BÀI THỰC HÀNH LẦN 3
FINITE DIFFERENT METHOD
PHƯƠNG TRÌNH NHIỆT

LƯU GIANG NAM[1,2]

MỤC LỤC

1 Mã số sinh viên: 1411174
2 Email: luugiangnam96@gmail.com

DANH SÁCH HÌNH VẼ

DANH SÁCH BẢNG

LỜI NÓI ĐẦU

Bài báo cáo thực hành lần 3 môn học Phương pháp sai phân hữu hạn (Finite Different Methods, FDM) Phương trình nhiệt - Heat Equations sẽ bao gồm 7 phần:

- Phần 1: Xây dựng thuật toán tổng quát. Phần này sẽ tập trung vào việc xây dựng lưới và giải bài toán với phương pháp theta tổng quát và giới thiệu các phương pháp thông dụng ứng với các θ cụ thể. Thêm vào đó sẽ giới thiệu code sai số và các ví dụ.
- Phần 2, 3 và 4 sẽ tập trung vào 3 phương pháp phổ biến nhất là Forward Euler, Backward Euler và Crank - Nicolson với 2 ví dụ và xuất ra mô phỏng kết quả cùng với sai số và bậc sai số.
- Phần 5 và 6 sẽ tập trung vào 2 phương pháp θ với $\theta = 1/3$ và $\theta = 2/3$ với 2 ví dụ và xuất ra mô phỏng kết quả cùng với sai số và bậc sai số.
- Phần 7 sẽ tổng kết các kết quả có được ở các phần trên và rút ra kết luận. Kèm thêm là các bảng sai số cho từng chuẩn và từng ví dụ.

Thành phố Hồ Chí Minh, ngày 30 tháng 12 năm 2017

Lưu Giang Nam

1 XÂY DỰNG THUẬT TOÁN TỔNG QUÁT

1.1 Bài toán

Ta trong bài viết này ta sẽ xét một bài toán Phương trình đạo hàm riêng dạng một phương trình nhiệt như sau:

$$\frac{\partial u}{\partial t} = \kappa \frac{\partial^2 u}{\partial x^2} + f(x), \quad \forall\ (x,t) \in [0,1] \times [0,T] \tag{1.1}$$

với $f \in L^2([0,1] \times [0,T])$.

Để giải được phương trình trên ta cần điều kiện đầu

$$u(x,0) = u_0(x) \tag{1.2}$$

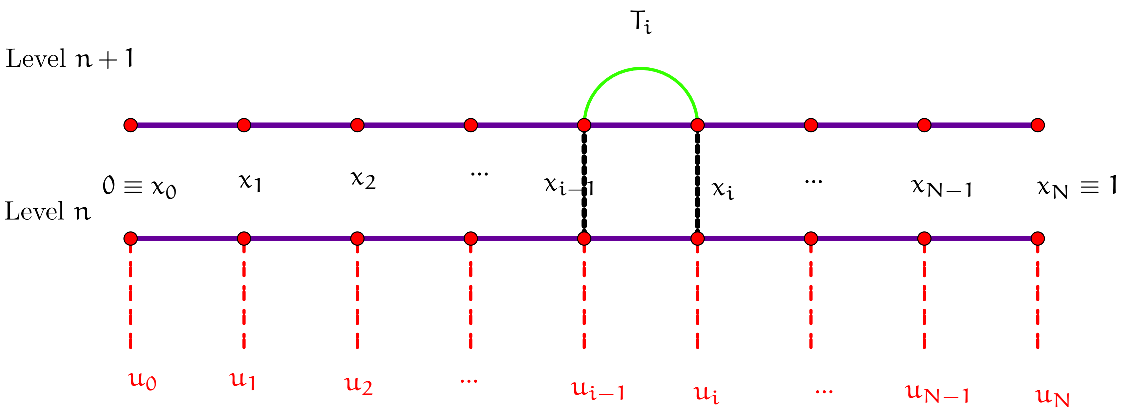
và điều kiện biên

$$u(0,t) = 0 \tag{1.3}$$

$$u(1,t) = 0 \tag{1.4}$$

1.2 Xây dựng lưới và các ma trận

Ta sẽ xét một lưới đều với $N_x + 1$ điểm x_i với $i = 0, 1, ..., N_x$ với khoảng cách là $h = \frac{1}{N_x}$. Trên trục thời gian ta cũng chia $[0,T]$ thành N_t khoảng với độ đo mỗi khoảng là $k = \frac{T}{N_t}$ (xem hình).



Khi đó ta có:

$$x_i = ih, \quad \text{và} \quad t_n = nk \tag{1.5}$$

Đặt $U_i^n = u(x_i, t_n)$ là giá trị xấp xỉ trên lưới tại điểm (x_i, t_n).

Code Matlab cho phần nhập số liệu.

```
%% ---------- Initial information -----------------
ax = 0;
bx = 1;
T = 1;
dx = 0.1;   %h
dt = 0.0025; %k
Nt = 400;
Nx = 10;

number_mesh=4;
number_mesh_point=zeros(number_mesh,1);
error_H0=zeros(number_mesh,1);
```

```
error_L2=zeros(number_mesh,1);
number_mesh_point(jj)=Nx;

theta = 2/3;
% kap = 1; % Example 1
% kap = 1/2; % Example 2
```

Code Matlab cho phần tạo lưới và các điểm:

```
%% ------------------Create vectors----------------

    x= zeros(Nx+1,1);
    for i = 1:Nx+1
        x(i) = ax + (bx - ax)*(i-1)/Nx;
    end

    t = zeros(Nt+1,1);
    for i = 1:Nt
        t(i) = dt*(i-1);
    end

    u0 = zeros(Nx-1,1);
    for i = 2:Nx
        u0(i-1) = fu0(x(i));
    end
```

Ta sẽ sử dụng phương pháp θ (θ- method) để giải bài toán trên:

$$\frac{U_i^{n+1} - U_i^n}{k} = \kappa((1-\theta)D_2U_i^n + \theta D_2U_i^{n+1}) + f(x_i, t_n) \tag{1.6}$$

với

$$D_2U_i^n = \frac{U_{i-1}^n - 2U_i^n + U_{i+1}^n}{h^2}, \quad D_2U_i^{n+1} = \frac{u_{i-1}^{n+1} - 2U_i^{n+1} + U_{i+1}^{n+1}}{h^2}$$

Khi đó ta có thể biến đổi thành:

$$\frac{U_i^{n+1} - U_i^n}{k} = \frac{\kappa}{h^2}\left((1-\theta)(U_{i-1}^n - 2U_i^n + U_{i+1}^n) + \theta(U_{i-1}^{n+1} - 2U_i^{n+1} + U_{i+1}^{n+1})\right) + f(x_i, t_n) \tag{1.7}$$

Hay

$$U_i^{n+1} = U_i^n + \frac{\kappa k}{h^2}\left((1-\theta)(U_{i-1}^n - 2U_i^n + U_{i+1}^n) + \theta(U_{i-1}^{n+1} - 2U_i^{n+1} + U_{i+1}^{n+1})\right) + kf(x_i, t_n) \tag{1.8}$$

Đặt $r = \frac{\kappa k}{h^2}$, khi đó ta có:

$$\begin{aligned} &-r\theta U_{i-1}^{n+1} + (1+2r\theta)U_i^{n+1} - r\theta U_{i+1}^{n+1} \\ &= r(1-\theta)U_{i-1}^n + (1-2r(1-\theta))U_i^n + r(1-\theta)U_{i+1}^n + kf(x_i, t_n) \end{aligned}$$

Tiếp tục đặt A thành:

$$A = \begin{bmatrix} -2r & r & 0 & 0 & \cdots & 0 & 0 & 0 \\ r & -2r & r & 0 & \cdots & 0 & 0 & 0 \\ 0 & r & -2r & r & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & r & -2r & r \\ 0 & 0 & 0 & 0 & \cdots & 0 & r & -2r \end{bmatrix} \tag{1.9}$$

Khi đó ta biến đổi thành dạng ma trận sau:

$$(I - \theta A)U^{n+1} = (I + (1-\theta)A)U^n + kF^n \tag{1.10}$$

với U^n là vector chứa các điểm $(U_i^n)_{i=\overline{0,N_x}}$ và F^n là vector chứa các điểm $(F(x_i,t_n))_{i=\overline{0,N_x}}$.

Khi đó, ta có thể tính U^{n+1} thành U^n như sau:

$$U^{n+1} = (I-\theta A)^{-1}(I+(1-\theta)A)U^n + k(I-\theta A)^{-1}F^n \tag{1.11}$$

Code Matlab cho ma trận A:

```
%% ---------------Create matric A---------------
A = sparse(Nx-1,Nx-1);
I = eye(Nx-1,Nx-1);
for i = 1:Nx-1
    if i == 1
        A(i,i) = -2;
        A(i,i+1) = 1;
    else
        if i == Nx-1
            A(i,i) = -2;
            A(i,i-1) = 1;
        else
            A(i,i)= -2;
            A(i,i+1) = 1;
            A(i,i-1) = 1;
        end
    end
end

A = A*(kap*dt/dx^2);
```

Ma trận G sẽ được tạo như sau:

```
%% ---------------Theta methods---------------
G1 = I - theta*A;
G2 = I + (1-theta)*A;
G = (G1^(-1))*(G2);
```

Tuy nhiên từng giá trị θ sẽ được chỉnh sửa lại để tính toán bớt phức tạp hơn, ví dụ nếu $\theta = 0$ thì không nhất thiết phải tính G_1^{-1} để bài toán nhanh hơn.

Sau khi có tất cả số liệu, ta sẽ giải bài toán và tính sai số:

```
for it = 2:Nt
    for i = 2:Nx
        F(i-1) = 0.5*f0(x(i),t(it))*dt+ 0.5*f0(x(i),t(it-1))*dt;
    end

    U = G*u0 + (G1^(-1))*F;
    u0 = U;

    figure(jj)
    for i = 2:Nx
        u_dis(i) = U(i-1);
    end
    for i = 2:Nx
        u_ex(i) = u_exact(x(i),t(it));
    end
    plot(x,u_dis,'r',x,u_ex,'b');
%% ---------------------Error---------------------

    for i = 2:Nx
        error_H0 = error_H0 + ...
            (((u_dis(i+1)-u_ex(i+1))-(u_dis(i)-u_ex(i))))^2*dx/dt;
    end

    for i = 2:Nx
        error_L2 = error_L2 + (u_dis(i)-u_ex(i))^2*dx*dt;
    end
end
```

```
    error_H0(jj) = (error_H0(jj))^(1/2);
    error_L2(jj) = (error_L2(jj))^(1/2);
```

Sau mỗi lần lặp, kích thước lưới sẽ giảm $1/2$, thời gian giảm $1/4$. Từ đó số khoảng của không gian và thời gian lần lượt sẽ tăng 2 lần và 4 lần.

```
    dx = dx/2;
    Nx = Nx*2;
    dt = dt/4;
    Nt = Nt*4;
```

Cuối cùng là vẽ bậc hội tu cho chuẩn H_0^2 và L^2.

```
%% ---------------------Error---------------------
figure(number_mesh+1)
plot(log(number_mesh_point), -log(error_H0),'blue',...
    log(number_mesh_point), -log(error_L2), 'red',...
    log(number_mesh_point), 2*log(number_mesh_point),'black',...
    log(number_mesh_point), log(number_mesh_point),'green',...
    log(number_mesh_point), 1/2*log(number_mesh_point),'yellow');
xlabel('Log(MeshPoint)');ylabel('-Log(Error)');
title('Errors');
legend('norm_{H_0}','norm_{L_2}','2x','x','1/2x','Location','NorthEastOutside');
```

Trong các code trên, ta sử dụng 3 functions là u_exact để biểu thị nghiệm chính xác (để so sánh với nghiệm được xấp xỉ), fu0 là giá trị $u_0 = u(x, 0)$ và hàm f0 biểu thị hàm f.

