差分スキーム

物理・化学・生物現象には微分方程式でモデル化される例が多い。モデルを使って現実の現象をコンピュータ上で再現することをシミュレーション（数値シミュレーション、コンピュータシミュレーション）と呼ぶ。そのためには、微分方程式をコンピュータ上で計算できる数値スキームで近似することが必要になる。その一つの方法が微分方程式を差分方程式におき直すことである。

微分方程式の差分化

次の１次元境界値問題を考える。

$$-\frac{d^2}{dx^2}u(x)+x\cdot u(x)=(1+2x-x^2)e^x,\qquad x\in(0,1)$$

$$u(0)=1,\ u(1)=0$$

格子上の関数

区間(0,1)を分割して、各格子点上での関数値を設定する。区間(0,1)をn分割すれば、格子点数はn+1で、

格子間隔 $h=\frac{1}{n}$

格子点の位置 $x_i=ih,\ i=0,1,\cdots n$

格子点 x_i でのuの値 $u(x_i)=u_i$

格子点での関数値 $1+2x_i-x_i^2,\exp(x_i)$

を定義できる。また、次の値を定義する。

$$u(x_i+h)=u_{i+1}$$

$$u(x_i-h)=u_{i-1}$$

１次微分u'(x), ２次微分u''(x)の差分を導出する。

Taylor展開による方法

1次微分u'(x)の差分

Mapleのtaylorコマンドを使ってu(x)をTaylor展開する。

```
> ua:=taylor(u(x+h),h,2);
```

$$ua:=u(x)+D(u)(x)\,h+O(h^2) \tag{1}$$

```
> ub:=taylor(u(x-h),h,2);
```

$$ub:=u(x)-D(u)(x)\,h+O(h^2) \tag{2}$$

前進差分

u(x+h)のTaylor展開より、

$$u'(x)\approx\frac{u(x+h)-u(x)}{h}$$

後退差分

u(x-h)のTaylor展開より、

$$u'(x) \approx \frac{u(x) - u(x-h)}{h}$$

```
> uaa:=taylor(u(x+h),h,3);
```

$$uaa := u(x) + D(u)(x)\,h + \frac{1}{2}\,D^{(2)}(u)(x)\,h^2 + O(h^3) \tag{3}$$

```
> ubb:=taylor(u(x-h),h,3);
```

$$ubb := u(x) - D(u)(x)\,h + \frac{1}{2}\,D^{(2)}(u)(x)\,h^2 + O(h^3) \tag{4}$$

中心差分

uaa-ubbを計算して、

$$u'(x) \approx \frac{u(x+h) - u(x-h)}{2\,h}$$

2次微分u''(x)の差分

uaa+ubbを計算して、2次微分u''(x)の差分を求めることができる。

$$u''(x) \approx \frac{u(x+h) + u(x-h) - 2\cdot u(x)}{h^2}$$

差分スキームの構成

例11.1の差分スキームを求める。

$$-\frac{u_{i+1} + u_{i-1} - 2\cdot u_i}{h^2} + x_i \cdot u_i = \left(1 + 2\,x_i - x_i^2\right) e^{x_i},\ \text{i=1,\$\$\$n-1}$$

$$u_0 = 1,\ u_n = 0$$

これを行列で表現すると、(n-1)×(n-1)行列として、

$$A = \frac{1}{h^2}\begin{bmatrix} 2 + h^2 x_1 & -1 & 0 & 0 & 0 \\ -1 & 2 + h^2 x_2 & -1 & 0 & 0 \\ 0 & \cdot & \cdot & \cdot & 0 \\ 0 & 0 & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & -1 & 2 + h^2 x_{n-1} \end{bmatrix}$$

$$b = \begin{bmatrix} \left(1 + 2\,x_1 - x_1^2\right) e^{x_1} + \frac{1}{h^2} \\ \left(1 + 2\,x_2 - x_2^2\right) e^{x_2} \\ \cdot \\ \cdot \\ \left(1 + 2\,x_{n-1} - x_{n-1}^2\right) e^{x_{n-1}} \end{bmatrix}$$

