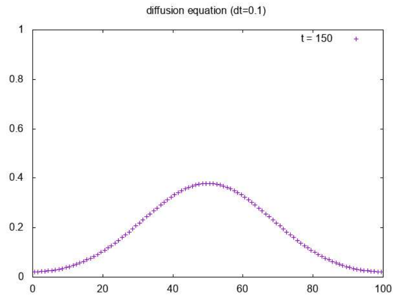
2.2 プログラム

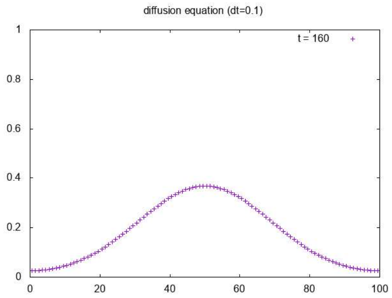
```fortran
program diffusion

    implicit None
    integer, parameter :: nx=100
    integer, parameter :: n_max=100000
    double precision :: x(nx)
    double precision :: f(nx)
    double precision :: fnew(nx)
    double precision :: dt
    double precision, parameter :: dx = 1.d0
    double precision, parameter :: neu = 1.d0
    double precision :: time
    integer :: i
    integer :: n
    character(1256) :: filename

    do i=1,nx
        x(i)=0.5*dx+(i-1)*dx                                           ! 計算に使用する格子点の設定
    end do


! ***** Case1 : dt = 0.1 *****

    dt = 10.d-2
    time = 0
    n = 0

    f(:)=exp(-(x(:)-5.d1)*(x(:)-5.d1)/(1.d1*1.d1))                     ! 初期条件(Gauss分布)

    open(UNIT=100,file='data/result0.1_000.dat',STATUS='REPLACE')
    do i=1,nx
        write(100,*) x(i),f(i)                                         ! 初期値の書き出し
    end do
    close(100)

    do n=1,n_max

        ! ***** 値の更新 *****

        fnew(1)=neu*dt/dx/dx*(f(2)-f(1))+f(1)                          ! 左端は境界条件を適用(Neumann条件)
        do i=2,nx-1
            fnew(i)=neu*dt/dx/dx*(f(i+1)-2*f(i)+f(i-1))+f(i)           ! 境界以外の値の更新
        end do
        fnew(nx)=neu*dt/dx/dx*(f(nx-1)-f(nx))+f(nx)                    ! 右端は境界条件を適用(Neumann条件)

        f(:)=fnew(:)                                                   ! 値を書き換える
        time = time + dt                                               ! 現在時刻の更新


        ! ***** 100時間ステップごとに書き出す *****

        if(mod(n,100)==0) then
            write(filename,'("data/result0.1_",i3.3,".dat")') n/100
            open(UNIT=100,file=filename,STATUS='replace')
            do i=1,nx
                write(100,*) x(i),f(i)
            end do
            close(100)
        end if


        ! ***** ループを抜け出す *****

        if(time>=300) exit                                             ! t=300で計算終了

    end do

end program diffusion
```

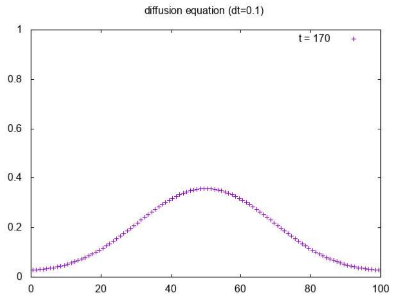
2.3 結果

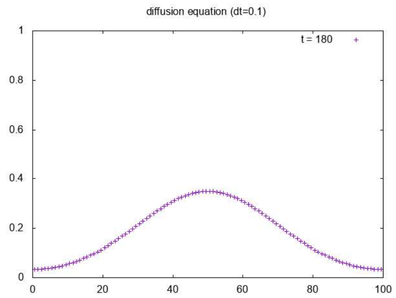
$\Delta t = 0.1, 0.5, 0.51$ の 3 つの場合について、拡散の様子をそれぞれ 100 時間ステップごとに描画した。