1.3 Các phương pháp giải

Với các giá trị θ nhất định ta sẽ có những phương pháp giải (1.10) khác nhau:

- Với $\theta = 0$ - Forward Euler Methods:
$$U^{n+1} = U^n + AU^n + kF^n \tag{1.12}$$
- Với $\theta = 1$ - Backward Euler
$$U^{n+1} = U^n + AU^{n+1} + kF^n \tag{1.13}$$
- Với $\theta = \frac{1}{2}$ - Crank - Nicolson:
$$U^{n+1} - U^n = \frac{1}{2}\left(AU^n + AU^{n+1}\right) + kF^n \tag{1.14}$$

Từ đó, ta có thể đưa về dạng tổng quát như sau:
$$U^{n+1} = GU^n + kG_1^{-1}F^n \tag{1.15}$$
với

- Forward: $G = I + A$ và $G_1 = I$.
- Backward: $G = (I - A)^{-1}$ và $G_1 = I - A$.
- Crank - Nicolson: $G = \left(I - \frac{1}{2}A\right)^{-1}\left(I + \frac{1}{2}A\right)$ và $G_1 = I - \frac{1}{2}A$.

Điều kiện ổn định của các phương pháp là:
$$r \leqslant \begin{matrix} \dfrac{1}{2(1-2\theta)} & \text{if } \ \theta < \dfrac{1}{2} \\ \infty & \text{if } \ \dfrac{1}{2} \leqslant \theta \leqslant 1 \end{matrix} \tag{1.16}$$

Trong bốn phần tiếp theo ta sẽ xét kĩ từng phương pháp và bổ sung thêm 2 phương pháp ứng với $\theta = \frac{1}{3}$ và $\theta = \frac{2}{3}$ cùng với thêm 1 ví dụ ngoài ví dụ đã có sẵn.

1.4 Ví dụ

Ta sẽ xét hai ví dụ, ví dụ 1 là có sẵn trong đề bài:

VÍ DỤ 1: Xét bài toán đạo hàm riêng - Heat equation sau:

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + f(x,t), \quad \text{với} \quad (x,t) \in [0,1] \times [0,T] \tag{1.17}$$

với $f(x,t) = (2 - 8x + 4x^2 - 2x^3)e^{-2t} \in L^2([0,1] \times [0,T])$.

Điều kiện đầu là:

$$u(x,0) = u_0(x) = x(1-x)^2 \tag{1.18}$$

Điều kiện biên là:

$$u(1,t) = u(0,t) = 0 \tag{1.19}$$

Khi đó nghiệm chính xác sẽ là

$$u_{\text{ex}} = x(1-x)^2 e^{-2t} \tag{1.20}$$

Khi đó ta thấy $\kappa = 1$ nên điều kiện ổn định là:

$$r = \frac{k}{h^2} \leqslant \frac{1}{2(1-2\theta)} \tag{1.21}$$

Nên có thể chọn tổng quát cho mọi θ là $k = \frac{h^2}{4}$.

Ví dụ 2 sẽ được thêm vào ứng với scheme của slide:

VÍ DỤ 2: Xét bài toán đạo hàm riêng - Heat equation sau:

$$\frac{\partial u}{\partial t} = \frac{1}{2}\frac{\partial^2 u}{\partial x^2}, \quad \text{với} \quad (x,t) \in [0,1] \times [0,T] \tag{1.22}$$

Điều kiện đầu là:

$$u(x,0) = u_0(x) = \sin(2\pi x) \tag{1.23}$$

Điều kiện biên là:

$$u(1,t) = u(0,t) = 0 \tag{1.24}$$

Khi đó nghiệm chính xác sẽ là

$$u_{\text{ex}} = e^{-2\pi^2 t}\sin(2\pi x) \tag{1.25}$$

Khi đó ta thấy $\kappa = \frac{1}{2}$ nên điều kiện ổn định là:

$$r = \frac{k}{2h^2} \leqslant \frac{1}{2(1-2\theta)} \tag{1.26}$$

Nên có thể chọn tổng quát cho mọi θ là $k = \frac{h^2}{4}$.

Khi đó sẽ chọn $k = \frac{h^2}{4}$, các giá trị k, h, N_x và N_t lần lượt là:

k	h	N_x	N_t
0.01	0.2	5	100
0.0025	0.1	10	400
0.625×10^{-3}	0.05	20	1600
0.15625×10^{-3}	0.025	40	6400
0.15625×10^{-3}	0.0125	80	25600
0.0390625×10^{-3}	0.00625	160	102400

Bảng 1: Bảng các số liệu về lưới qua các lần lặp

2 PHƯƠNG PHÁP FORWARD EULER

2.1 Mô phỏng kết quả của Phương pháp Forward Euler

2.1.1 *Nội dung phương pháp*

Có thể giải thích từ Forward ở đây là vì ở phương trình (1.6) với $\theta = 0$ thì phần $D_2U_i^{n+1}$ sẽ biến mất và chỉ còn $D_1U_i^n$ và bài toán sẽ chuyển về dạng:

$$U^{n+1} = U^n + AU^n + kF^n \tag{2.1}$$

hay

$$U^{n+1} = GU^n + kF^n \tag{2.2}$$

với $G = I + A$.

Như vậy ta có thể tính trực tiếp U^{n+1} theo U^n mà không cần sử dụng đến phép nghịch đảo nào.

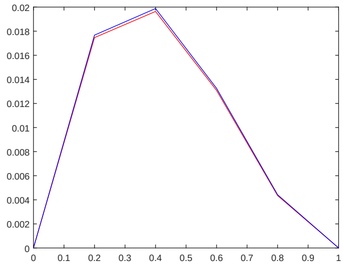
Code Matlab cho G ứng với $\theta = 0$:

```
%% ----------------Forward Method----------------
G = I+A;
```

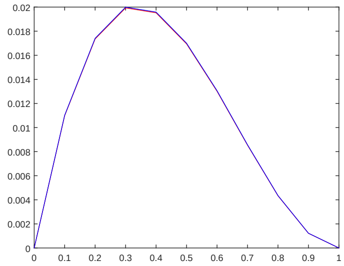
Ngoài ra vì $G_1 = I$ nên sẽ không cần code G_1 để tính G_1^{-1}.

2.1.2 *Các kết quả cho ví dụ 1*

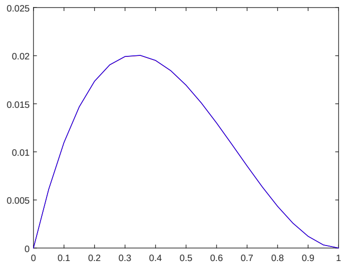
Sau khi thực hành trên Matlab ta có được kết quả xấp xỉ so với kết quả chính thức như sau:



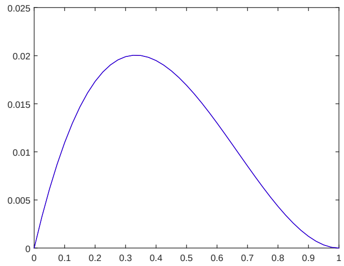
(a) Lần lặp đầu tiên



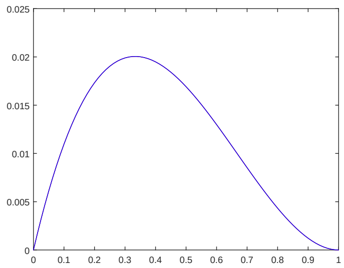
(b) Lần lặp thứ hai



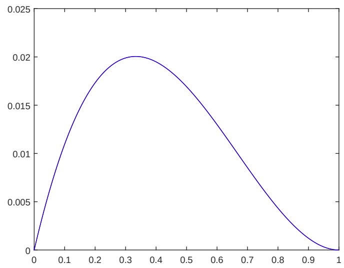
(c) Lần lặp thứ ba



(d) Lần lặp thứ bốn

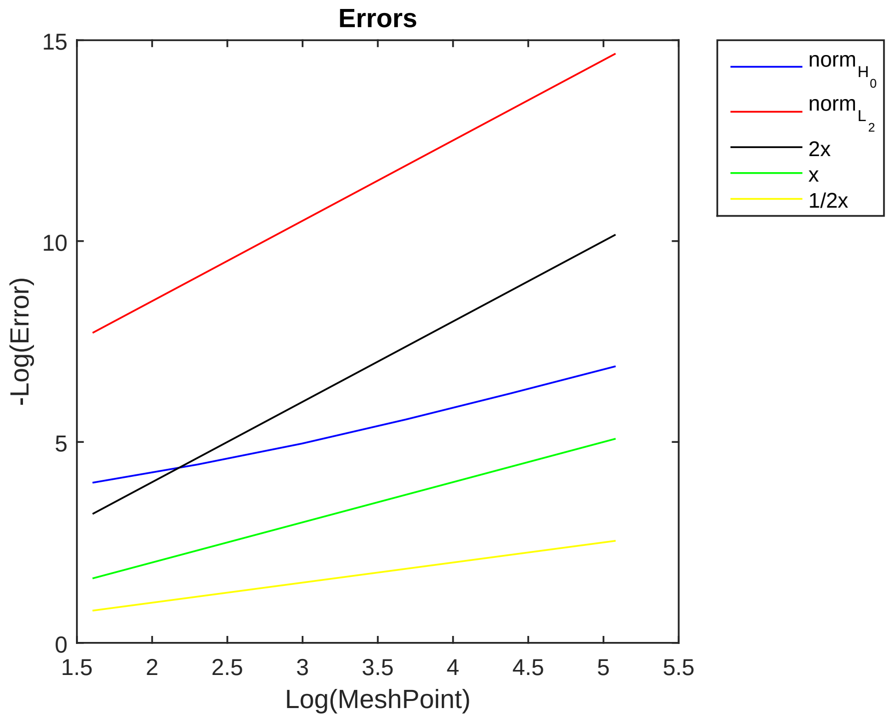


(e) Lần lặp thứ năm



(f) Lần lặp thứ sáu

Hình 1: Ví dụ 1 của Forward Euler Method được mô phỏng qua 6 lần chia lưới mịn hơn



