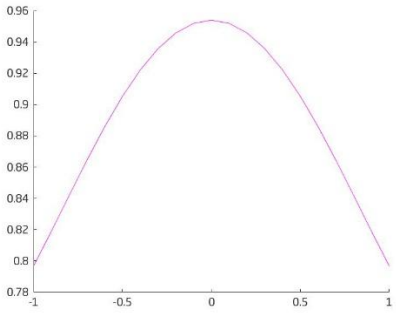
1.解$u_t = bu_{xx}$的数值解，边界条件和初值条件用精确解，其精确解为

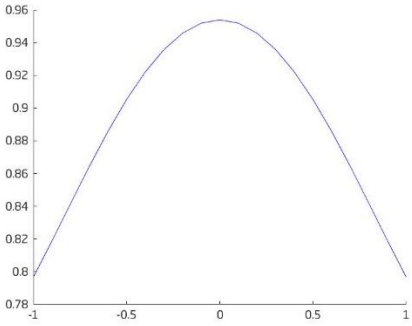
$$u(t,x) = \exp\left(\frac{-(x-y)^2}{4b(t+\tau)}\right)$$

在这里，我们取$b = 0.1$,$\tau = 10$，$y = 0$，其结果为

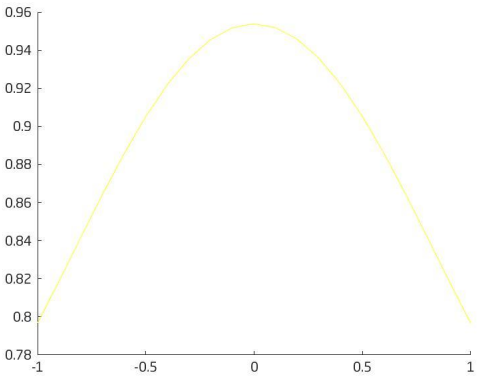
C-N shame



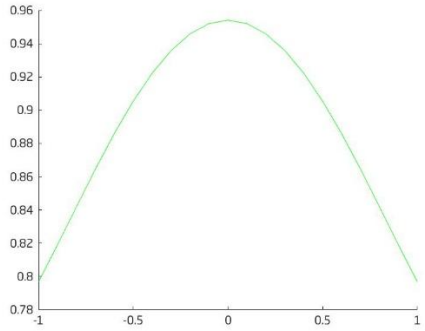
forward-time central-space shame



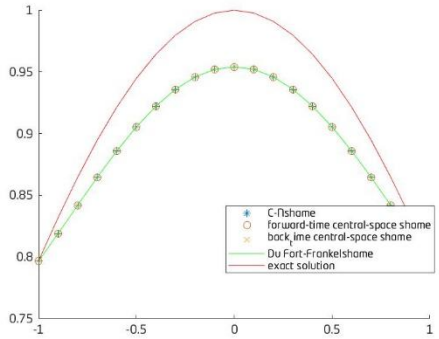
back-time central-space shame



Du Fort-Frankel shame



将其放到一个图像中为：



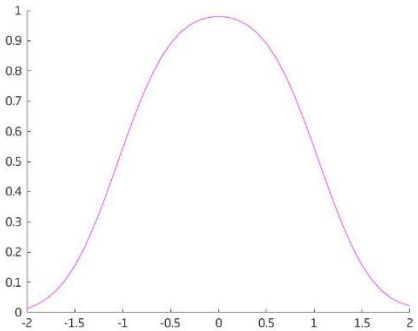
2. 画出方程$u_t = bu_{xx}$的精确解

初值条件为$u_0(x) = \begin{cases} 1 & if\ |x| \le 1 \\ 0 & if\ |x| > 1 \end{cases}$

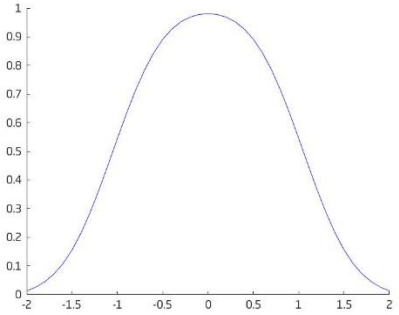
边界条件和初值条件用精确解，其精确解为

$$u(t,x) = \frac{1}{2}\left(erf\left(\frac{a-x}{\sqrt{abt}}\right) + erf\left(\frac{a+x}{\sqrt{abt}}\right)\right)$$