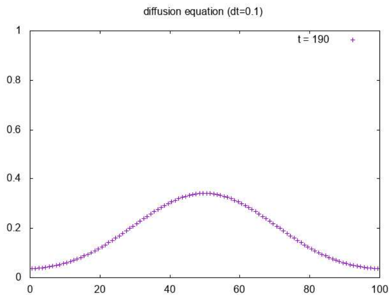


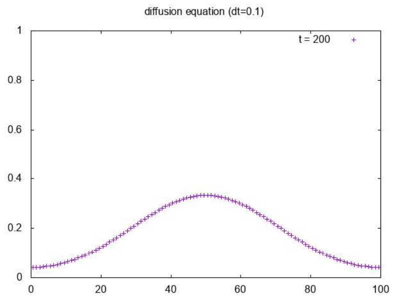

diffusion equation (dt=0.1)
t = 150

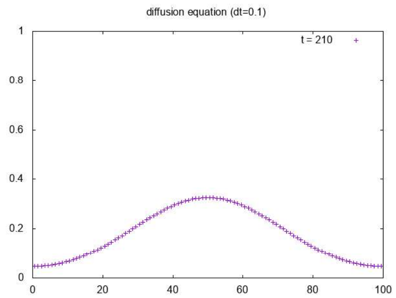

diffusion equation (dt=0.1)
t = 160

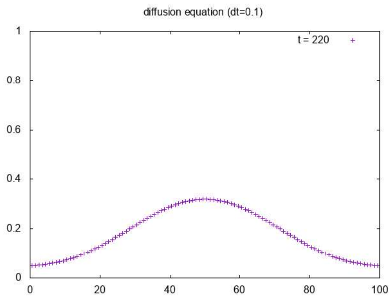

diffusion equation (dt=0.1)
t = 170

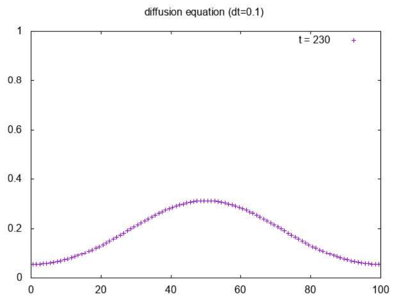

diffusion equation (dt=0.1)
t = 180

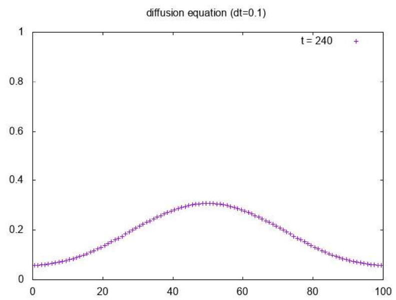

diffusion equation (dt=0.1)
t = 190

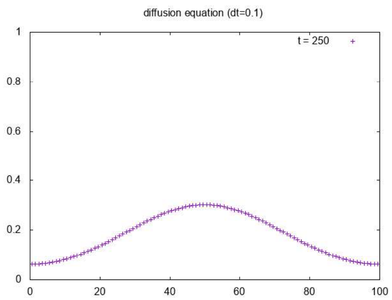

diffusion equation (dt=0.1)
t = 200

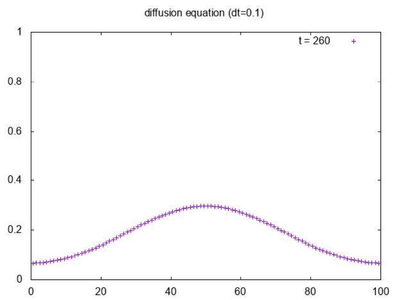

diffusion equation (dt=0.1)
t = 210

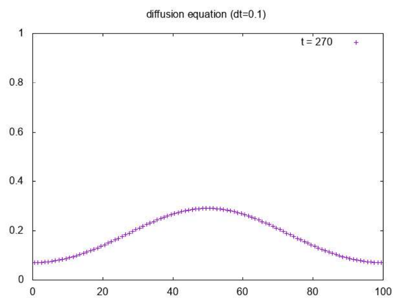

diffusion equation (dt=0.1)
t = 220

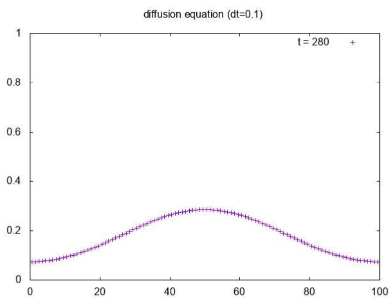

diffusion equation (dt=0.1)
t = 230

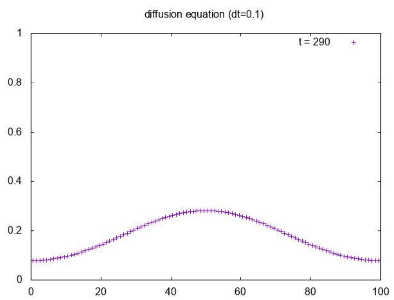

diffusion equation (dt=0.1)
t = 240

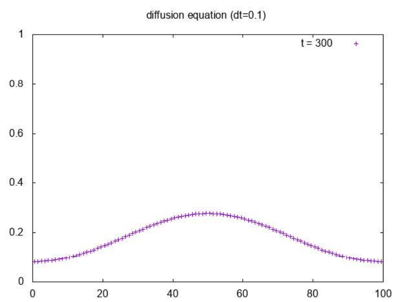

diffusion equation (dt=0.1)
t = 250

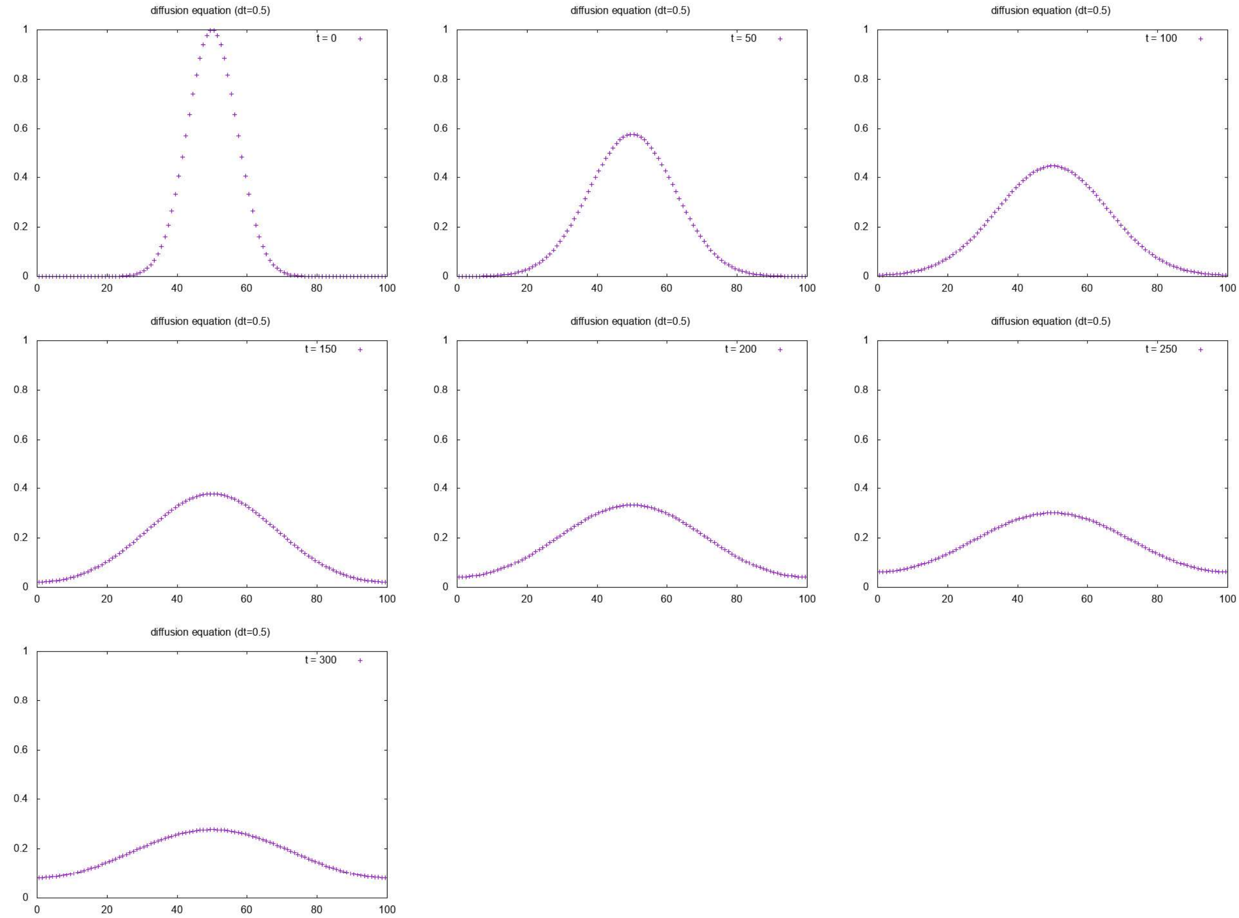

diffusion equation (dt=0.1)
t = 260

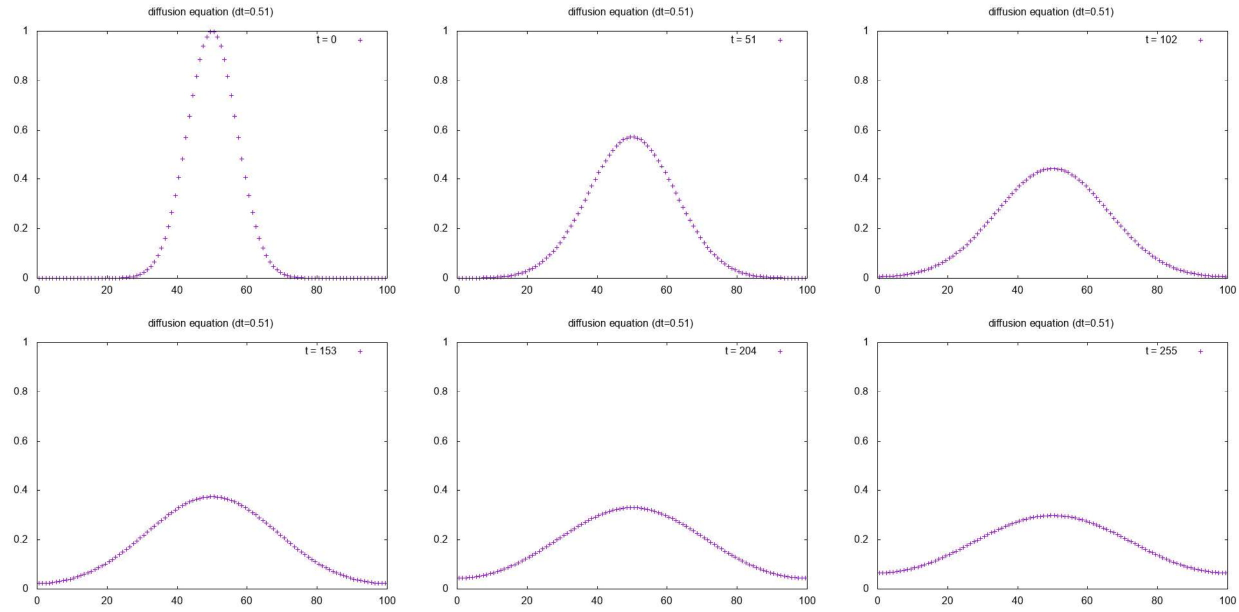

diffusion equation (dt=0.1)
t = 270


diffusion equation (dt=0.1)
t = 280


diffusion equation (dt=0.1)
t = 290


diffusion equation (dt=0.1)