Hình 2: Bậc sai số của Ví dụ 1 của Forward Euler Method

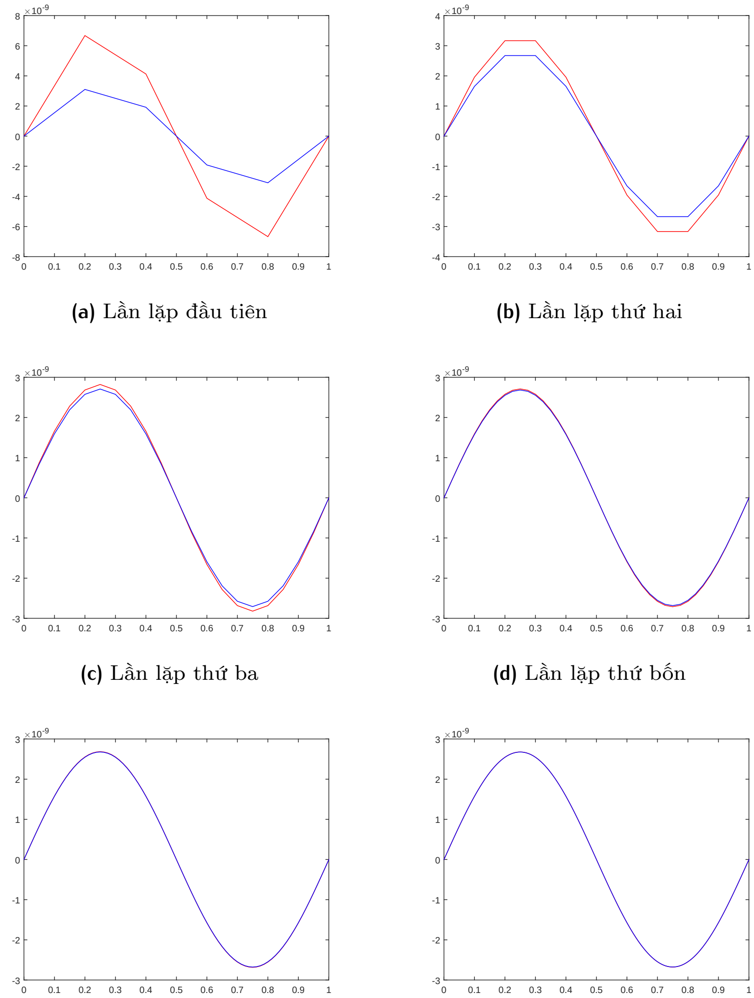
Dưới đây là kết quả tính xấp xỉ sau các lần chia lưới min hơn của Forward Euler Method, ví dụ 1:

k	Chuẩn H_0^2	Chuẩn L^2
0.01	0.018502231608067	0.441231466935645 e-03
0.0025	0.011802220786129	0.110434926763612 e-03
0.625×10^{-3}	0.007010926438865	0.027603490478027 e-03
0.15625×10^{-3}	0.003841827380480	0.006900331734321 e-03
0.0390625×10^{-3}	0.002012051965775	0.001725045753503 e-03
$0.009765625 \times 10^{-3}$	0.001029558806433	0.000431259053170 e-03

Bảng 2: Bảng sai số cho các chuẩn H_0^2 và chuẩn L^2, ví dụ 1 Forward Euler Method

2.1.3 *Các kết quả cho ví dụ 2*

Sau khi thực hành trên Matlab ta có được kết quả xấp xỉ so với kết quả chính thức như sau:



(a) Lần lặp đầu tiên

(b) Lần lặp thứ hai

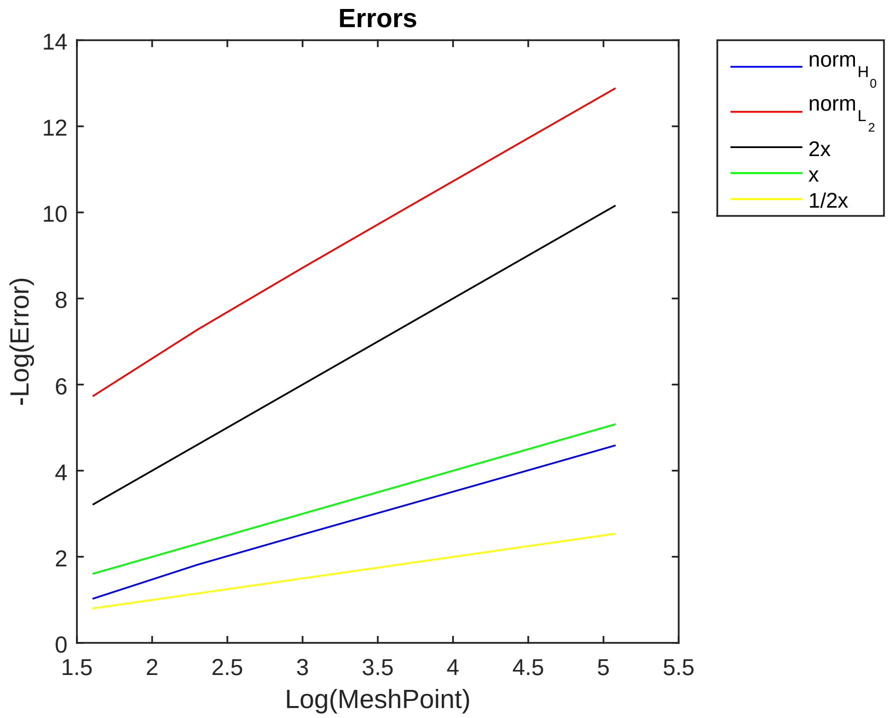
(c) Lần lặp thứ ba

(d) Lần lặp thứ bốn

(e) Lần lặp thứ năm

(f) Lần lặp thứ sáu

Hình 3: Ví dụ 2 của Forward Euler Method được mô phỏng qua 4 lần chia lưới mịn hơn



Hình 4: Bậc sai số của Ví dụ 2 của Forward Euler Method

Dưới đây là kết quả tính xấp xỉ sau các lần chia lưới min hơn của Forward Euler Method, ví dụ 2:

k	Chuẩn H_0^2	Chuẩn L^2
0.01	0.357242828023685	0.003219206545401
0.0025	0.162611779302316	0.000689521074159
0.625×10^{-3}	0.080964483576979	0.000165775353510
0.15625×10^{-3}	0.040695383263897	0.000041039985698
0.0390625×10^{-3}	0.020419185652100	0.000010234893453
$0.009765625 \times 10^{-3}$	0.010217833319680	0.000002557156252

Bảng 3: Bảng sai số cho các chuẩn H_0^2 và chuẩn L^2, ví dụ 2 Forward Euler Method

2.2 Nhận xét

- Dựa vào hình (2) của ví dụ 1 ta thấy bậc hội tụ của sai số theo chuẩn L^2 là bậc 2, chuẩn H_0^2 là bậc 1 (cũng có thể là giữa $1/2$ và 1).
- Dựa vào hình (4) của ví dụ 2 ta thấy bậc hội tụ của sai số theo chuẩn L^2 là bậc 2, chuẩn H_0^2 chính xác là bậc 1.
- Theo 6 hình mô phỏng ở (1) và (3) ta thấy k càng bé thì hình xấp xỉ càng hội tụ về hình chính xác.
- Theo bảng (2) ta thấy sai số theo chuẩn H_0^2 đạt tầm 10^{-3} nếu k đủ bé. Chuẩn theo L^2 đạt tầm 10^{-5} nếu k đủ bé.
- Theo bảng (3) ta thấy sai số theo chuẩn H_0^2 đạt tầm 10^{-2} nếu k đủ bé. Chuẩn theo L^2 đạt tầm 10^{-4} nếu k đủ bé.

3 PHƯƠNG PHÁP BACKWARD EULER

3.1 Mô phỏng kết quả của Phương pháp Backward Euler

3.1.1 *Nội dung phương pháp*

Có thể giải thích từ Backward ở đây là vì ở phương trình (1.6) với $\theta = 1$ thì phần $D_2 U_i^{n+1}$ sẽ biến mất và chỉ còn $D_1 U_i^n$ và bài toán sẽ chuyển về dạng:

$$U^{n+1} - AU^{n+1} = U^n + kF^n \tag{3.1}$$

hay

$$U^{n+1} = GU^n + kGF^n \tag{3.2}$$

với $G = (I - A)^{-1}$.

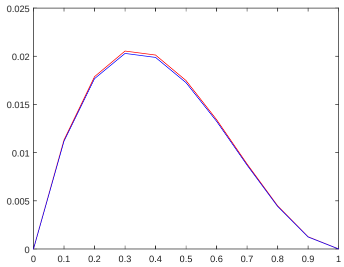
Như vậy ta phải tính U^{n+1} theo U^n bằng việc tính nghịch đảo một ma trận.