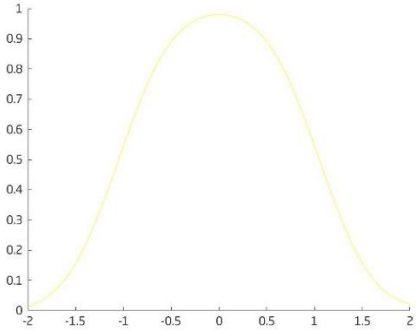
C-N shame



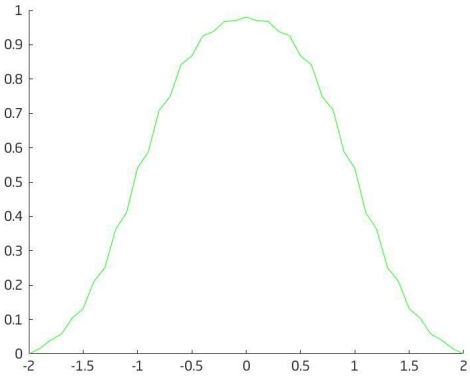
forward-time central-space shame



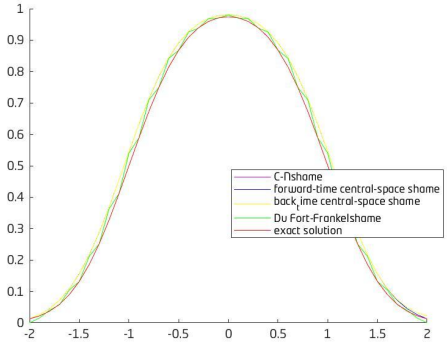
back-time central-space shame



Du Fort-Frankel shame



将上述四种方案与精确解放在一起，其图像为：



附件 源代码

```
%%追赶法 Thomas Algorithm 解三对角矩阵
function [ w] = TA( A,v,d )
%A为一个储存系数矩阵系数的矩阵，A的每一行分别代表a，b,c.
%v是一个二维向量，储存着w0和wm的值
%d代表线性方程组增广矩阵的最后一列
n=length(d);%n存储将要求解未知量的个数m-1个
p=zeros(1,n+1);
q=zeros(1,n+1);
w=zeros(1,n+2);%w0放在第一个位置，wi放在第i+1个位置
w(1)=v(1);
w(n+2)=v(2);
q(1)=v(1);%为p1，q1赋初值
for i=1:1:n %先求出pq的值
    p(i+1)=-1/(A(1)*p(i)+A(2))*A(3);
    q(i+1)=1/(A(1)*p(i)+A(2))*(d(i)-A(1)*q(i));
end
for j=n:-1:1
    w(j+1)=p(j+1)*w(j+2)+q(j+1);
end
end
%%%n个未知量最后输出n+2个值
```

问题1的求解

```
clear all;
clc
te=1;%所求的时刻
b=0.1;
tau=10;
y=0;
h=0.1;
nu=1/2;
k=nu*h*h;%%%%可以修改
x=-1:h:1;
t=0:k:te;
ujingque=exp(-(x-y).^2/(4*b*(te+tau)));%精确解
%%%%%%%%%%%C—N格式
V=zeros(length(t),length(x));
V(1,:)=exp(-(x-y).^2/(4*b*(0+tau)));%初值条件
A=ones(1,3);
A(1)= -b*nu/2;        %存储a
A(2)= 1+b*nu   ;  %存储b
```

```
A(3)=-b*nu/2;        %存储c
d=zeros(1,length(x)-2);
for i=2:1:length(t)%%开始求每个时间所对应的值

v=[exp(-(-1-y)^2/(4*b*(t(i)+tau))),exp(-(1-y)^2/(4*b*(t(i)+tau)))]';
    for j=1:length(d)

d(j)=1/2*nu*b*V(i-1,j+2)+(1-b*nu)*V(i-1,j+1)+1/2*nu*b*V(i-1,j);
    end
   [V(i,:)]=TA(A,v,d); %%v，d在不同步骤中不同，需要分别求
end
%%%%%%%%古典显格式
V2=zeros(length(t),length(x));
V2(1,:)=exp(-(x-y).^2/(4*b*(0+tau)));%初值条件
for i=2:1:length(t)%%开始求每个时间所对应的值
    V2(i,1)=exp(-(-1-y).^2/(4*b*(t(i)+tau)));
    V2(i,length(x))=(exp(-(1-y).^2/(4*b*(t(i)+tau))));
    for j=2:length(x)-1
        V2(i,j)=(1-2*b*nu)*V2(i-1,j)+b*nu*(V2(i-1,j-1)+V2(i-1,j+1));
    end
end
%%%%%%%%%%%back_time central-space格式
V3=zeros(length(t),length(x));
V3(1,:)=exp(-(x-y).^2/(4*b*(0+tau)));%初值条件
A=ones(1,3);
A(1)= -b*nu;         %存储a
A(2)= 1+2*b*nu   ;   %存储b
A(3)=-b*nu;        %存储c
d=zeros(1,length(x)-2);
for i=2:1:length(t)%%开始求每个时间所对应的值

v=[exp(-(-1-y)^2/(4*b*(t(i)+tau))),exp(-(1-y)^2/(4*b*(t(i)+tau)))]';
    for j=1:length(d)
        d(j)=V(i-1,j+1);
    end
   [V3(i,:)]=TA(A,v,d); %%v，d在不同步骤中不同，需要分别求
end
%%%%%DFF格式
V4=zeros(length(t),length(x));
V4(1,:)=exp(-(x-y).^2/(4*b*(0+tau)));%初值条件
V4(2,:)=exp(-(x-y).^2/(4*b*(k+tau)));%初值条件
for i=3:1:length(t)%%开始求每个时间所对应的值
    V4(i,1)=exp(-(-1-y)^2/(4*b*(t(i)+tau)));
    V4(i,length(x))=(exp(-(1-y)^2/(4*b*(t(i)+tau))));
```

```
    for j=2:length(x)-1

V4(i,j)=1/(1+2*b*nu)*(2*b*nu*(V4(i-1,j+1)+V4(i-1,j-1))+(1-2*b*nu)*V4(
i-2,j));
    end
end

hold on;
plot(x,(V(length(t),:))',*');
plot(x,(V2(length(t),:))','o');
plot(x,(V3(length(t),:))','x');
plot(x,(V4(length(t),:))','g');
plot(x,ujingque,'r');
legend('C-Nshame','forward-time central-space shame','back_time
central-space shame','Du Fort-Frankelshame','exact solution');
```