t = 300

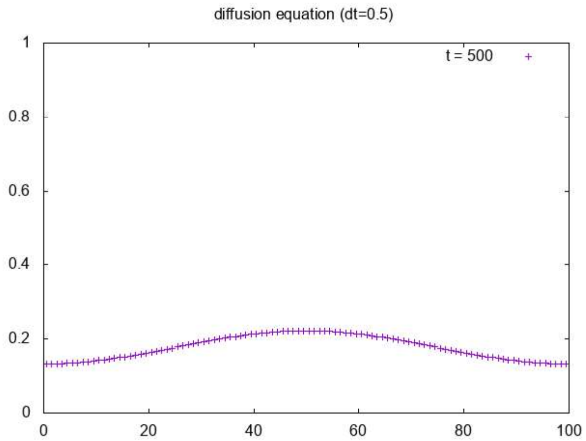


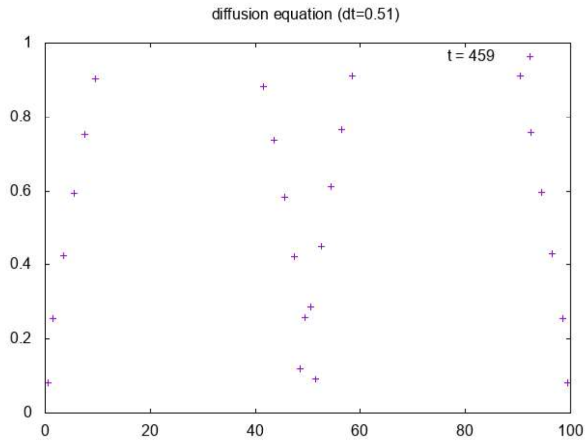


2.4　考察

時間経過にしたがって、物質が拡散して物質が均一に広がっていく様子が読み取れた。Δt が大きい場合でも十分計算出来ていることがわかる。

ただし、$t > 300$ 以降も計算を続行した場合、$\Delta t = 0.5, 0.51$ で大きな差が生じた。





$\Delta t = 0.51$ の場合、

$$\frac{\Delta x^2}{2\nu} = 0.5 < \Delta t \tag{35}$$

となり、数値振動が生じることが説明できる。

3　移流方程式

3.1　概要

粒子の輸送は偏微分方程式の形で表すことができる。物質が速度 u で移動する様子を表す 1 次元の移流方程式

$$\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x} = 0 \tag{36}$$

について解くことを考える。

3.1.1　風上 1 次差分法

時間変動、空間変動について Euler 陽解法を用いると

$$\frac{f(t+\Delta t, x) - f(t, x)}{\Delta t} + u\frac{f(t, x+\Delta x) - f(t, x)}{\Delta x} = 0 \tag{37}$$

ここで、$u > 0$ においてクーラン数 C

$$C = \left|\frac{u\Delta t}{\Delta x}\right| \tag{38}$$

を用いて整理すると

$$f(t+\Delta t, x) = (1+C)f(t, x) - Cf(x+\Delta x) \tag{39}$$

である。ただし、この場合、$f(x+\Delta x)$ 項が負のため、値によっては計算が不安定になるおそれがある。

そこで、空間変動について、後退差分を用いる。

$$\frac{\partial f}{\partial x} = \frac{f(t, x) - f(t, x-\Delta x)}{\Delta x} \tag{40}$$

より

$$\frac{f(t+\Delta t, x) - f(t, x)}{\Delta t} + u\frac{f(t, x) - f(t, x-\Delta x)}{\Delta x} = 0 \tag{41}$$

整理すると

$$f(t+\Delta t, x) = (1-C)f(t, x) + Cf(x-\Delta x) \tag{42}$$

