ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC BÁCH KHOA

BÁO CÁO BÀI TẬP LỚN

PHƯƠNG PHÁP TÍNH

ĐỀ TÀI

ĐO DỮ LIỆU HOẠT ĐỘNG CỦA CPU USAGE TRONG MỘT KHOẢNG THỜI GIAN, SỬ DỤNG CÁC PHƯƠNG PHÁP NỘI SUY ĐỂ MÔ TẢ DỮ LIỆU VÀ ĐÁNH GIÁ SAI SỐ

Giảng viên hướng dẫn:	VÕ TRẦN AN
Lớp:	L07
Nhóm:	10

TP. HỒ CHÍ MINH, THÁNG 5 NĂM 2024

STT	Họ và Tên	MSSV	Đóng góp	Nội dung
1	Nguyễn Thanh Phong	2312626	100%	Cơ sở lý thuyết, code Lagrange
2	Nguyễn Thanh Tùng	2213874	100%	Cơ sở lý thuyết, code Spline bậc 3
3	Nguyễn Tiến Phát	2014088	0%	Không tham gia làm việc
4	Nguyễn Trọng Trí	2313613	100%	Cơ sở lý thuyết, code Lagrange
5	Nguyễn Vĩnh Anh Kiên	2311739	100%	Cơ sở lý thuyết, code Newton
6	Nguyễn Xuân Đức	2310792	100%	Cơ sở lý thuyết, code Spline bậc 3
7	Phạm Công Vinh	2313927	100%	Soạn thảo latex
8	Phạm Hoàng Nam	2312187	0%	Không tham gia làm việc

LỜI NÓI ĐẦU

Trong cuộc cách mạng số hóa hiện nay, CPU (Central Processing Unit) chính là trái tim của các thiết bị điện tử, từ máy tính cá nhân đến các hệ thống máy chủ phức tạp. Việc hiểu rõ về hoạt động của CPU và cách nó tương tác với hệ thống là vô cùng quan trọng.

Trong bài cáo này, chúng tôi sẽ ứng dụng các phương pháp nội suy trong việc mô tả dữ liệu và đánh giá sai số của quá trình đo lường dữ liệu hoạt động của CPU usage trong một khoảng thời gian.

Từ việc áp dụng các phương pháp nội suy vào đo lường và phân tích dữ liệu CPU usage, chúng ta có thể xây dựng được cái nhìn toàn diện về cách CPU hoạt động. Đồng thời, việc đánh giá sai số cũng giúp chúng ta hiểu rõ hơn về độ chính xác của dữ liệu đo lường, từ đó chúng ta có thể dễ dàng nắm bắt được bức tranh tổng thể về hiệu suất hoạt động của CPU và hệ thống, phòng tránh các vấn đề tiềm ẩn, đảm bảo rằng chúng hoạt động ổn định và hiệu quả trong môi trường số hóa ngày nay.

MỤC LỤC

CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1 CPU usage

1.1.1 Khái niệm về CPU usage

CPU usage là một chỉ số đo lường mức độ sử dụng của bộ xử lý trung tâm (CPU) trong một thời gian nhất định. Đây là một yếu tố quan trọng để đo lường hiệu suất của máy tính hoặc hệ thống. Mức độ CPU usage thường được biểu diễn dưới dạng phần trăm, với 100% là mức tối đa mà CPU có thể làm việc.

Mỗi CPU đều có một khả năng giới hạn để thực hiện các lệnh và chạy các chương trình khác nhau. CPU càng mạnh mẽ, bạn càng có thể thực hiện đồng thời nhiều nhiệm vụ mà không gặp trở ngại. Tuy nhiên, dù CPU mạnh mẽ đến đâu, mỗi vi chip cuối cùng cũng sẽ đạt đến công suất tối đa và bắt đầu hoạt động chậm lại [1].

Ta có thể hiểu CPU usage giống như độ "bận rộn" của CPU. Khi bạn thực hiện các hoạt động trên máy tính như mở các ứng dụng, duyệt web hoặc chơi game, CPU sẽ phải xử lý các yêu cầu này. Khi bạn không chạy nhiều ứng dụng, CPU usage trở nên thấp, mọi thứ sẽ chạy rất mượt mà. Ngược lại, khi CPU usage càng cao, tức là CPU đang phải làm việc nhiều hơn, bạn có thể nhận thấy thời gian phản hồi tăng lên, máy tính có thể trở nên chậm chạp hoặc đáp ứng kém đi.