问题 2 的求解

```
clear all;
clc
te=1;%所求的时刻

b=0.1;
tau=10;
y=0;

h=0.1;
nu=1/2;
k=nu*h*h;%%%%可以修改
x=-2:h:2;
t=0:k:te;
ujingque=1/2*(erf((1-x)/sqrt(4*b*te))+erf((1+x)/sqrt(4*b*te)));%精确
解
%%%%%初值条件
u0=zeros(1,length(x));
for i=(1/h+1):1:(3/h+1)
    u0(i)=1;
end

%%%%%%%%%%C—N格式

V=zeros(length(t),length(x));
V(1,:)=u0;%初值条件
```

```
A=ones(1,3);
A(1)= -b*nu/2;          %存储a
A(2)= 1+b*nu   ;  %存储b
A(3)=-b*nu/2;        %存储c
d=zeros(1,length(x)-2);
for i=2:1:length(t)%%开始求每个时间所对应的值

v=[1/2*(erf((1-x)/sqrt(4*b*t(i)))+erf((1+x)/sqrt(4*b*t(i)))),1/2*(erf
((1-x)/sqrt(4*b*t(i)))+erf((1+x)/sqrt(4*b*t(i))))]';
    for j=1:length(d)

d(j)=1/2*nu*b*V(i-1,j+2)+(1-b*nu)*V(i-1,j+1)+1/2*nu*b*V(i-1,j);
    end
   [V(i,:)]=TA(A,v,d); %%v，d在不同步骤中不同，需要分别求
end
%%%%%%%%古典显格式
V2=zeros(length(t),length(x));
V2(1,:)=u0;%初值条件
for i=2:1:length(t)%%开始求每个时间所对应的值

V2(i,1)=1/2*(erf((1-(-2))/sqrt(4*b*t(i)))+erf((1+(-2))/sqrt(4*b*t(i)))
);

V2(i,length(x))=1/2*(erf((1-2)/sqrt(4*b*t(i)))+erf((1+2)/sqrt(4*b*t(i)
)));
    for j=2:length(x)-1
        V2(i,j)=(1-2*b*nu)*V2(i-1,j)+b*nu*(V2(i-1,j-1)+V2(i-1,j+1));
    end
end
%%%%%%%%%%%back_time central-space格式
V3=zeros(length(t),length(x));
V3(1,:)=u0;%初值条件
A=ones(1,3);
A(1)= -b*nu;          %存储a
A(2)= 1+2*b*nu   ;  %存储b
A(3)=-b*nu;        %存储c
d=zeros(1,length(x)-2);
for i=2:1:length(t)%%开始求每个时间所对应的值

v=[1/2*(erf((1-x)/sqrt(4*b*t(i)))+erf((1+x)/sqrt(4*b*t(i)))),1/2*(erf
((1-x)/sqrt(4*b*t(i)))+erf((1+x)/sqrt(4*b*t(i))))]';
    for j=1:length(d)
        d(j)=V(i-1,j+1);
    end
```

```
   [V3(i,:)]=TA(A,v,d); %%v，d在不同步骤中不同，需要分别求
end

%%%%%DFF格式
V4=zeros(length(t),length(x));
V4(1,:)=u0;%初值条件
V4(2,:)=1/2*(erf((1-x)/sqrt(4*b*t(2)))+erf((1+x)/sqrt(4*b*t(2))));%初
值条件
for i=3:1:length(t)%%开始求每个时间所对应的值

V2(i,1)=1/2*(erf((1-(-2))/sqrt(4*b*t(i)))+erf((1+(-2))/sqrt(4*b*t(i)))
);

V2(i,length(x))=1/2*(erf((1-2)/sqrt(4*b*t(i)))+erf((1+2)/sqrt(4*b*t(i)
)));
    for j=2:length(x)-1

V4(i,j)=1/(1+2*b*nu)*(2*b*nu*(V4(i-1,j+1)+V4(i-1,j-1))+(1-2*b*nu)*V4(
i-2,j));
    end
end

hold on;

plot(x,(V(length(t),:))',’m’);
 plot(x,(V2(length(t),:))',’b’);
plot(x,(V3(length(t),:))',’y’);
 plot(x,(V4(length(t),:))',’g’);
plot(x,ujingque,’r’);
legend(’C-Nshame’,’forward-time central-space shame’,’back_time
central-space shame’,’Du Fort-Frankelshame’,’exact solution’);
```