Code Matlab cho G ứng với $\theta = 0$:

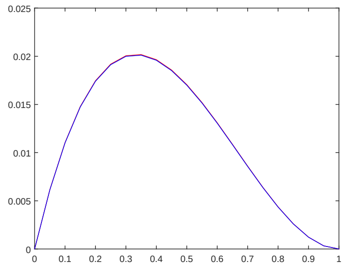
```
%% ---------------Backward Method---------------
G = (I-A)^(-1);
```

3.1.2 *Các kết quả cho ví dụ 1*

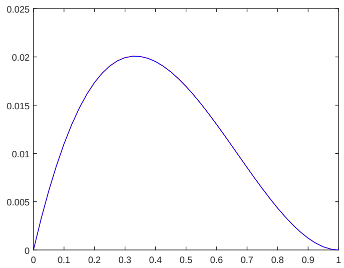
Sau khi thực hành trên Matlab ta có được kết quả xấp xỉ so với kết quả chính thức như sau:



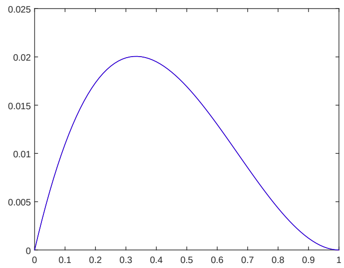
(a) Lần lặp đầu tiên



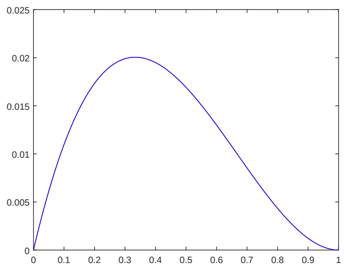
(b) Lần lặp thứ hai



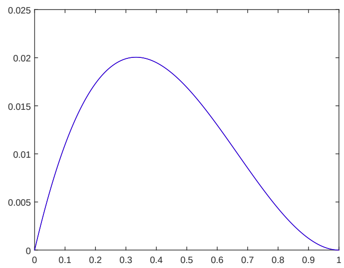
(c) Lần lặp thứ ba



(d) Lần lặp thứ bốn

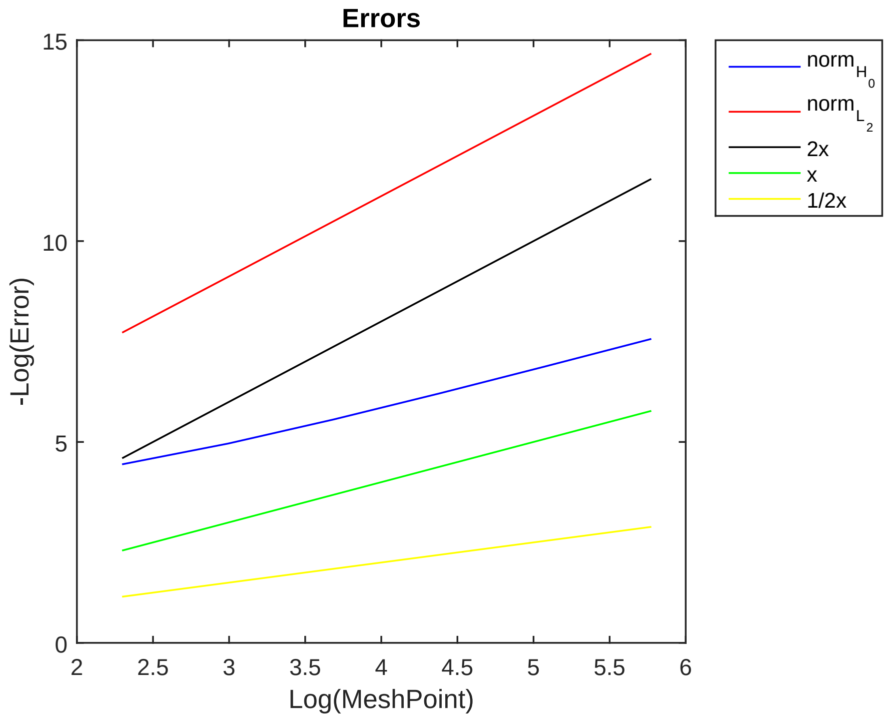


(e) Lần lặp thứ năm



(f) Lần lặp thứ sáu

Hình 5: Ví dụ 1 của Backward Method được mô phỏng qua 6 lần chia lưới mịn hơn



Hình 6: Bậc sai số của Ví dụ 1 của Backward Euler Method

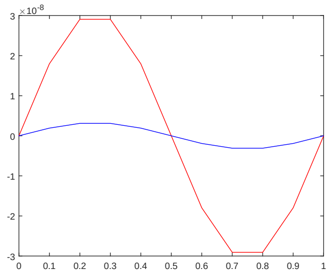
Dưới đây là kết quả tính xấp xỉ sau các lần chia lưới min hơn của Backward Euler Method, ví dụ 1:

Lần lặp	Chuẩn H_0^2	Chuẩn L^2
0.01	0.011706122383061	0.438785776219263 e-03
0.0025	0.006997582933084	0.110223368198195 e-03
0.625×10^{-3}	0.003840320808161	0.027589308396108 e-03
0.15625×10^{-3}	0.002012097568975	0.006899430157105 e-03
0.0390625×10^{-3}	0.001029801762013	0.001724989156575 e-03
$0.009765625 \times 10^{-3}$	0.000520903940181	0.000431255473675 e-03

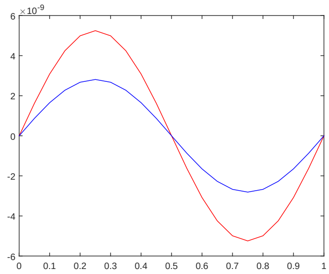
Bảng 4: Bảng sai số cho các chuẩn H_0^2 và chuẩn L^2, ví dụ 1 Backward Euler Method

3.1.3 *Các kết quả cho ví dụ 2*

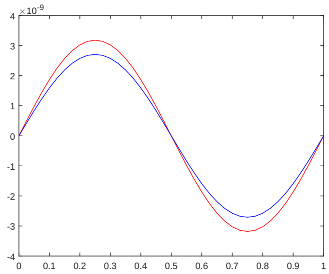
Sau khi thực hành trên Matlab ta có được kết quả xấp xỉ so với kết quả chính thức như sau:



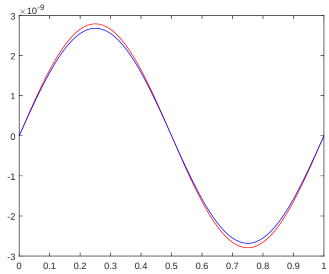
(a) Lần lặp đầu tiên



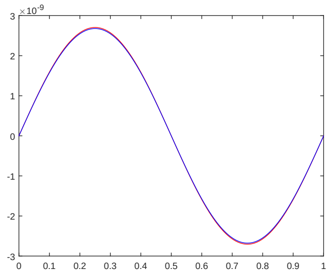
(b) Lần lặp thứ hai



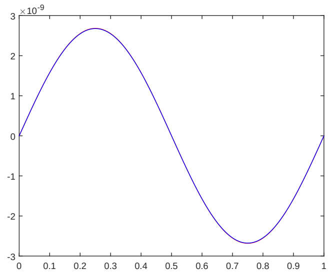
(c) Lần lặp thứ ba



(d) Lần lặp thứ bốn

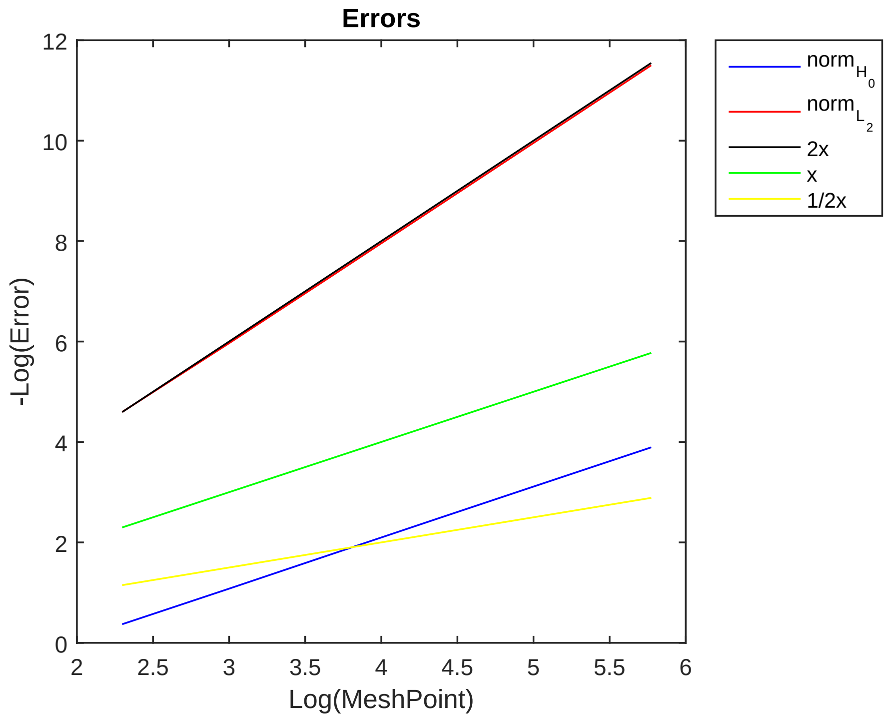


(e) Lần lặp thứ năm



(f) Lần lặp thứ sáu

Hình 7: Ví dụ 2 của Backward Euler Method được mô phỏng qua 6 lần chia lưới mịn hơn



Hình 8: Bậc sai số của Ví dụ 2 của Backward Euler Method

Dưới đây là kết quả tính xấp xỉ sau các lần chia lưới min hơn của Backward Euler Method, ví dụ 2:

Lần lặp	Chuẩn H_0^2	Chuẩn L^2
0.01	0.686676309981495	0.009984858466287
0.0025	0.341430155397487	0.002584983109759
0.625×10^{-3}	0.168337016528976	0.000652389590770
0.15625×10^{-3}	0.083215372195986	0.000163492067689
0.0390625×10^{-3}	0.041274681872285	0.000040897851827
$0.009765625 \times 10^{-3}$	0.020495943373668	0.000010226012985

Bảng 5: Bảng sai số cho các chuẩn H_0^2 và chuẩn L^2, ví dụ 2 Backward Euler Method

3.2 Nhận xét

- Dựa vào hình (6) của ví dụ 1 ta thấy bậc hội tụ của sai số theo chuẩn L^2 là bậc 2, chuẩn H_0^2 là bậc 1.
- Dựa vào hình (8) của ví dụ 2 ta thấy bậc hội tụ của sai số theo chuẩn L^2 là bậc 2, chuẩn H_0^2 cũng là bậc 1.
- Theo 6 hình mô phỏng ở (5) và (7) ta thấy k càng bé thì hình xấp xỉ càng hội tụ về hình chính xác.
- Theo bảng (4) ta thấy sai số theo chuẩn H_0^2 đạt tầm 10^{-3} nếu k đủ bé. Chuẩn theo L^2 đạt tầm 10^{-5} nếu k đủ bé.
- Theo bảng (5) ta thấy sai số theo chuẩn H_0^2 đạt tầm 10^{-2} nếu k đủ bé. Chuẩn theo L^2 đạt tầm 10^{-4} nếu k đủ bé.

4 PHƯƠNG PHÁP CRACK – NICOLSON

4.1 Mô phỏng kết quả của Phương pháp Crack – Nicolson

4.1.1 *Nội dung phương pháp*

Với $\theta = \frac{1}{2}$ phương trình (1.6) sẽ chuyển về dạng:

$$\left(I - \frac{1}{2}A\right) U^{n+1} = \left(I + \frac{1}{2}A\right) U^n + kF^n \tag{4.1}$$

hay

$$U^{n+1} = GU^n + kG_1^{-1}F^n \tag{4.2}$$

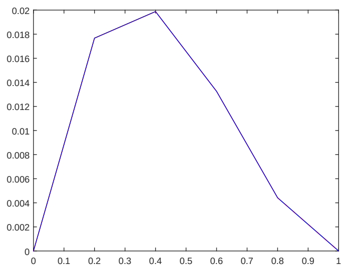
với $G_1 = \left(I - \frac{1}{2}A\right)$ và $G_2 = \left(I + \frac{1}{2}A\right)$ và $G = G_1^{-1}G_2$.