1.1.2 Cách tính CPU usage

Để tính toán CPU usage, bạn có thể thực hiện các bước sau:

1. Đo lường tổng thời gian CPU hoạt động: Ghi lại thời điểm bắt đầu và kết thúc của quá trình.
2. Đo lường thời gian CPU không làm việc (chương trình rảnh).
3. Khi CPU phải thực hiện nhiệm vụ, chương trình rảnh được tắt; thời gian CPU làm việc bằng tổng thời gian CPU hoạt động trừ cho thời gian CPU không làm việc.

$$\%CPUusage = \frac{t_{tong} - t_{ranh}}{t_{tong}} \times 100\%$$

1.2 Các phương pháp nội suy

Bạn thường xuyên cần ước lượng giá trị trung gian giữa các điểm dữ liệu cụ thể, và phương pháp phổ biến nhất được sử dụng cho mục đích này là nội suy đa

thức. Hãy nhớ rằng công thức tổng quát cho một đa thức bậc thứ n là:

$$f(x) = a_0 + a_1x + a_2x^2 + ... + a_nx^n$$

Đối với $n + 1$ điểm dữ liệu, có một và chỉ một đa thức bậc n đi qua tất cả các điểm đó. Ví dụ, khi có hai điểm dữ liệu, chúng ta có thể tìm ra một đường thẳng duy nhất (một đa thức bậc nhất) kết nối chúng. Tương tự, với ba điểm dữ liệu, chỉ có một parabol (một đa thức bậc hai) đi qua tất cả ba điểm. Nội suy đa thức bao gồm việc xác định đa thức bậc n duy nhất này mà phù hợp với các điểm dữ liệu đã cho, giúp chúng ta tính toán các giá trị trung gian. Mặc dù chỉ có một đa thức phù hợp với các điểm dữ liệu, có nhiều định dạng toán học khác nhau để biểu diễn đa thức này [2] .

1.2.1 Phương pháp Newton

1.2.1.1 Định nghĩa tỉ sai phân

Trên đoạn $[x_k, x_{k+1}]$ ta định nghĩa đại lượng

$$f[x_k, x_{k+1}] = \frac{y_{k+1} - y_k}{x_{k+1} - x_k}$$

được gọi là tỉ sai phân cấp 1. Tương tự

$$f[x_k, x_{k+1}, x_{k+2}] = \frac{f[x_{k+1}, x_{k+2}] - f[x_k, x_{k+1}]}{x_{k+2} - x_k}$$

được gọi là tỉ sai phân cấp 2. Bằng quy nạp, ta có tỉ phân cấp p

$$f[x_k, x_{k+1}, ..., x_{k+p}] = \frac{f[x_{k+1}, x_{k+2}, ..., x_{k+p}] - f[x_k, x_{k+1}, ...x_{k+p-1}]}{x_{k+p} - x_k} [3]$$

1.2.1.2 Công thức tổng quát

(1) Newton tiến:

$\mathcal{N}_n^{(1)}(x) = y_0 + f[x_0, x_1](x - x_0) + f[x_0, x_1, x_2](x - x_0)(x - x_1) + ... + f[x_0, x_1, ..., x_n](x - x_0)(x - x_1)...(x - x_{n-1})$

(2) Newton lùi:

$\mathcal{N}_n^{(2)}(x) = y_n + f[x_{n-1}, x_n](x - x_n) + f[x_{n-2}, x_{n-1}, x_n](x - x_n)(x - x_{n-1}) + ... + f[x_0, x_1, ..., x_n](x - x_n)(x - x_{n-1})...(x - x_1)$ [3]

1.2.1.3 Trường hợp đặc biệt: Các điểm nút cách đều nhau khoảng h

(a) Định nghĩa sai phân:

Sai phân tiến cấp 1

$$\Delta y_j = y_{j+1} - y_j$$

Sai phân tiến cấp k+1

$$\Delta^{k+1}y_j = \Delta^k y_{j+1} - \Delta^k y_j$$

(b) Công thức Newton tiến: Đặt $q = \dfrac{x-x_0}{h}$

$$\mathcal{N}_n^{(1)}(x) = y_0 + \frac{\Delta y_0}{1!}q + \frac{\Delta^2 y_0}{2!}q(q-1) + ...\frac{\Delta^n y_0}{n!}q(q-n+1)$$