$$= f(t, x) - C(f(t, x) - f(x-\Delta x)) \tag{43}$$

] となる。計算に風上側の値を用いるために風上差分とよばれる。計算に $f(t, x-\Delta x)$ を用いるため、風上側の端部では境界条件を定める必要がある。

3.1.2　Leith 法

速度 u で物質が輸送されるとき、時間 Δt の間に物質は $u\Delta t$ だけ進む。つまり

$$f(t+\Delta t, x) = f(t, x-u\Delta t) \tag{44}$$

ここでは、時刻 t における位置 $x - u\Delta t$ の値を補間により得ることを考える。

補間には様々な手法が提案されているが、ここでは 2 次多項式による補間を考える。時刻 t での状態が関数

$$f(t, X) = a(X-x)^2 + b(X-x) + c \tag{45}$$

で表されるとする。ここで、$f(t,x+\Delta x), f(t,x), f(t,x-\Delta x)$ の値は既知であるから、$X=x+\Delta x, x, x-\Delta x$ を代入して

$$f(t,x+\Delta x) = a(x+\Delta x-x)^2 + b(x+\Delta x-x) + c = a\Delta x^2 + b\Delta x + c \tag{46}$$

$$f(t,x) = a(x-x)^2 + b(x-x) + c = c \tag{47}$$

$$f(t,x-\Delta x) = a(x-\Delta x-x)^2 + b(x-\Delta x-x) + c = a\Delta x^2 - b\Delta x + c \tag{48}$$

$$\tag{49}$$

以上より、

$$a = \frac{1}{2\Delta x^2}(f(t,x+\Delta x) - 2f(t,x) + f(t,x-\Delta x)) \tag{50}$$

$$b = \frac{1}{2\Delta x}(f(t,x+\Delta x) - f(t,x-\Delta x)) \tag{51}$$

$$c = f(t,x) \tag{52}$$

より定数 a,b,c が定まる。

次の時間ステップの値 $f(t+\Delta t, x)$ は

$$f(t+\Delta t,x) = f(t,x-u\Delta t) = a(x-u\Delta t-x)^2 + b(x-u\Delta t-x) + c = au\Delta t^2 - bu\Delta t + c \tag{53}$$