Code Matlab cho $G,\ G_1,\ G_2$ ứng với $\theta = \frac{1}{2}$:

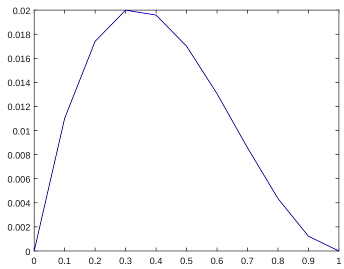
```
%% ---------------Crack_Nicolson---------------
G1 = I - 0.5*A;
G2 = I + 0.5*A;
G = (G1^(-1))*(G2);
```

4.1.2 *Các kết quả cho ví dụ 1*

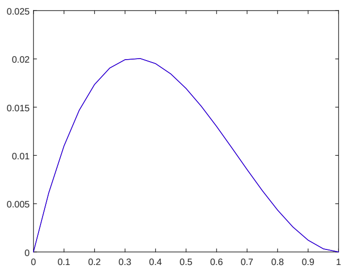
Sau khi thực hành trên Matlab ta có được kết quả xấp xỉ so với kết quả chính thức như sau:



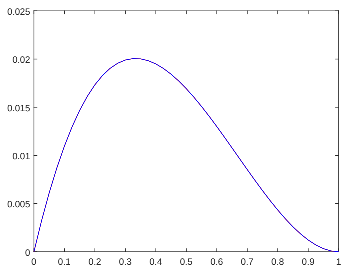
(a) Lần lặp đầu tiên



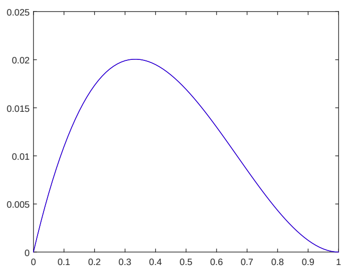
(b) Lần lặp thứ hai



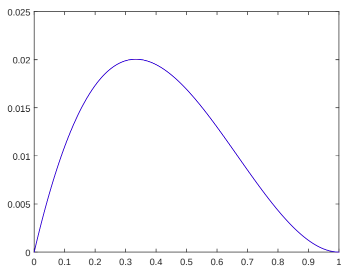
(c) Lần lặp thứ ba



(d) Lần lặp thứ bốn

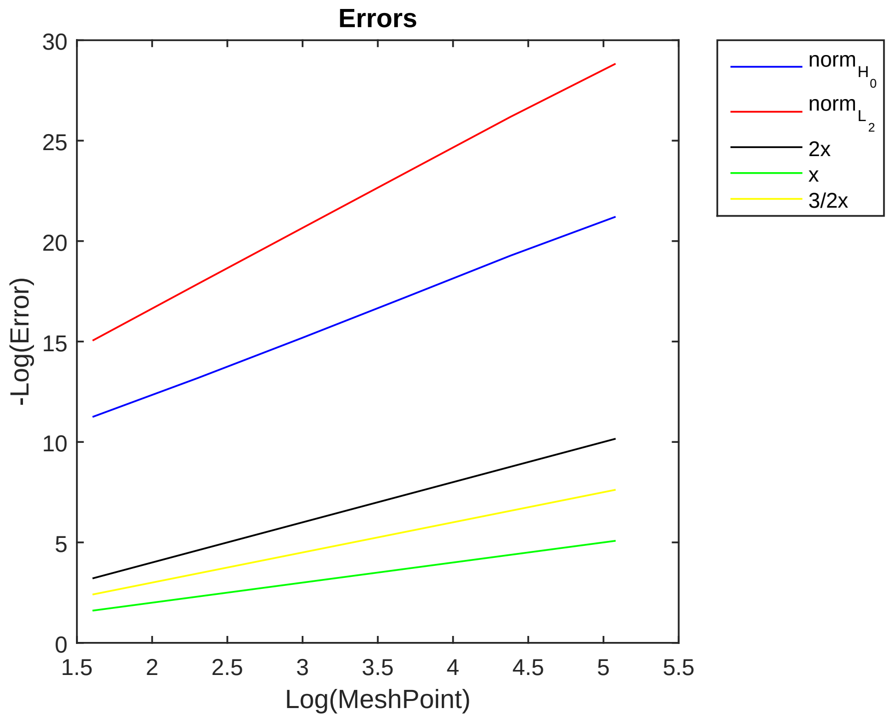


(e) Lần lặp thứ năm



(f) Lần lặp thứ sáu

Hình 9: Ví dụ 1 của Crack - Nicolson Method được mô phỏng qua 6 lần chia lưới mịn hơn



Hình 10: Bậc sai số của Ví dụ 1 của Crack - Nicolson Method

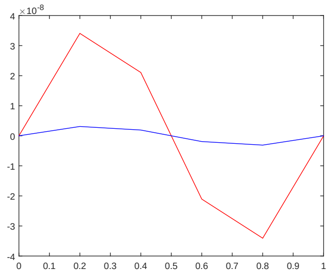
Dưới đây là kết quả tính xấp xỉ sau các lần chia lưới min hơn của Crack - Nicolson Method, ví dụ 1:

Lần lặp	Chuẩn H_0^2	Chuẩn L^2
0.01	0.128580018438350 e-04	0.285873062990016 e-06
0.0025	0.018879466171753 e-04	0.017471480204806 e-06
0.625×10^{-3}	0.002562144057480 e-04	0.001085789840178 e-06
0.15625×10^{-3} e-04	0.000333038501291 e-04	0.000067768238854 e-06
0.0390625×10^{-3}	0.000042523373207 e-04	0.000004244210287 e-06
$0.009765625 \times 10^{-3}$	0.000006223653388 e-04	0.000000307338789 e-06

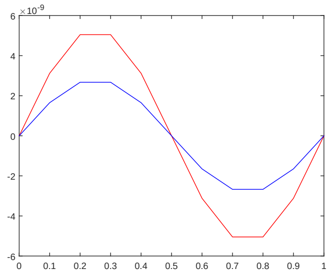
Bảng 6: Bảng sai số cho các chuẩn H_0^2 và chuẩn L^2, ví dụ 1 Crank - Nicolson Method

4.1.3 *Các kết quả cho ví dụ 2*

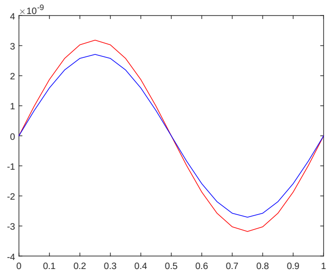
Sau khi thực hành trên Matlab ta có được kết quả xấp xỉ so với kết quả chính thức như sau:



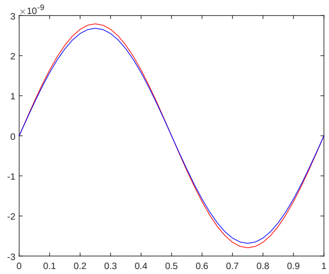
(a) Lần lặp đầu tiên



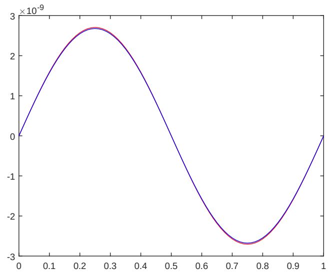
(b) Lần lặp thứ hai



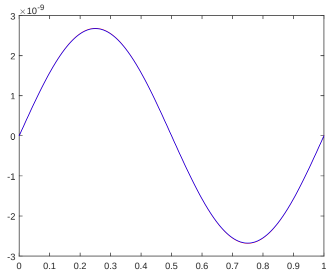
(c) Lần lặp thứ ba



(d) Lần lặp thứ bốn

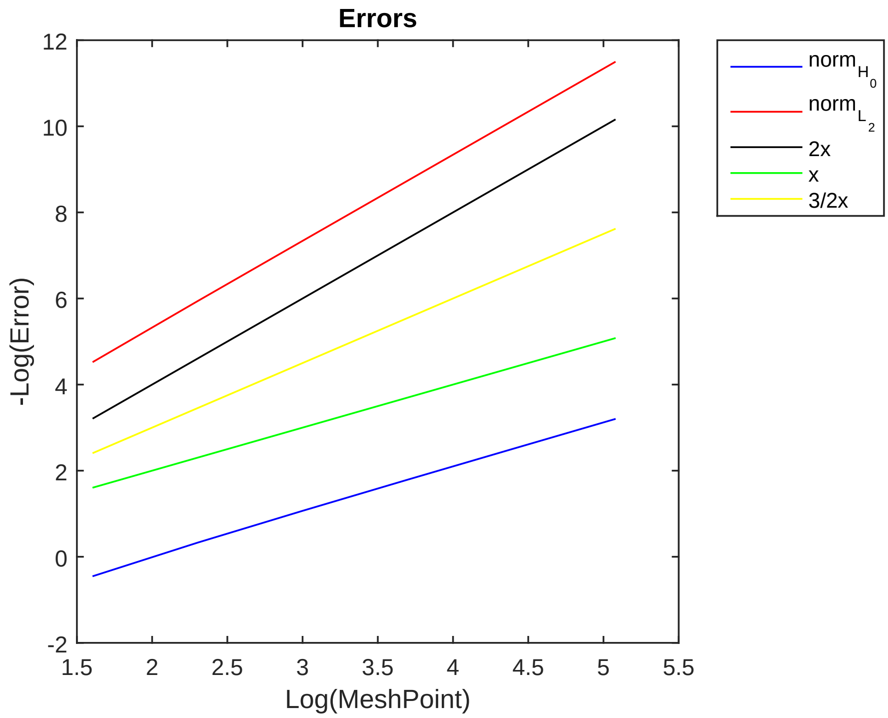


(e) Lần lặp thứ năm



(f) Lần lặp thứ sáu

Hình 11: Ví dụ 2 của Crack - Nicolson Method được mô phỏng qua 4 lần chia lưới mịn hơn



Hình 12: Bậc sai số của Ví dụ 2 của Crack - Nicolson Method

Dưới đây là kết quả tính xấp xỉ sau các lần chia lưới min hơn của Crack - Nicolson Method, ví dụ 2:

Lần lặp	Chuẩn H_0^2	Chuẩn L^2
0.01	1.564605327502403	0.010764349696421
0.0025	0.720180006532168	0.002633452713550
0.625×10^{-3}	0.345609374050834	0.000655417684385
0.15625×10^{-3}	0.168506506805770	0.000163681316257
0.0390625×10^{-3}	0.082869489313216	0.000040909679778
$0.009765625 \times 10^{-3}$	0.040863849760776	0.000010226752153

Bảng 7: Bảng sai số cho các chuẩn H_0^2 và chuẩn L^2, ví dụ 2 Crank - Nicolson Method

4.2 Nhận xét

- Dựa vào hình (10) của ví dụ 1 ta thấy bậc hội tụ của sai số theo chuẩn L^2 lớn hơn bậc 2, chuẩn H_0^2 là bậc $3/2$.
- Dựa vào hình (12) của ví dụ 2 ta thấy bậc hội tụ của sai số theo chuẩn L^2 là bậc 2, chuẩn H_0^2 cũng là bậc $3/2$.
- Theo 6 hình mô phỏng ở (9) và (11) ta thấy k càng bé thì hình xấp xỉ càng hội tụ về hình chính xác.
- Theo bảng (6) ta thấy sai số theo chuẩn H_0^2 đạt tầm 10^{-6} nếu k đủ bé. Chuẩn theo L^2 đạt tầm 10^{-9} nếu k đủ bé.
- Theo bảng (7) ta thấy sai số theo chuẩn H_0^2 đạt tầm 10^{-2} nếu k đủ bé. Chuẩn theo L^2 đạt tầm 10^{-3} nếu k đủ bé.