(c) Công thức Newton lùi: Đặt $p = \dfrac{x-x_n}{h}$

$$\mathcal{N}_n^{(2)}(x) = y_n + \frac{\Delta y_{n-1}}{1!}p + \frac{\Delta^2 y_{n-2}}{2!}p(p+1) + ...\frac{\Delta^n y_0}{n!}p(p+1)...(p+n-1)[3]$$

1.2.2 Phương pháp Lagrange

1.2.2.1 Công thức chung

$L_n(x) = \omega(x).\sum_{k=0}^{n}\dfrac{y_k}{D_k}$ với $\omega(x) = (x-x_0)(x-x_1)...(x-x_n)$ và $D_k = \omega'(x_k)(x-x_k)[3]$

1.2.2.2 Trường hợp đặc biệt: Các điểm nút cách đều nhau với bước $h = x_{k+1} - x_k$

Đặt $q = \dfrac{x-x_0}{h}$, khi đó ta có:

$$L_n(x) = \prod_{k=0}^{n}(q-k)\sum_{k=0}^{n}\frac{(-1)^{n-k}y_k}{k!(n-k)!(q-k)}[3]$$

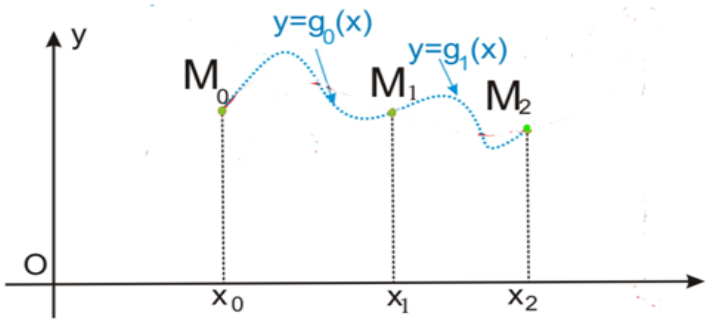
1.2.3 Phương pháp nội suy spline bậc 3

Việc xây dựng một đa thức đi qua các điểm nội suy cho trước trong trường hợp n lớn là rất khó khăn. Biện pháp khắc phục là trên từng đoạn liên tiếp của các cặp điểm nút nội suy ta nối chúng bởi các đường cong đơn giản như đoạn thẳng. Tuy nhiên, khi đó tại các điểm nút hàm sẽ mất tính khả vi. Do đó, phải xây dựng đường cong bằng cách nối các đoạn cong nhỏ lại với nhau sao cho vẫn bảo toàn tính khả vi của hàm. Đường cong như vậy được gọi là đường spline (đường ghép trơn). Các hàm trên các đoạn nhỏ này thường là các đa thức và bậc cao nhất của các đa thức đó gọi là bậc của spline.

1.2.3.1 Định nghĩa 1

Cho $f(x)$ xác định trên đoạn $[a,b]$ và một phép nhân hoạch của nó: $a = x_0 < x_1 < x_2 = b$. Đặt $y_0 = f(x_0)$, $y_1 = f(x_1), y_2 = f(x_2)$. Một spline bậc ba nội suy hàm $f(x)$ trên $[a,b]$ là hàm $g(x)$ thỏa các điều kiện sau:

1. $g(x)$ có đạo hàm đến cấp 2 liên tục trên $[a,b]$.
2. $g(x) = \begin{cases} g_0(x), & x \in [x_0,x_1] \\ g_1(x), & x \in [x_1,x_2] \end{cases}$ ở đây $g_0(x)$, $g_1(x)$ là các đa thức bậc ba.
3. $g(x_0) = f(x_0) = y_0, g(x_1) = f(x_1) = y_1, g(x_2) = f(x_2) = y_2$.



Hình 1.1

1.2.3.2 Định nghĩa 2

Cho $f(x)$ xác định trên đoạn $[a,b]$ và một phép nhân hoạch của nó: $a = x_0 < x_1 < x_2 < ... < x_n = b$. Đặt $y_k = f(x_k)$, $k = 0.....n$. Một spline bậc ba nội suy hàm $f(x)$ trên $[a,b]$ là hàm $g(x)$ thỏa các điều kiện sau:

1. $g(x)$ có đạo hàm đến cấp 2 liên tục trên $[a,b]$.
2. Trên mỗi đoạn $[x_k,x_{k+1}]$, $k = 0.....n-1$, $g(x) = g_k(x)$ là 1 đa thức bậc ba.
3. $g(x_k) = f(x_k) = y_k, \forall k = 0.....n$.