により求められる。計算に $f(x+\Delta x), f(x-\Delta x)$ を用いるため、両端で境界条件を定める必要がある。

3.1.3 CIP 法

Leith 法を拡張し、3 次多項式で保管する方法として Cubic-Spline 法がある。ここでは、格子点以外の条件により係数決定をする方法として、勾配

$$g(t,x) = \frac{\partial}{\partial x} f(t,x) \tag{54}$$

に着目する。Cubic-Spline 法では勾配が滑らかになるように条件式を立てる。

Cubic-Spline 法を改良した手法として CIP 法がある。u が一定 $(\partial u/\partial x = 0)$ として (36) を x で偏微分して

$$\frac{\partial}{\partial x}\left(\frac{\partial f}{\partial t} + u\frac{\partial f}{\partial x}\right) = 0 \tag{55}$$

$$\frac{\partial}{\partial t}\frac{\partial f}{\partial x} + u\frac{\partial}{\partial x}\frac{\partial f}{\partial x} = 0 \tag{56}$$

$$\frac{\partial g}{\partial t} + u\frac{\partial g}{\partial x} = 0 \tag{57}$$

より、同様に $g(t,x)$ についても移流方程式の形が得られた。したがって、$g(t,x)$ についても同様に時間発展を計算すれば、必要な既知の格子点数を減らすことができる。

補間関数を考える。時刻 t での状態が関数

$$f(t,X) = a(X-x)^3 + b(X-x)^2 + c(X-x) + d \tag{58}$$

で表されるとする。ここで、$f(t,x), f(t,x-\Delta x)$ の値は既知であるから、$X=x, x-\Delta x$ を代入して

$$f(t,x) = a(x-x)^3 + b(x-x)^2 + c(x-x) + d = d \tag{59}$$

$$f(t,x-\Delta x) = a(x-\Delta x-x)^3 + b(x-\Delta x-x)^2 + c(x-\Delta x-x) + d = -a\Delta x^3 + b\Delta x^2 - c\Delta x + d \tag{60}$$

ここで

$$g(t,x) = 3a(X-x)^2 + 2b(X-x) + c \tag{61}$$

について $g(t,x), g(t,x-\Delta x)$ の値は既知であるから、$X = x, x-\Delta x$ を代入して

$$g(t,x) = 3a(x-x)^2 + 2b(x-x) + c = c \tag{62}$$

$$g(t,x-\Delta x) = 3a(x-\Delta x-x)^2 + 2b(x-\Delta x-x) + c = 3a\Delta x^2 - 2b\Delta x + c \tag{63}$$

以上より、

$$a = -\frac{2}{\Delta x^3}(f(t,x) - f(t,x-\Delta x)) + \frac{1}{\Delta x^2}(g(t,x) + g(t,x-\Delta x)) \tag{64}$$

$$b = -\frac{3}{\Delta x^2}(f(t,x) - f(t,x-\Delta x)) + \frac{1}{\Delta x}(2g(t,x) + g(t,x-\Delta x))) \tag{65}$$

$$c = g(t,x) \tag{66}$$

$$d = f(t,x) \tag{67}$$

より定数 a, b, c, d が定まる。計算には $f(x-\Delta x)$ のみ用いるため、風上側の端部でのみ境界条件を定めればよい。

3.2　プログラム

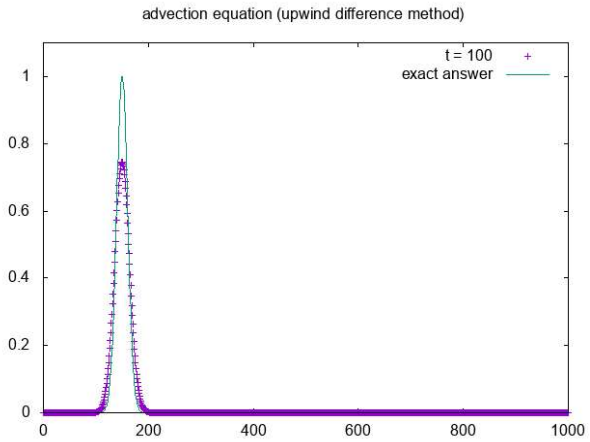
```fortran
program ADVECTION_case1

    implicit None
    integer, parameter :: nx = 1000
    double precision :: x(nx), f(nx), fnew(nx)
    double precision, parameter :: dt = 1.d-1
    double precision, parameter :: dx = 1.d0
    double precision, parameter :: u =1.d0
    double precision :: time
    double precision :: ftheory
    integer, parameter :: n_max = 7000
    integer :: i, n
    character(1256) :: filename

    do i=1,nx
        x(i)=0.5*dx+(i-1)*dx                                                    ! 計算に使用する格子点の設定
    end do


! ***** Case1 : 風上1次差分法 *****

    time=0.d0
    f(:)=exp(-(x(:)-5.d1)*(x(:)-5.d1)/(1.5d1*1.5d1))                            ! 初期条件(Gauss分布)

    open(UNIT=100,FILE='data/result_upwind_difference_000.dat',STATUS='REPLACE')
    do i=1,nx
        write(100,*) x(i),f(i)                                                  ! 初期値の書き出し
    end do
    close(100)

    do n=1,n_max

        ! ***** 値の更新 *****

        ! --- 左端(i=1)は境界条件を適用(Neumann条件) ---
        fnew(1)=f(1)

        ! --- 境界以外の値の更新 ---
        do i=2,nx
            fnew(i)=f(i)-u*dt/dx*(f(i)-f(i-1))
        end do

        f(:)=fnew(:)                                                            ! 値を書き換える
        time = time + dt                                                        ! 現在時刻の更新


        ! ***** 1000時間ステップごと(100sごと)に書き出す *****

        if(mod(n,1000)==0) then
            write(filename,'("data/result_upwind_difference_",i3.3,".dat")') n/1000
            open(UNIT=100,FILE=filename,STATUS='replace')
            do i=1,nx
                ftheory=exp(-(x(i)-u*time-5.d1)*(x(i)-u*time-5.d1)/(1.5d1*1.5d1)) ! 解析解の計算
                write(100,*) x(i),f(i),ftheory                                  ! 計算結果と解析解の書き出し
            end do
            close(100)
        end if
    end do

end program ADVECTION_case1
```