5 PHƯƠNG PHÁP THETA ỨNG VỚI $\theta = 1/3$

5.1 Mô phỏng kết quả của Phương pháp Theta ứng với $\theta = 1/3$

5.1.1 *Nội dung phương pháp*

Ứng với $\theta = 1/3$ thì phương trình (1.6) sẽ chuyển về dạng:

$$\left(I - \frac{1}{3}A\right) U^{n+1} = \left(I + \frac{2}{3}A\right) U^n + kF^n \tag{5.1}$$

hay

$$U^{n+1} = GU^n + kG_1^{-1}F^n \tag{5.2}$$

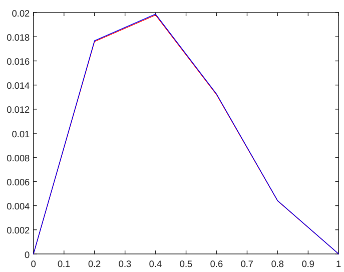
với $G_1 = \left(I - \frac{1}{3}A\right)$ và $G_2 = \left(I + \frac{2}{3}A\right)$ và $G = G_1^{-1}G_2$.

Code Matlab cho G ứng với $\theta = 1/3$:

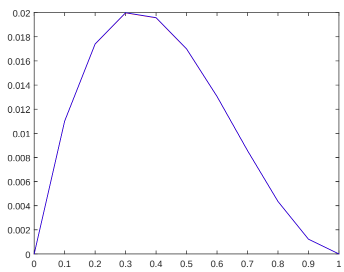
```
theta = 1/3;
%% ---------------Theta methods---------------
G1 = I - theta*A;
G2 = I + (1-theta)*A;
G = (G1^(-1))*(G2);
```

5.1.2 *Các kết quả cho ví dụ 1*

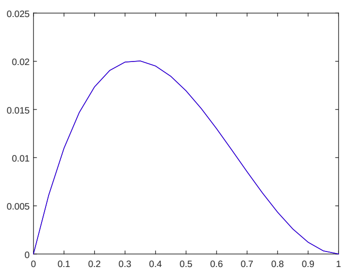
Sau khi thực hành trên Matlab ta có được kết quả xấp xỉ so với kết quả chính thức như sau:



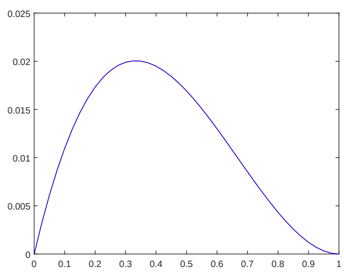
(a) Lần lặp đầu tiên



(b) Lần lặp thứ hai

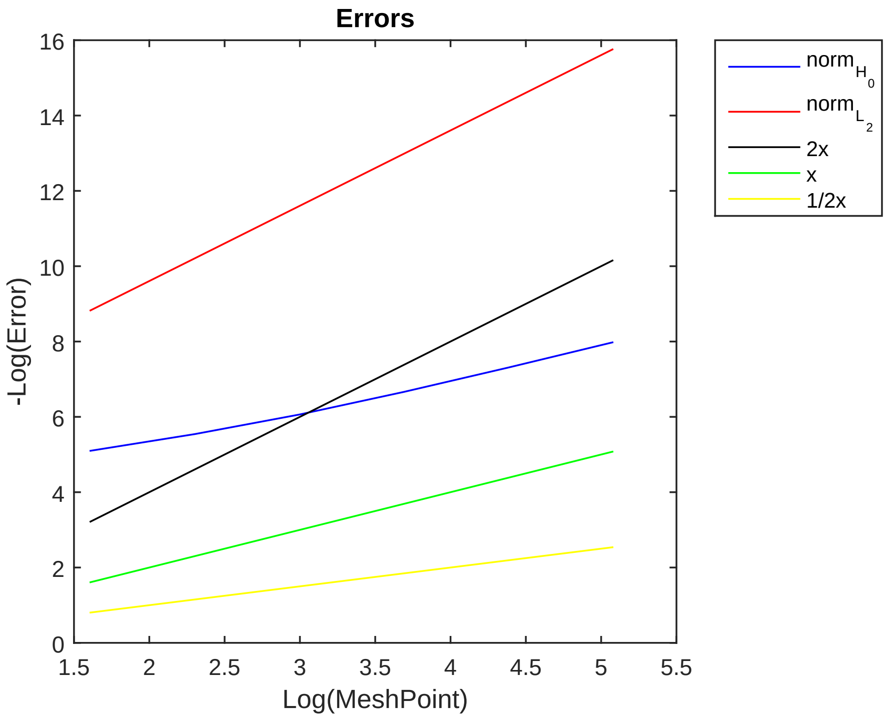


(c) Lần lặp thứ ba



(d) Lần lặp thứ bốn

Hình 13: Ví dụ 1 của θ $1/3$ Method được mô phỏng qua 6 lần chia lưới mịn hơn



Hình 14: Bậc sai số của Ví dụ 1 của θ $1/3$ Method

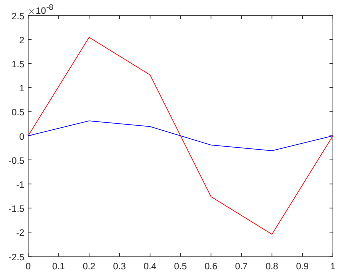
Dưới đây là kết quả tính xấp xỉ sau các lần chia lưới min hơn của θ $1/3$ Method, ví dụ 1:

Lần lập	Chuẩn H_0^2	Chuẩn L^2
0.01	0.006099359843553	0.146782938359894 e-03
0.0025	0.003915721254173	0.036793146738984 e-03
0.625×10^{-3}	0.002332166084042	0.009200004575935 e-03
0.15625×10^{-3}	0.001279423544489	0.002300038093731 e-03
0.0390625×10^{-3}	0.000670441981759	0.000575010720124 e-03
$0.009765625 \times 10^{-3}$	0.000343185451731	0.000143752731160 e-03

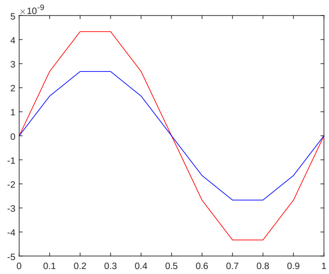
Bảng 8: Bảng sai số cho các chuẩn H_0^2 và chuẩn L^2, ví dụ 1 θ $1/3$ Method

5.1.3 *Các kết quả cho ví dụ 2*

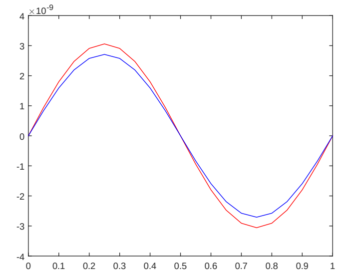
Sau khi thực hành trên Matlab ta có được kết quả xấp xỉ so với kết quả chính thức như sau:



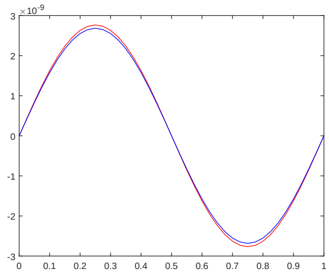
(a) Lần lặp đầu tiên



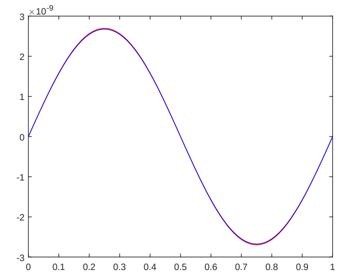
(b) Lần lặp thứ hai



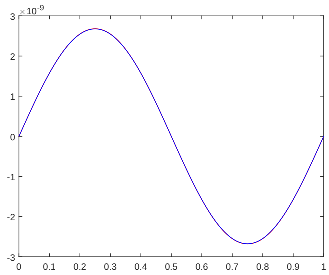
(c) Lần lặp thứ ba



(d) Lần lặp thứ bốn

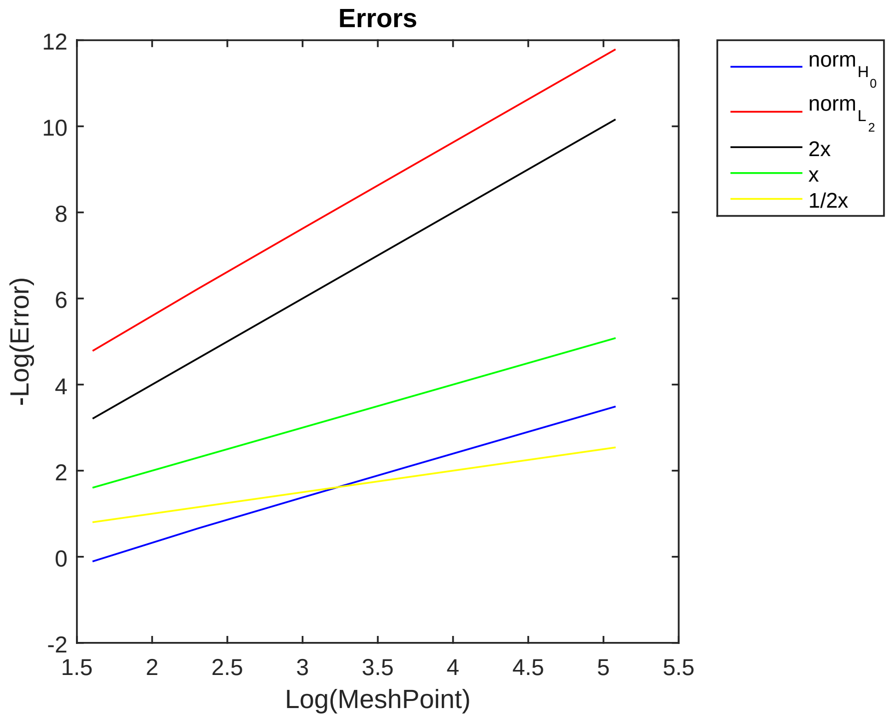


(e) Lần lặp thứ năm



(f) Lần lặp thứ sáu

Hình 15: Ví dụ 2 của θ $1/3$ Method được mô phỏng qua 6 lần chia lưới mịn hơn



Hình 16: Bậc sai số của Ví dụ 2 của θ $1/3$ Method

Dưới đây là kết quả tính xấp xỉ sau các lần chia lưới min hơn của θ $1/3$ Method, ví dụ 2:

k	Chuẩn H_0^2	Chuẩn L^2
0.01	1.108390655040441	0.008287804912336
00025	0.518668694174075	0.001988046887769
0.625×10^{-3}	0.253240616333336	0.000492367206631
0.15625×10^{-3}	0.124880768661539	0.000122811148186
0.0390625×10^{-3}	0.061845329004146	0.000030685394804
$0.009765625 \times 10^{-3}$	0.030648675289198	0.000007670261315