与えられたnに対して連立１次方程式Au=bを解くことによって、境界値問題の近似解として差分解を求めることができる。

差分スキームのプログラミング

ステップ１　格子点の生成

例えば、区間(0,1)をn=6分割する。

```
> n:=7;
```

$$n := 7 \tag{5}$$

```
> gridpoints:=n;
```

$$gridpoints := 7 \tag{6}$$

```
> h:=evalf(1/gridpoints,7);
```

$$h := 0.1428571 \tag{7}$$

```
> x:=Vector(1..n+1):
> for i from 1 to n+1 do:
  x[i]:=h*(i-1);
  end do;
```

$$x_1 := 0.$$
$$x_2 := 0.1428571$$
$$x_3 := 0.2857142$$
$$x_4 := 0.4285713$$
$$x_5 := 0.5714284$$
$$x_6 := 0.7142855$$
$$x_7 := 0.8571426$$
$$x_8 := 0.9999997 \tag{8}$$

ステップ２　係数行列の設定

```
> A:=Matrix(1..n-1,1..n-1):
> for i from 1 to n-2 do:
    A[i,i+1]:=-1.0/(h*h):
    A[i+1,i]:=-1.0/(h*h):
  end do:
> for i from 1 to n-1 do:
    A[i,i]:=2.0/(h*h)+x[i+1]:
  end do:
> A;
```

$$\begin{bmatrix} 98.14291590 & -49.00002940 & 0 & 0 & 0 & 0 \\ -49.00002940 & 98.28577300 & -49.00002940 & 0 & 0 & 0 \\ 0 & -49.00002940 & 98.42863010 & -49.00002940 & 0 & 0 \\ 0 & 0 & -49.00002940 & 98.57148720 & -49.00002940 & 0 \\ 0 & 0 & 0 & -49.00002940 & 98.71434430 & -49.00002940 \\ 0 & 0 & 0 & 0 & -49.00002940 & 98.85720140 \end{bmatrix} \tag{9}$$

ステップ３　ベクトルbの設定

```
> b:=Vector(1..n-1);
```

$$b := \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{10}$$

```
> for i from 1 to n-1 do:
    b[i]:=(1.0+2.0*x[i+1]-x[i+1]^2)*exp(x[i+1]):
  end do;
```

$$b_1 := 1.459612703$$
$$b_2 := 1.982489267$$
$$b_3 := 2.568880399$$
$$b_4 := 3.216341042$$
$$b_5 := 3.918699820$$
$$b_6 := 4.664745338 \tag{11}$$

ステップ４　境界条件の設定

```
> b[1]:=b[1]+1.0/(h*h);
```

$$b_1 := 50.45964210 \tag{12}$$

ステップ５　行列の解法

```
> with(LinearAlgebra):
> u:=LinearSolve(A,b);
```

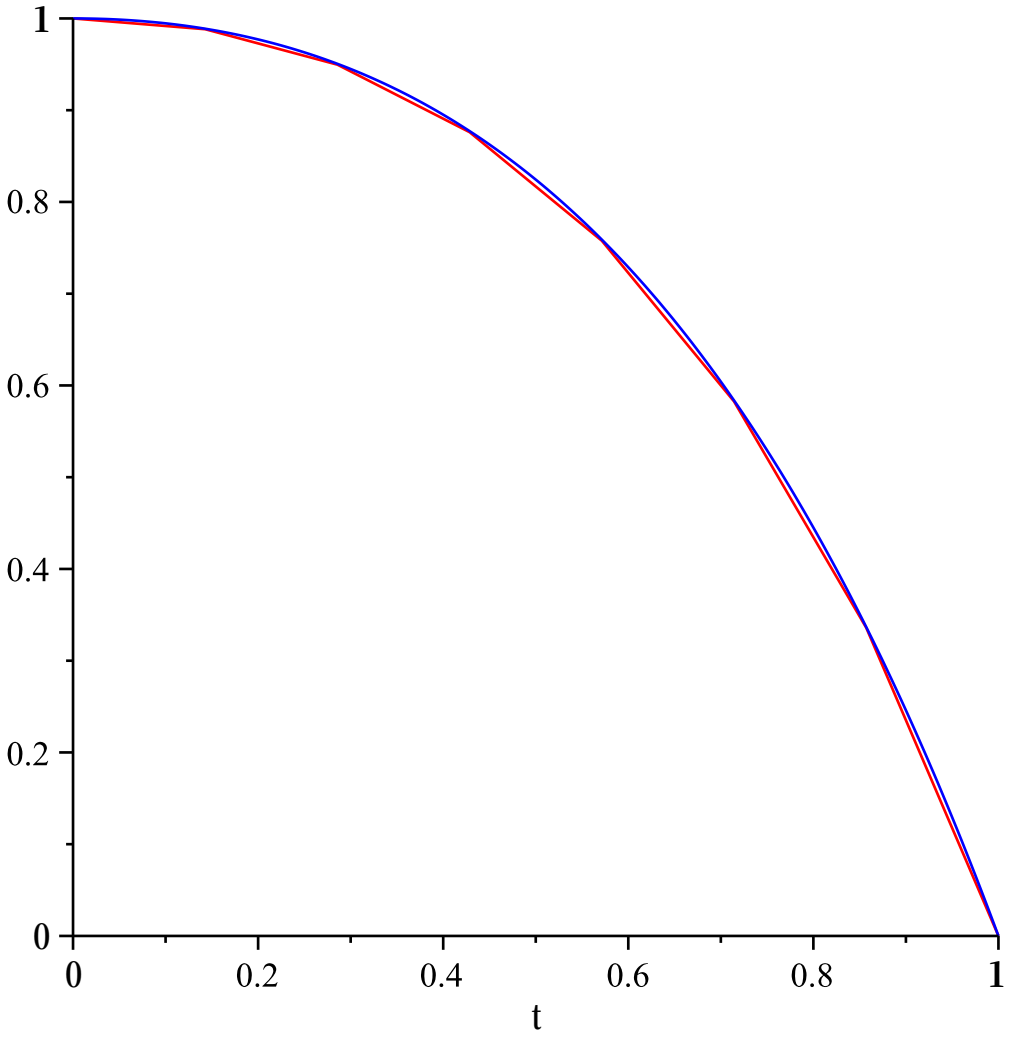
$$u := \begin{bmatrix} 0.988257292849144 \\ 0.949607803289506 \\ 0.876036440128515 \\ 0.757701097233937 \\ 0.582562337067749 \\ 0.335942313876310 \end{bmatrix} \tag{13}$$