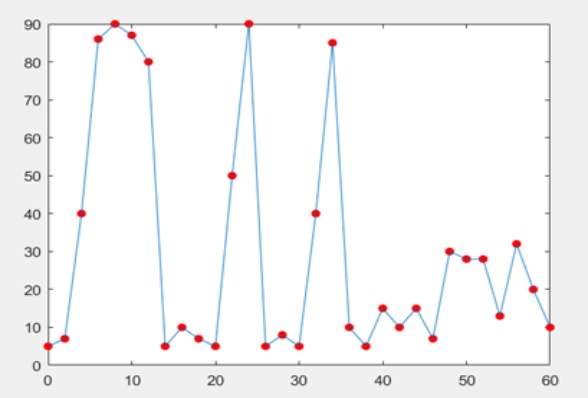
CHƯƠNG 2: THIẾT KẾ GIẢI THUẬT & CODE MATLAB

Bảng 2.1 Dữ liệu đầu vào từ mẫu

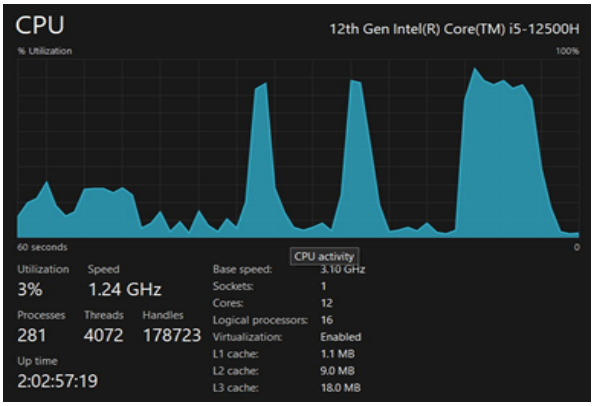
Thời gian hoạt động t	t(s)	0	2	4	6	8	10	12	14	16	18
%CPU usage A(t)	A(t) (%)	5	7	40	86	90	87	80	5	10	7

Thời gian hoạt động t	t(s)	20	22	24	26	28	30	32	34	36	38
%CPU usage A(t)	A(t) (%)	5	50	90	5	8	5	40	85	10	5

Thời gian hoạt động t	t(s)	40	42	44	46	48	50	52	54	56	58	60
%CPU usage A(t)	A(t) (%)	15	10	15	7	30	28	28	13	32	20	10