Bảng 9: Bảng sai số cho các chuẩn H_0^2 và chuẩn L^2, ví dụ 2 θ $1/3$ Method

5.2 Nhận xét

- Dựa vào hình (14) của ví dụ 1 ta thấy bậc hội tụ của sai số theo chuẩn L^2 lớn bậc 2, chuẩn H_0^2 là bậc nằm giữa bậc 1 và bậc $1/2$.
- Dựa vào hình (16) của ví dụ 2 ta thấy bậc hội tụ của sai số theo chuẩn L^2 là bậc 2, chuẩn H_0^2 cũng là bậc 1.
- Theo 6 hình mô phỏng ở (13) và (15) ta thấy k càng bé thì hình xấp xỉ càng hội tụ về hình chính xác.
- Theo bảng (8) ta thấy sai số theo chuẩn H_0^2 đạt tầm 10^{-3} nếu k đủ bé. Chuẩn theo L^2 đạt tầm 10^{-5} nếu k đủ bé.
- Theo bảng (9) ta thấy sai số theo chuẩn H_0^2 đạt tầm 10^{-2} nếu k đủ bé. Chuẩn theo L^2 đạt tầm 10^{-3} nếu k đủ bé.

6 PHƯƠNG PHÁP THETA ỨNG VỚI $\theta = 2/3$

6.1 Mô phỏng kết quả của Phương pháp Theta ứng với $\theta = 2/3$

6.1.1 *Nội dung phương pháp*

Với $\theta = \dfrac{2}{3}$ thì ta có $\theta \in \left(\dfrac{1}{2}, 1\right)$

$$\left(I - \frac{2}{3}A\right)U^{n+1} = \left(I + \frac{1}{3}A\right)U^n + AU^n + kF^n \tag{6.1}$$

hay

$$U^{n+1} = GU^n + kG_1^{-1}F^n \tag{6.2}$$

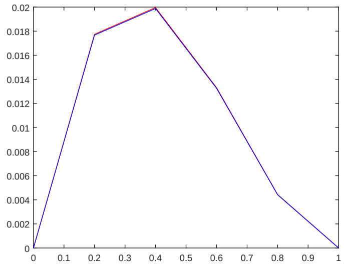
với $G_1 = \left(I - \dfrac{2}{3}A\right)$, $G_2 = \left(I + \dfrac{1}{3}A\right)$, $G = G_1^{-1}G_2$.

Code Matlab cho G ứng với $\theta = \dfrac{2}{3}$:

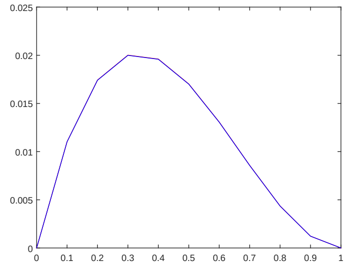
```
theta = 2/3;
%% ---------------Forward Method---------------
G1 = I - theta*A;
G2 = I + (1-theta)*A;
G = (G1^(-1))*(G2);
```

6.1.2 *Các kết quả cho ví dụ 1*

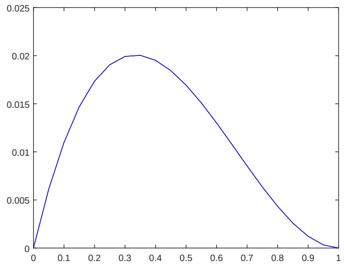
Sau khi thực hành trên Matlab ta có được kết quả xấp xỉ so với kết quả chính thức như sau:



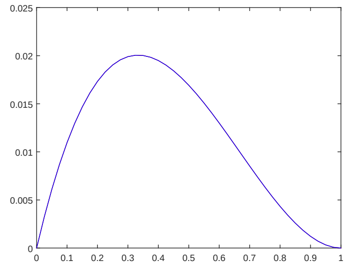
(a) Lần lặp đầu tiên



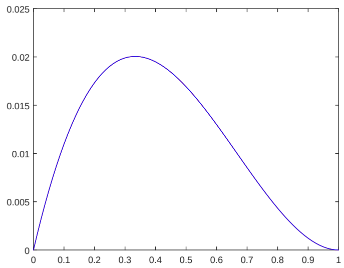
(b) Lần lặp thứ hai



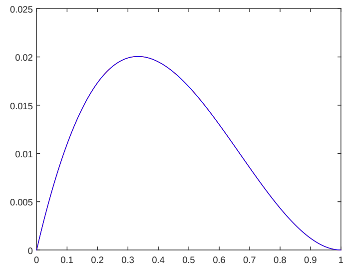
(c) Lần lặp thứ ba



(d) Lần lặp thứ bốn

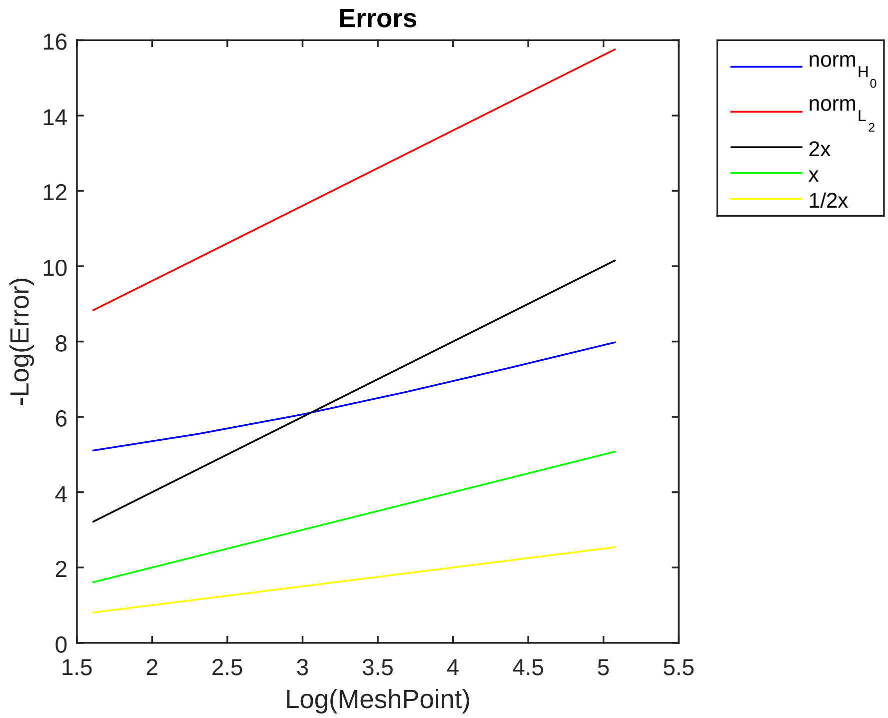


(e) Lần lặp thứ ba



(f) Lần lặp thứ bốn

Hình 17: Ví dụ 1 của θ $2/_3$ Method được mô phỏng qua 6 lần chia lưới mịn hơn



Hình 18: Bậc sai số của Ví dụ 1 của θ $2/3$ Method

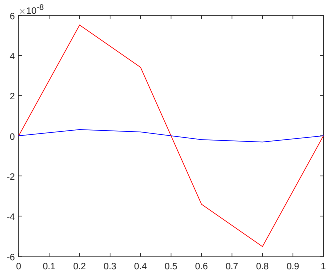
Dưới đây là kết quả tính xấp xỉ sau các lần chia lưới min hơn của θ $2/3$ Method, ví dụ 1:

Lần lặp	Chuẩn H_0^2	Chuẩn L^2
0.01	0.006050945678498	0.145760739794344 e-03
0.0025	0.003908417286471	0.036730173706002 e-03
0.625×10^{-3}	0.002331173342142	0.009196081603004 e-03
0.15625×10^{-3}	0.001279295105803	0.002299793099011 e-03
0.0390625×10^{-3}	0.000670425688198	0.000574995409813 e-03
$0.009765625 \times 10^{-3}$	0.000343183390733	0.000143751767513 e-03

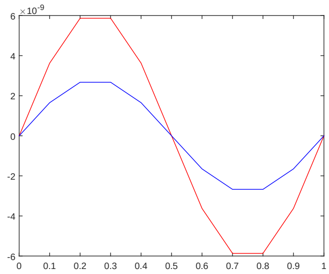
Bảng 10: Bảng sai số cho các chuẩn H_0^2 và chuẩn L^2, ví dụ 1 θ $2/3$ Method

6.1.3 *Các kết quả cho ví dụ 2*

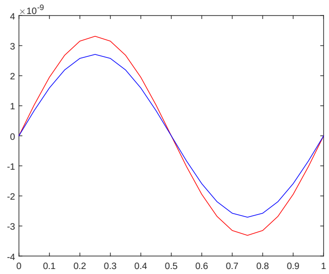
Sau khi thực hành trên Matlab ta có được kết quả xấp xỉ so với kết quả chính thức như sau:



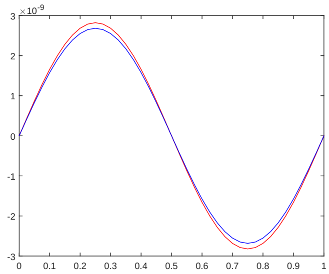
(a) Lần lặp đầu tiên



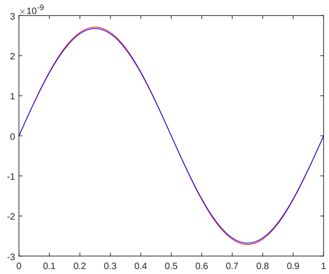
(b) Lần lặp thứ hai



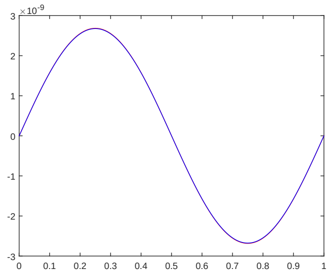
(c) Lần lặp thứ ba



(d) Lần lặp thứ bốn

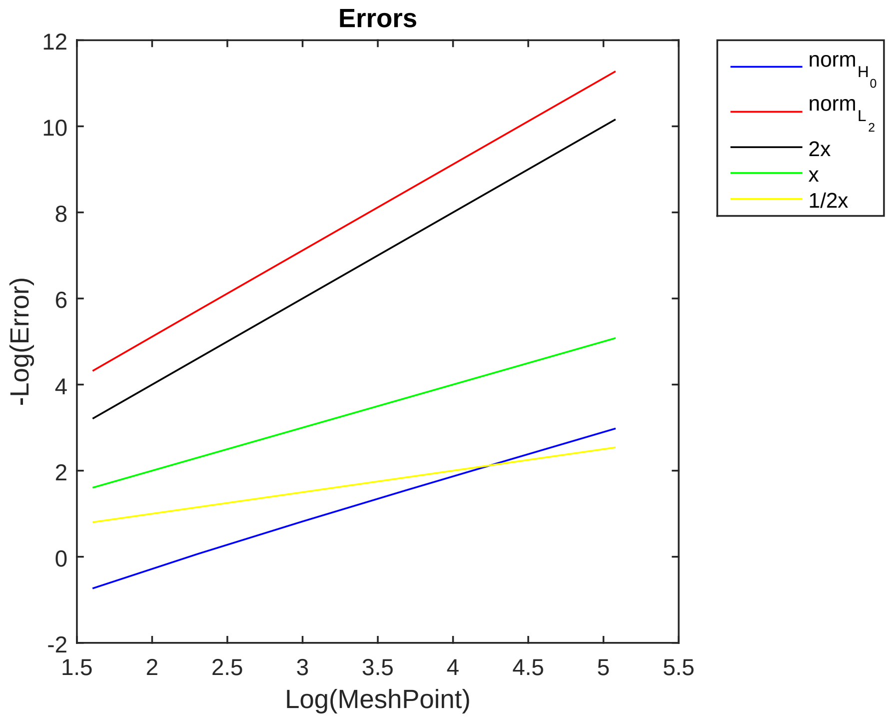


(e) Lần lặp thứ năm



(f) Lần lặp thứ sáu

Hình 19: Ví dụ 2 của Forward Method được mô phỏng qua 4 lần chia lưới mịn hơn



Hình 20: Bậc sai số của Ví dụ 2 của Forward Method

Dưới đây là kết quả tính xấp xỉ sau các lần chia lưới min hơn của Forward Method, ví dụ 2:

k	Chuẩn H_0^2	Chuẩn L^2
0.01	2.074045047444489	0.013204224321147
00025	0.937301033463379	0.003276318790990
0.625×10^{-3}	0.442128658389419	0.000818305039311
0.15625×10^{-3}	0.213154058198032	0.000204541216917
0.0390625×10^{-3}	0.104101029005825	0.000051133321115
$0.009765625 \times 10^{-3}$	0.051078860486658	0.000012783202011

Bảng 11: Bảng sai số cho các chuẩn H_0^2 và chuẩn L^2, ví dụ 2 θ $2/3$ Method

6.2 Nhận xét

- Dựa vào hình (18) của ví dụ 1 ta thấy bậc hội tụ của sai số theo chuẩn L^2 lớn bậc 2, chuẩn H_0^2 là bậc nằm giữa bậc 1 và bậc $1/2$.
- Dựa vào hình (20) của ví dụ 2 ta thấy bậc hội tụ của sai số theo chuẩn L^2 là bậc 2, chuẩn H_0^2 cũng là bậc 1.
- Theo 6 hình mô phỏng ở (17) và (19) ta thấy k càng bé thì hình xấp xỉ càng hội tụ về hình chính xác.
- Theo bảng (10) ta thấy sai số theo chuẩn H_0^2 đạt tầm 10^{-3} nếu k đủ bé. Chuẩn theo L^2 đạt tầm 10^{-6} nếu k đủ bé.
- Theo bảng (11) ta thấy sai số theo chuẩn H_0^2 đạt tầm 10^{-1} nếu k đủ bé. Chuẩn theo L^2 đạt tầm 10^{-3} nếu k đủ bé.

7 TỔNG KẾT

SỰ SAI SỐ QUA CÁC BƯỚC LẶP :

1. Qua các bảng số liệu về sai số ở các bảng (2), (4), (6), (8) và (10) ta thấy
   - Phương pháp Crank - Nicolson ứng với $\theta = \frac{1}{2}$ cho kết quả xấp xỉ tốt nhất cho cả 2 chuẩn, sai số 10^{-6} cho chuẩn Sobolev H_0^2 và 10^{-9} cho chuẩn L^2.
   - Giữa Forward Euler $\theta = 0$ và Backward Euler$\theta = 1$ thì tương đương nhau (Backward tốt hơn chút đỉnh) như Backward sẽ yêu cầu thời gian chạy code lâu hơn Forward vì phải tính nghịch đảo của một ma trận thưa thớt (sparse matrix) $I - A$.
   - Việc xét thêm trường hợp $\theta = \frac{1}{3}$ và $\theta = \frac{2}{3}$ cho thấy 2 phương pháp này cho kết quả xấp xỉ tốt hơn khá nhiều so với Backward và Forward Euler. Tuy nhiên do phải thực hiện cả việc tính nghịch đảo của một ma trận thưa thớt và nhân hai ma trận thưa thớt nên thời gian tính toán khá lâu.
2. Qua các bảng số liệu về sai số ở các bảng (3), (5), (7), (9) và (11) ta thấy
   - Cả ba phương pháp Forward Euler, Backward Euler và Crank - Nicolson đều cho kết quả xấp xỉ gần như nhau cho cả 2 chuẩn, sai số 10^{-2} cho chuẩn Sobolev H_0^2 và 10^{-4} cho chuẩn L^2. Tỷ lệ này khá bé so với ví dụ 1 vì ví dụ 2 có thêm yếu tố lượng giác làm cho bài toán phức tạp hơn khá nhiều.
   - Việc xét thêm trường hợp $\theta = \frac{1}{3}$ và $\theta = \frac{2}{3}$ cho thấy 2 phương pháp này cho kết quả xấp xỉ tốt gần bằng so với các phương pháp trên. Tuy nhiên do phải thực hiện cả việc tính nghịch đảo của một ma trận thưa thớt và nhân hai ma trận thưa thớt nên thời gian tính toán khá lâu.

BẬC HỘI TỤ :

1. Qua các đồ thị (2), (6), (10), (14) và (18) ta thấy
   - Bậc hội tụ theo phương pháp Forward Euler, Backward Euler của chuẩn H_0^2 đều gần bậc 1, nằm trong khoảng bậc $1/2$ và bậc 1. Trong khi đó với phương pháp Crank - Nicolson thì ta có chuẩn H_0^2 sẽ hội theo bậc $3/2$ (đúng theo lý thuyết chứng minh được.
   - Bậc hội tụ theo phương pháp Forward Euler, Backward Euler, Crank - Nicolson của chuẩn L^2 đều là bậc 2.
2. Qua các đồ thị (4), (8), (12), (16) và (20) ta thấy
   - Bậc hội tụ theo cả 2 phương pháp θ ứng với $1/3$ và $2/3$ của chuẩn H_0^2 đều là bậc 1 (chính xác).
   - Bậc hội tụ theo cả 2 phương pháp θ ứng với $1/3$ và $2/3$ của chuẩn L^2 đều là bậc 2.

TỔNG KẾT CÁC BẢNG SAI SỐ : Chỉ cho các giá trị k là 0.01, 0.15625×10^{-3} và $0.009765625 \times 10^{-3}$.

Ví dụ 1 theo chuẩn H_0^2:

k	0.01	0.15625×10^{-3}	$0.009765625 \times 10^{-3}$
Forward Euler	0.018502231608067	0.003841827380480	0.001029558806433
Backward Euler	0.011706122383061	0.002012097568975	0.000520903940181
Crank - Nicolson	0.128580018438350 e-04	0.000333038501291 e-04	0.000006223653388 e-04
$\theta = 1/3$	0.006099359843553	0.001279423544489	0.000343185451731
$\theta = 1/3$	0.006050945678498	0.001279295105803	0.000343183390733

Ví dụ 1 theo chuẩn L^2:

k	0.01	0.15625×10^{-3}	$0.009765625 \times 10^{-3}$
Forward Euler	0.441231466935645 e-03	0.006900331734321 e-03	0.000431259053170 e-03
Backward Euler	0.438785776219263 e-03	0.006899430157105 e-03	0.000431255473675 e-03
Crank - Nicolson	0.285873062990016 e-06	0.000067768238854 e-06	0.000000307338789 e-06
$\theta = 1/3$	0.146782938359894 e-03	0.002300038093731 e-03	0.000143752731160 e-03
$\theta = 1/3$	0.145760739794344 e-03	0.002299793099011 e-03	0.000143751767513 e-03

Ví dụ 2 theo chuẩn H_0^2:

k	0.01	0.15625×10^{-3}	$0.009765625 \times 10^{-3}$
Forward Euler	0.357242828023685	0.040695383263897	0.010217833319680
Backward Euler	0.686676309981495	0.083215372195986	0.020495943373668
Crank - Nicolson	1.564605327502403	0.168506506805770	0.040863849760776
$\theta = 1/3$	1.108390655040441	0.124880768661539	0.030648675289198
$\theta = 1/3$	2.074045047444489	0.213154058198032	0.051078860486658

Ví dụ 2 theo chuẩn L^2:

k	0.01	0.15625×10^{-3}	$0.009765625 \times 10^{-3}$
Forward Euler	0.003219206545401	0.000041039985698	0.000002557156252
Backward Euler	0.009984858466287	0.000163492067689	0.000010226012985
Crank - Nicolson	0.010764349696421	0.000163681316257	0.000010226752153
$\theta = 1/3$	0.008287804912336	0.000122811148186	0.000007670261315
$\theta = 1/3$	0.013204224321147	0.000204541216917	0.000012783202011

TÀI LIỆU

[1] Finite Different Methods in 1D, Le Anh Ha (Lecture Note). Khoa Toán - Tin, Đại học Khoa học tự nhiên TPHCM, 2017.

[2] Finite Different Methods for Ordinary and Partial Differential Equations, Randall J.LeVeque. [Steady - Stable and Time - Dependent Problems]. University of Washington, Seattle, Washington.