ステップ６　境界値の付加

```
> uu:=Vector(1..n+1):
> for i from 1 to n-1 do
    uu[i+1]:=u[i];
  end do:
> uu[1]:=1.0:
> uu[n+1]:=0.0:
```

```
> f1:=plot(x,uu):
```

```
> f2:=plot((1-t)*exp(t),t=0.0..1.0,color=blue):
> with (plots):
> display(f1,f2);
```




```
>
```

非定常な問題：拡散現象

時間発展する非定常な問題として、拡散現象を考える。

1次元初期値問題：拡散方程式

$$\frac{\partial u(x,t)}{\partial t} = D\frac{\partial^2}{\partial x^2} u(x,t) \qquad x \in R$$

$$u(x,0)=\exp(-x^2)$$

格子上の関数

空間に対して区間(-K,K)を分割し、時間に対して区間(0,T)を分割して、各格子点上での関数値を設定する。区間(-K,K)をn分割し、
区間(0,T)をm分割する。
格子点数はn+1で、

格子間隔 $h=\frac{1}{n}$

格子点の位置 $x_i = ih,\ i = 0, 1, \cdots n$

時刻$t_k = k\Delta t,\ k = 0, 1, 2, \cdots m$
格子点x_iでのuの値$u(x_i,t_k)=u_{i,k}$
格子点での関数値$\exp\left(-(x_i)^2\right)$
を定義できる。

時間の差分スキーム

陽的スキーム

$$\frac{u_{i,k+1}-u_{i,k}}{\Delta t} = D_{i+\frac{1}{2}} \frac{u_{i+1,k}+u_{i-1,k}-2\cdot u_{i,k}}{h^2}$$

$$u_{i,0} = u_0(ih), \quad 1 \le i \le n-1$$

$$u_{0,k} = u_{n,k} = 0, \quad k \ge 0$$

陰的スキーム

$$\frac{u_{i,k}-u_{i,k-1}}{\Delta t} = D_{i+\frac{1}{2}} \frac{u_{i+1,k}+u_{i-1,k}-2\cdot u_{i,k}}{h^2}$$

$$u_{i,0} = u_0(ih), \quad 1 \le i \le n-1$$

$$u_{0,k} = u_{n,k} = 0, \quad k \ge 0$$

D≡1とする。これを行列で表現すると、(n-1)×(n-1)行列として、

$$A=\frac{1}{h^2}\begin{bmatrix} 2 & -1 & 0 & 0 & 0 \\ -1 & 2 & -1 & 0 & 0 \\ 0 & \cdot & \cdot & \cdot & 0 \\ 0 & 0 & \cdot & \cdot & -1 \\ 0 & 0 & 0 & -1 & 2 \end{bmatrix}$$