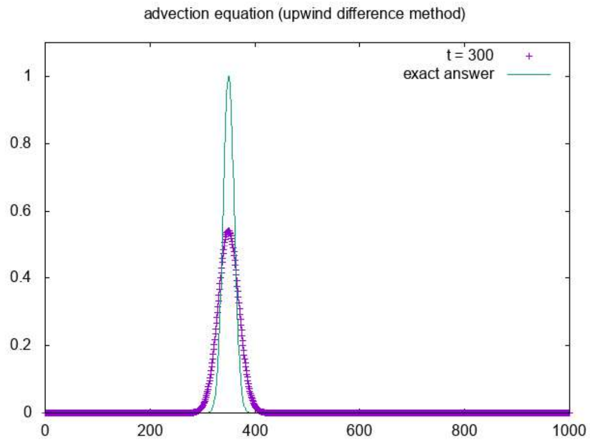
```fortran
program ADVECTION_case2

    implicit None
    integer, parameter :: nx = 1000
    double precision :: x(nx), f(nx), fnew(nx)
    double precision, parameter :: dt = 1.d-1
    double precision, parameter :: dx = 1.d0
    double precision, parameter :: u =1.d0
    double precision :: time
    double precision :: ftheory
    double precision :: a, b, xp
    integer, parameter :: n_max = 7000
    integer :: i, n
    character(1256) :: filename

    do i=1,nx
        x(i)=0.5*dx+(i-1)*dx                                              ! 計算に使用する格子点の設定
    end do


! ***** Case2 : Leith法 *****

    time=0.d0
    f(:)=exp(-(x(:)-5.d1)*(x(:)-5.d1)/(1.5d1*1.5d1))                      ! 初期条件(Gauss分布)

    open(UNIT=100,FILE='data/result_Leith_000.dat',STATUS='REPLACE')
    do i=1,nx
        write(100,*) x(i),f(i)                                            ! 初期値の書き出し
    end do
    close(100)

    do n=1,n_max

        ! ***** 値の更新 *****

        ! --- 左端(i=1)は境界条件を適用(Neumann条件) ---
        a=(-f(1)+f(2))/(2.d0*dx*dx)
        b=(f(2)-f(1))/(2.d0*dx)
        xp=-u*dt
        fnew(1)=a*xp*xp+b*xp+f(1)

        ! --- 境界以外の値の更新 ---
        do i=2,nx-1
            a=(f(i+1)-2.d0*f(i)+f(i-1))/(2.d0*dx*dx)
            b=(f(i+1)-f(i-1))/(2.d0*dx)
            fnew(i)=a*xp*xp+b*xp+f(i)
        end do

        ! --- 右端(i=1000)は境界条件を適用(Neumann条件) ---
        a=(-f(1000)+f(999))/(2.d0*dx*dx)
        b=(f(1000)-f(999))/(2.d0*dx)
        fnew(1000)=a*xp*xp+b*xp+f(1000)


        f(:)=fnew(:)                                                      ! 値を書き換える
        time = time + dt                                                  ! 現在時刻の更新


        ! ***** 1000時間ステップごと(100sごと)に書き出す *****

        if(mod(n,1000)==0) then
            write(filename,'("data/result_Leith_",i3.3,".dat")') n/1000
            open(UNIT=100,FILE=filename,STATUS='replace')
            do i=1,nx
                ftheory=exp(-(x(i)-u*time-5.d1)*(x(i)-u*time-5.d1)/(1.5d1*1.5d1)) ! 解析解の計算
                WRITE(100,*) x(i),f(i),ftheory                            ! 計算結果と解析解の書き出し
            end do
            close(100)
        end if
    end do

end program ADVECTION_case2
```