とすると、

陽的スキーム

$$u^{k+1} = (E - \Delta t A)u^k, \quad \frac{D\Delta t}{h^2} \le \frac{1}{2}$$

$$u^0,\ given$$

陰的スキーム

$$(E + \Delta t A)u^k = u^{k-1}$$

$$u^0,\ given$$

差分スキームのプログラミング（陰的スキームと陽的スキーム）

ステップ１　格子点の生成

例えば、空間の区間(-10,10)をn=50分割, 時間の区間(0,1.6)を10分割する。D≡1とする。

```
> n:=50;
```

$$n := 50 \qquad (14)$$

```
> gridpoints:=n;
```

(15)

$$gridpoints := 50 \quad (15)$$

```
> h:=evalf(20/gridpoints,7);
```

$$h := 0.4000000 \quad (16)$$

```
> deltat:=0.16;
```

$$deltat := 0.16 \quad (17)$$

```
> x:=Vector(1..n-1):
> for i from 1 to n-1 do:
  x[i]:=-10.0+h*i;
  end do:
```

ステップ２　係数行列の設定

```
> E:=Matrix(n-1,n-1,shape=identity):
> A:=Matrix(1..n-1,1..n-1):
> for i from 1 to n-2 do:
     A[i,i+1]:=-1.0/(h*h):
     A[i+1,i]:=-1.0/(h*h):
  end do:
> for i from 1 to n-1 do:
     A[i,i]:=2.0/(h*h):
  end do:
> B:=E+deltat*A:
> C:=E-deltat*A:
> A:
```

ステップ３　初期値の設定

```
> u0:=Vector(1..n-1):
> uk:=Vector(1..n-1):
> vk:=Vector(1..n-1):
> uu:=Vector(1..n-1):
> for i from 1 to n-1 do:
      u0[i]:=exp(-x[i]*x[i]):
  end do:
> uk:=u0:
> vk:=u0:
> with(LinearAlgebra):
> m:=10;
```

$$m := 10 \quad (18)$$

```
> k:=1;
```

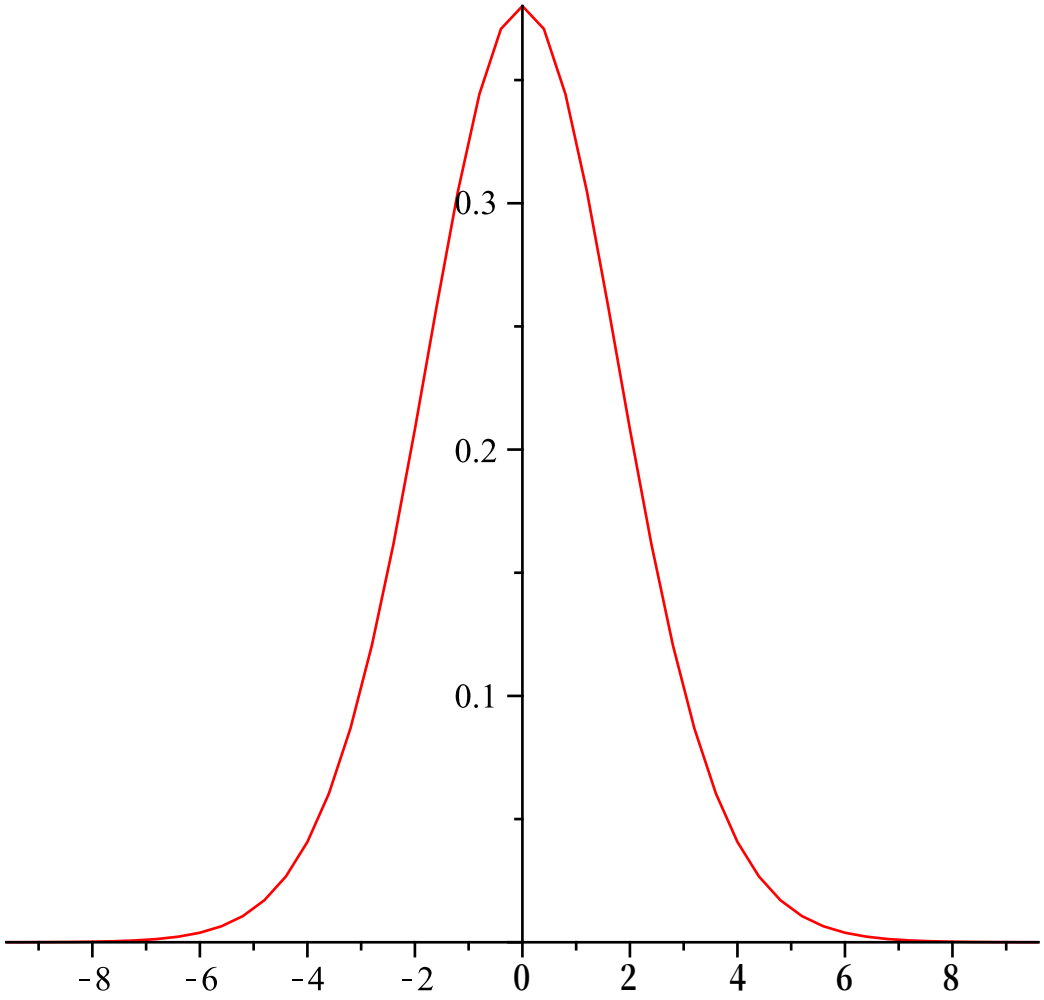
$$k := 1 \quad (19)$$

ステップ４　解法

```
> for k from 1 to m do:
     uu:=uk:
     v:=C.vk:
     u:=LinearSolve(B,uk):
     uk:=u:
     vk:=v:
  end do:
>
```