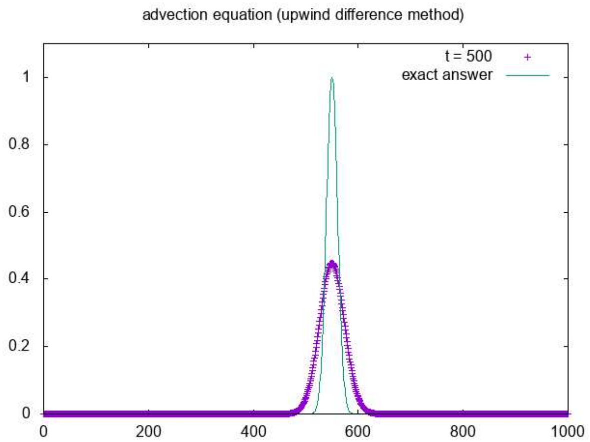
```fortran
program ADVECTION_case3

    implicit None
    integer, parameter :: nx = 1000
    double precision :: x(nx), f(nx), fnew(nx), g(nx), gnew(nx)
    double precision, parameter :: dt = 1.d-1
    double precision, parameter :: dx = 1.d0
    double precision, parameter :: u =1.d0
    double precision :: time
    double precision :: ftheory
    double precision :: a, b, xp
    integer, parameter :: n_max = 7000
    integer :: i, n
    character(1256) :: filename

    do i=1,nx
        x(i)=0.5*dx+(i-1)*dx                                                    ! 計算に使用する格子点の設定
    end do


! ***** Case3 : CIP法 *****

    time=0.d0
    f(:)=exp(-(x(:)-5.d1)*(x(:)-5.d1)/(1.5d1*1.5d1))                            ! 初期条件(Gauss分布)
    g(:)=-2.d0*(x(:)-5.d1)/(1.5d1*1.5d1)*exp(-(x(:)-5.d1)*(x(:)-5.d1)/(1.5d1*1.5d1)) ! 初期条件(Gauss分布の微分)

    open(UNIT=100,FILE='data/result_CIP_000.dat',STATUS='REPLACE')
    do i=1,nx
        write(100,*) x(i),f(i)                                                  ! 初期値の書き出し
    end do
    close(100)

    do n=1,n_max

        ! ***** 値の更新 *****

        ! --- 左端(i=1)は境界条件を適用(Neumann条件) ---

        fnew(1)=f(1)
        gnew(1)=0

        ! --- 境界以外の値の更新 ---

        do i=2,nx
            a=2.d0/dx/dx/dx*(f(i-1)-f(i))+(g(i)+g(i-1))/dx/dx
            b=-3.d0/dx/dx*(f(i)-f(i-1))+(2.d0*g(i)+g(i-1))/dx
            xp=-u*dt
            fnew(i)=a*xp*xp*xp+b*xp*xp+g(i)*xp+f(i)
            gnew(i)=3.d0*a*xp*xp+2*b*xp+g(i)
        end do


        f(:)=fnew(:)                                                            ! fの値を書き換える
        g(:)=gnew(:)                                                            ! gの値を書き換える
        time = time + dt                                                        ! 現在時刻の更新


        ! ***** 1000時間ステップごと(100sごと)に書き出す *****

        if(mod(n,1000)==0) then
            write(filename,'("data/result_CIP_",i3.3,".dat")') n/1000
            open(UNIT=100,FILE=filename,STATUS='replace')
            do i=1,nx
                ftheory=exp(-(x(i)-u*time-5.d1)*(x(i)-u*time-5.d1)/(1.5d1*1.5d1))    ! 解析解の計算
                write(100,*) x(i),f(i),ftheory                                  ! 計算結果と解析解の書き出し
            end do
            close(100)
        end if
    end do

end program ADVECTION_case3
```

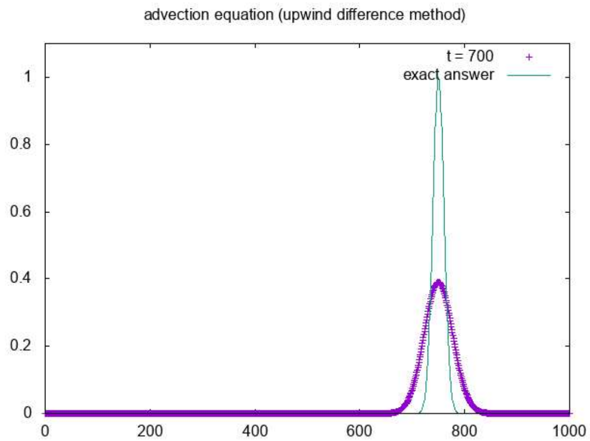
3.3 結果

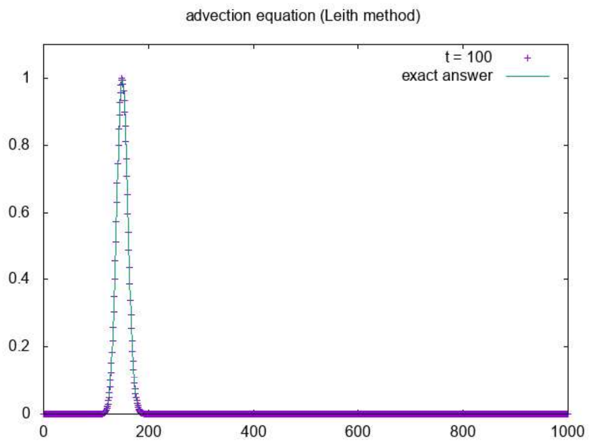
case1:風上 1 次差分法、case2:Leith 法、case3:CIP 法についてそれぞれ移流方程式を解いた結果と解析解を比較する。