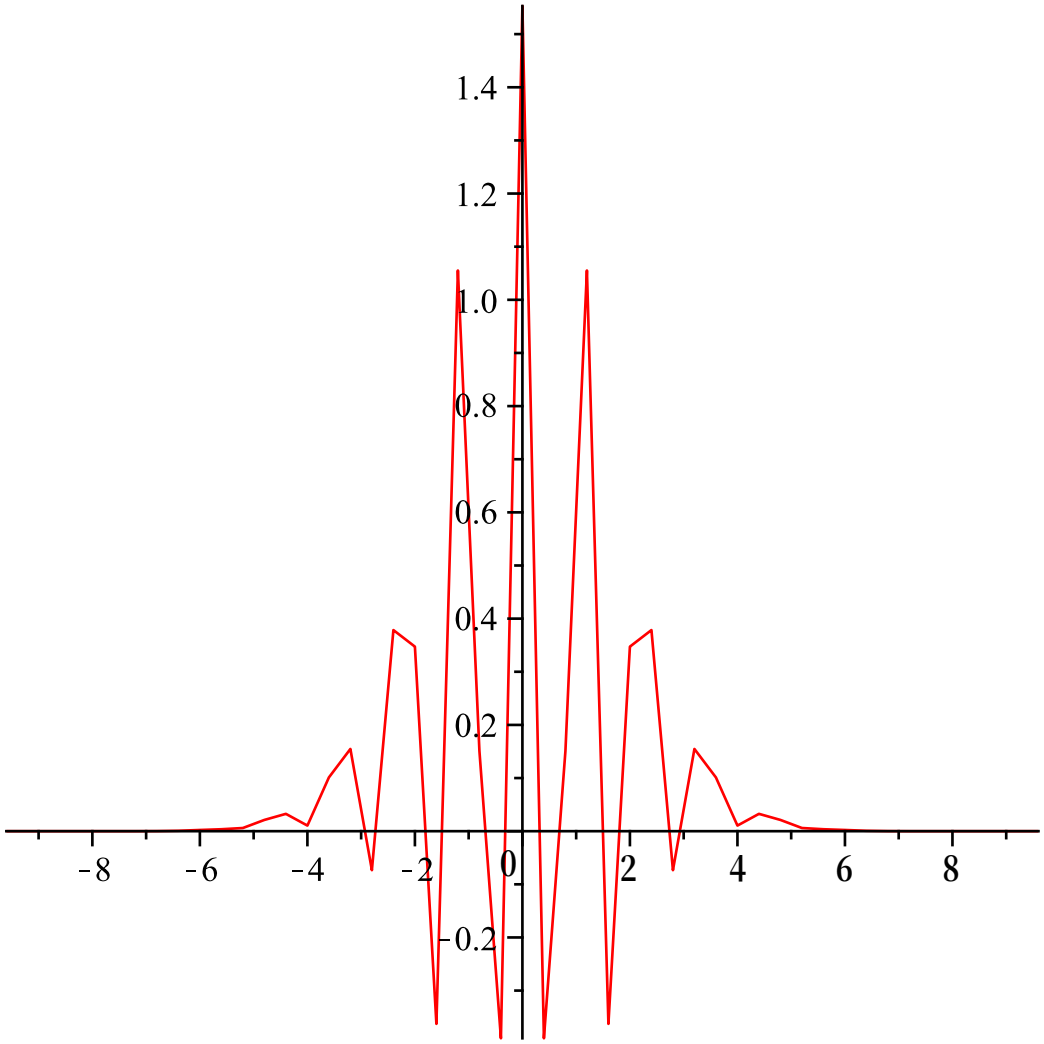
陰的スキームによる解

```
> plot(x,u);
```

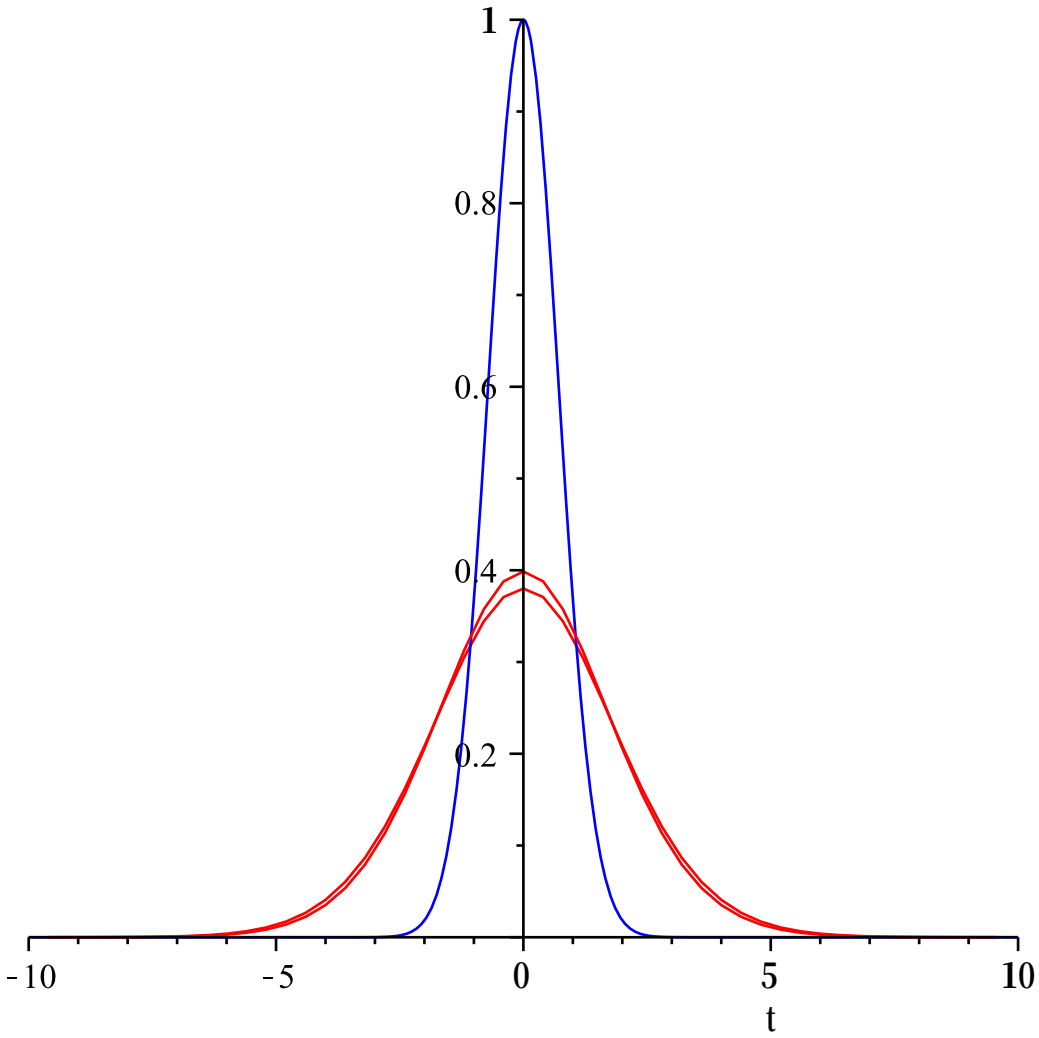


陽的スキームによる解 $\left(\frac{D\Delta t}{h^2} > \frac{1}{2}\text{の場合}\right)$

```
> plot(x,v);
```




```
> f1:=plot(x,u):
> f2:=plot(exp(-t*t),t=-10.0..10.0,color=blue):
> f3:=plot(x,uu):
> with (plots):
> display(f1,f2,f3);
```



>

非定常な問題：移流現象

時間発展する非定常な問題として、移流現象を考える。

1次元初期値問題：移流方程式

$$\frac{\partial u(x,t)}{\partial t} + a\frac{\partial}{\partial x}u(x,t) = 0 \qquad x \in R,\ t \geq 0$$

$$u0(x) = \begin{cases} 1.0 & x < 0 \\ 0.0 & x > 0 \end{cases}$$

上記の初期値問題をRiemann問題と呼ぶが、その厳密解はu0(x-at)である。

>

格子上の関数

空間に対して区間(-K,K)を分割し、時間に対して区間(0,T)を分割して、各格子点上での関数値を設定する。区間(-K,K)をn分割し、
区間(0,T)をm分割する。
格子点数はn+1で、

格子間隔 h=$\frac{1}{n}$

格子点の位置$x_i = ih,\ \ i = 0, 1, \cdots n$

時刻$t_k = k\Delta t,\ \ k = 0, 1, 2, \cdots m$

格子点x_iでのuの値$u(x_i,t_k)=u_{i,k}$

を定義できる。

陽的差分スキーム

風上スキーム

$$u_{i,k+1} = u_{i,k} - \frac{ak}{h}\left(u_{i,k} - u_{i-1,k}\right)$$

$$u_{i,0} = u_0(ih),\quad 1 \le i \le n-1$$

Lax-Wendroffスキーム

$$u_{i,k+1} = u_{i,k} - \frac{ak}{2h}\left(u_{i+1,k} - u_{i-1,k}\right) + \frac{k^2a^2}{2h^2}\left(u_{i+1,k} + u_{i-1,k} - 2\cdot u_{i,k}\right)$$

$$u_{i,0} = u_0(ih),\quad 1 \le i \le n-1$$

Lax-Wendroffスキームの導出

Taylor展開して、

$u(x,t+k)=u(x,t)+ku_t(x,t)+\frac{1}{2}k^2u_{tt}(x,t)+\cdots$

ここで、

$u_{tt} = -au_{tx} = -a\left(-au_x\right)_x = a^2u_{xx}$

を代入して、

$u(x,t+k)=u(x,t)-aku_x(x,t)+\frac{1}{2}k^2a^2u_{xx}(x,t)+\cdots$

これより導出される高精度スキームになっている。

差分スキームのプログラミング

ステップ１　格子点の生成

空間の区間(-1.0,1.0)をn=200分割。a≡1とする。

```
> n:=200;
```

$$n := 200 \qquad (20)$$

```
> gridpoints:=n;
```

$$gridpoints := 200 \qquad (21)$$

```
> h:=evalf(2.0/gridpoints,9);
```

$$h := 0.0100000000 \qquad (22)$$

```
> deltat:=evalf(0.125*h,9);
```

$$deltat := 0.00125000000 \qquad (23)$$

```
> a:=1.0;
```

$$(24)$$

$$a := 1.0 \tag{24}$$

```
> xx:=Vector(1..n+1):
> for i from 1 to n+1 do:
  xx[i]:=evalf(-1.0+h*(i-1),9);
  end do:
```

ステップ2　初期値の設定

```
> u0:=Vector(1..n+1):
> uexact:=Vector(1..n+1):
> uk:=Vector(1..n+1):
> vk:=Vector(1..n+1):
> u:=Vector(1..n+1):
> v:=Vector(1..n+1):
> for i from 1 to n+1 do:
     if xx[i]<=0.0 then
        u0[i]:=1.0
     else
        u0[i]:=0.0:
     end if:
  end do;
```

```
> uk:=u0:
> vk:=u0:
> m:=200;
```

$$m := 200 \tag{25}$$

```
> at:=evalf(a*deltat*m,9);
```

$$at := 0.250000000 \tag{26}$$

ステップ3　厳密解

```
> for i from 1 to n+1 do:
     if xx[i]<=at then
        uexact[i]:=1.0
     else
        uexact[i]:=0.0:
     end if:
  end do;
> k:=1;
```