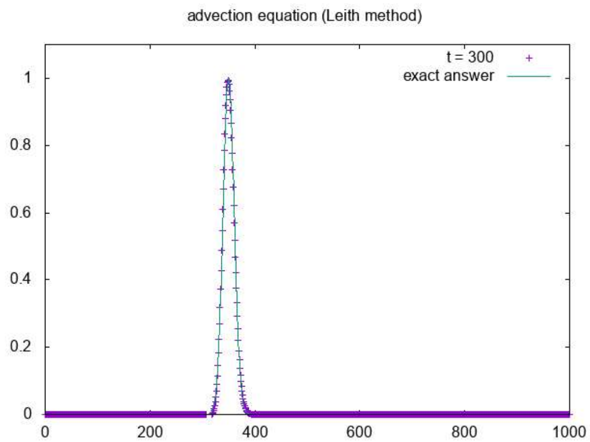


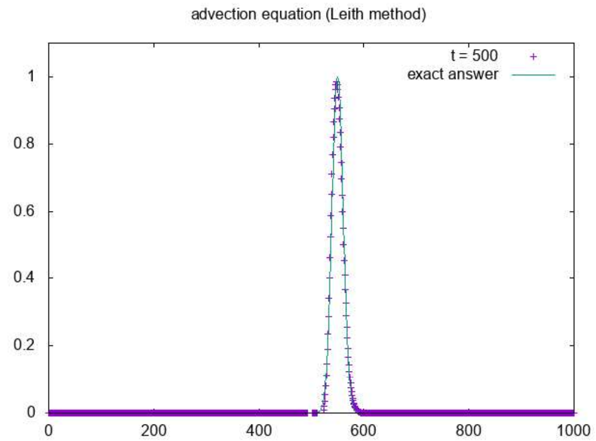


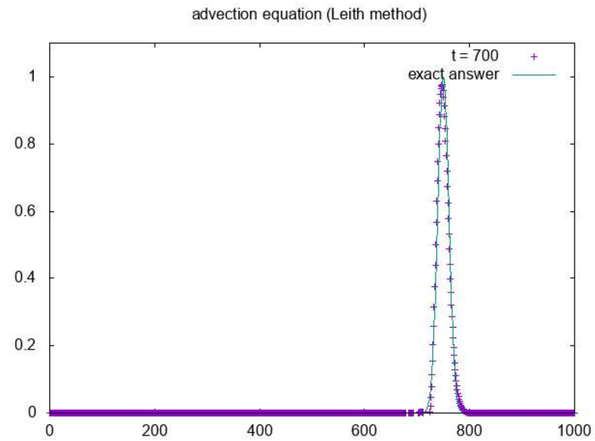


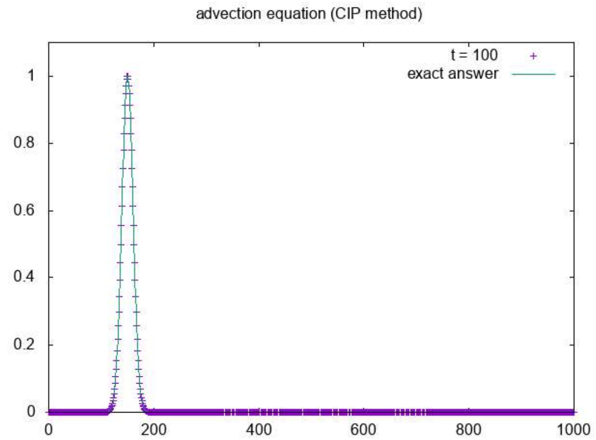


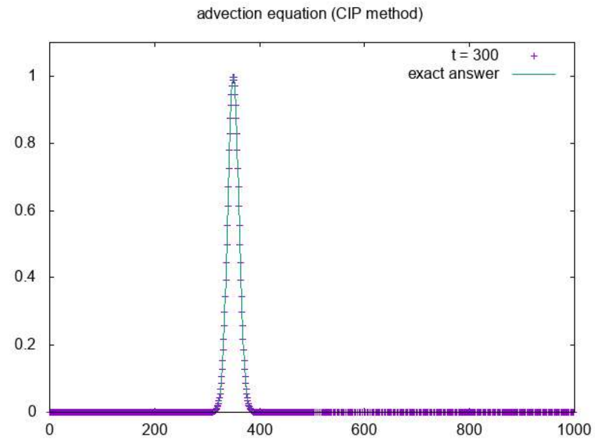


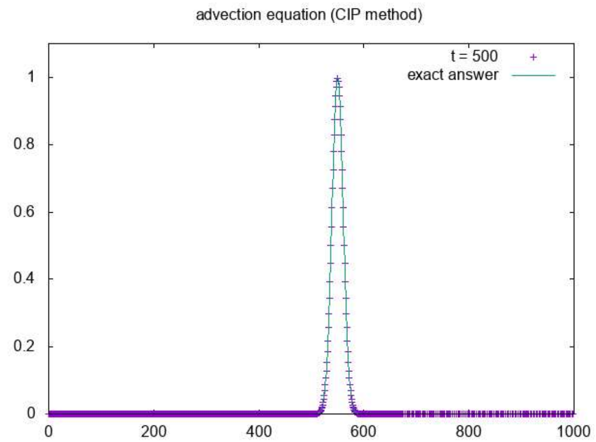


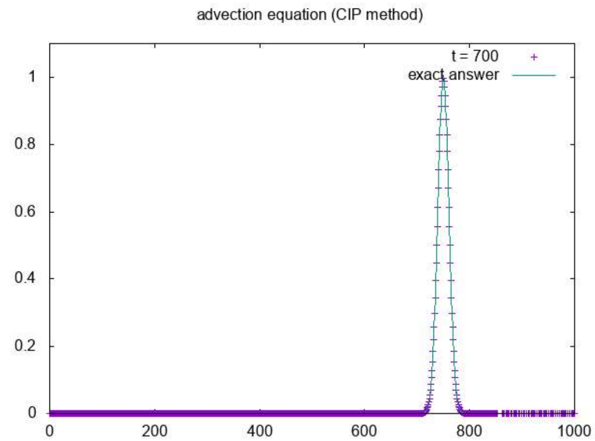












3.4 考察

3 つの手法を用いて計算した。Leith 法と CIP 法については、解析解とほぼ一致した結果が得られた。しかし、差分法では物質が徐々に減少していく結果となった。$0 < C < 1$ により計算は安定しているが、内推補間を行っているため、$t + \Delta t$ での位置 x の数値は時刻 t の $x - u\Delta t$ 値と一致するかもしくは減少する。微小な減少が蓄積してこのように徐々に極値が減少する推移となったと考えられる。