$$k := 1 \tag{27}$$

```
> l:=1;
```

$$l := 1 \tag{28}$$

ステップ4 解法　upwind

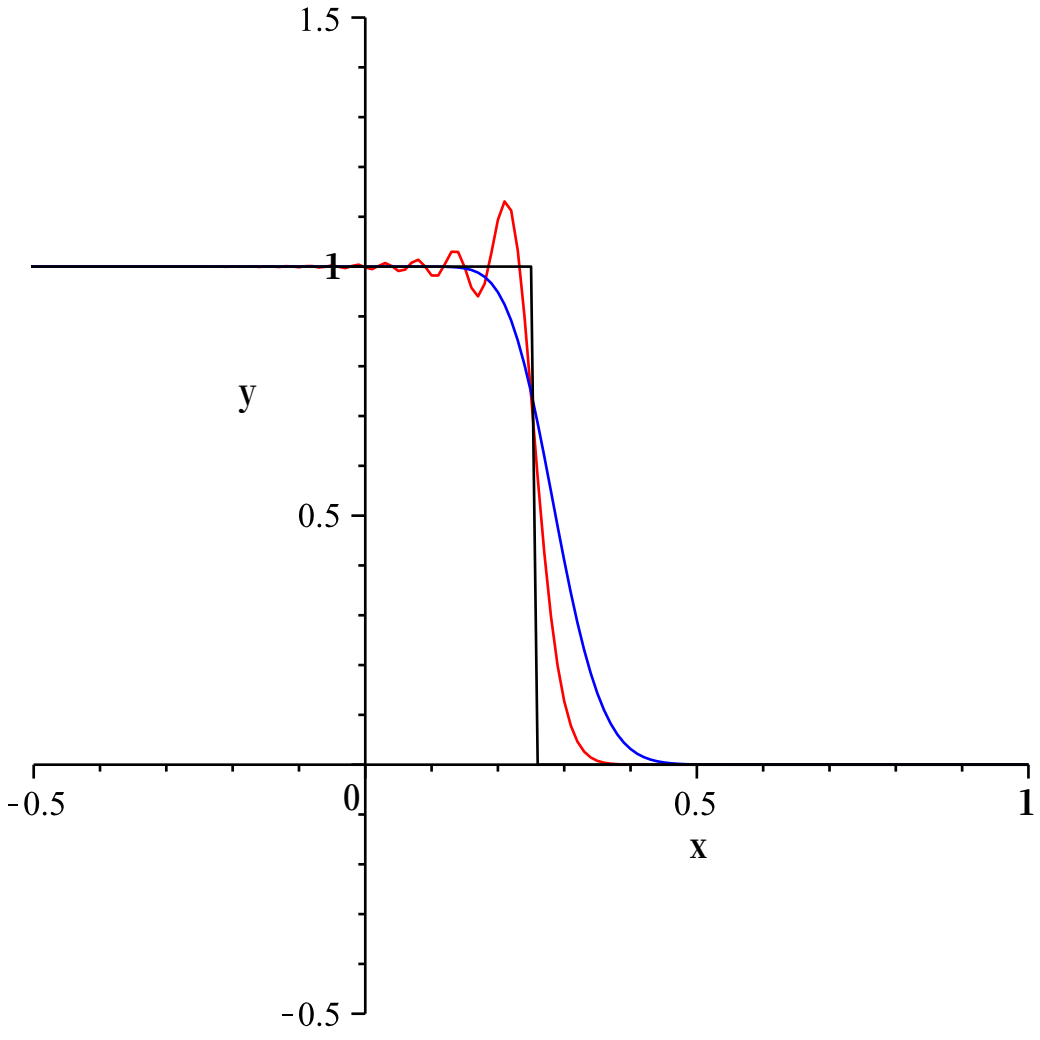
```
> while (k<m ) do:
    for i from 2 to n do:
     u[i]:=uk[i]-(a*deltat/h)*(uk[i]-uk[i-1]):
    end do:
     u[1]:=1.0:
     u[n+1]:=0.0:
     uk:=u:
     k:=k+1
  end do:
```

ステップ4 解法　Lax-Wendroff

```
> while (l<m ) do:
```

```
    for i from 2 to n do:
      v[i]:=vk[i]-(0.5*a*deltat/h)*(vk[i+1]-vk[i-1])+(0.5*a^2*
    (deltat^2)/(h^2))*(vk[i+1]-2.0*vk[i]+vk[i-1]):
    end do:
      v[1]:=1.0:
      v[n+1]:=0.0:
      vk:=v:
      l:=l+1:
    end do:
```

```
> f1:=plot((xx,v),x=-0.5..1.0,y=-0.5..1.5, color=red):
> f2:=plot((xx,u),x=-0.5..1.0,y=-0.5..1.5,color=blue):
> f3:=plot((xx,uexact),x=-0.5..1.0,y=-0.5..1.5,color=black):
> with (plots):
> display(f1,f2,f3);
```




```
>
>
```