Hình 2.2 Biên độ hóa mẫu

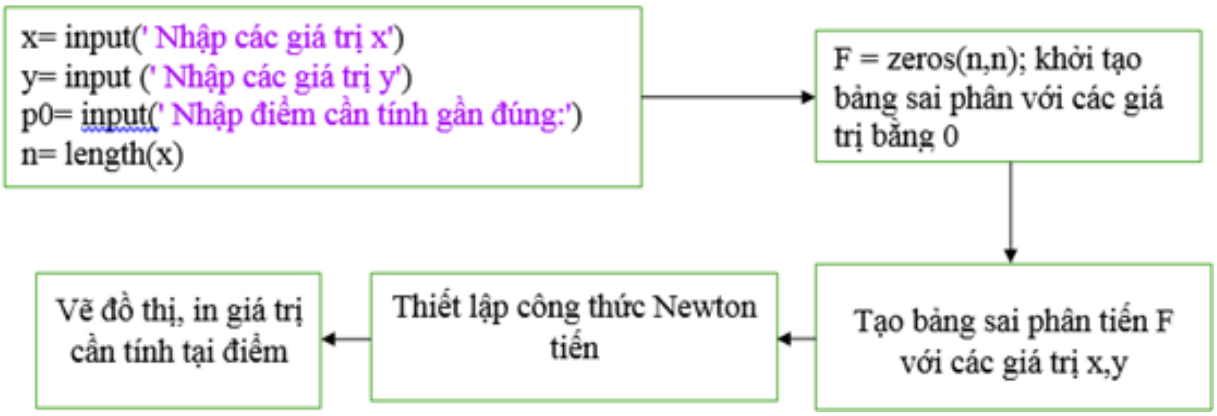


Hình 2.3

2.3 Phương pháp Newton *(Xét trường hợp các mốc cách đều)*

2.3.1 Newton tiến

2.3.1.1 Sơ đồ giải thuật



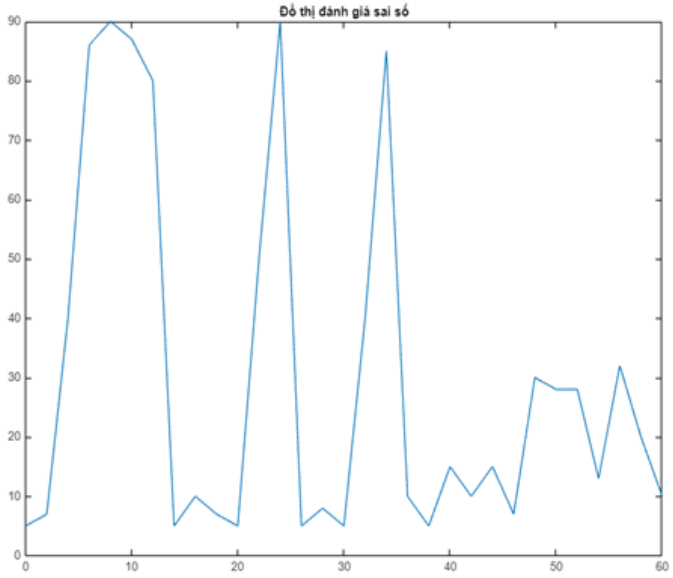
Hình 2.4

2.3.1.2 Code matlab

```
% Tao bang sai phan tien
x= input(' Nhap cac gia tri x');
y= input(' Nhap cac gia tri y');
p= input(' Nhap gia tri can tinh');
n= length(x);
F= zeros(n,n);
F(:,1)=y;
for j=2:n
   for i=1:(n-j+1)
       F(i,j)= F(i+1,j-1)-F(i,j-1);
   end
end
F
% Thiet lap cong thuc
h= x(2)-x(1);
u= (p-x(1))/h;
A= y(1); G=u;
for k=1:n-1
   A=A+G*F(1,k+1);
   G= (u-k)/(k+1)*G;
end
% Ve do thi
plot(x, y);
title('Do thi danh gia sai so');
```

```
fprintf(' Gia tri cua Y(%f)=%f\n',p,A);
```

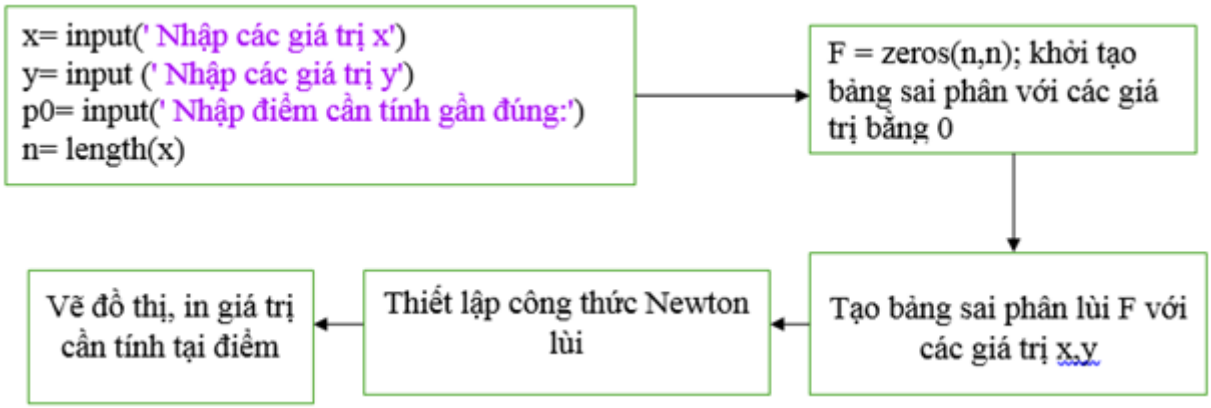
2.3.1.3 *Kết quả*



Hình 2.5 Đồ thị đánh giá sai số (Newton tiến)

2.3.2 *Newton lùi*

2.3.2.1 *Sơ đồ giải thuật*



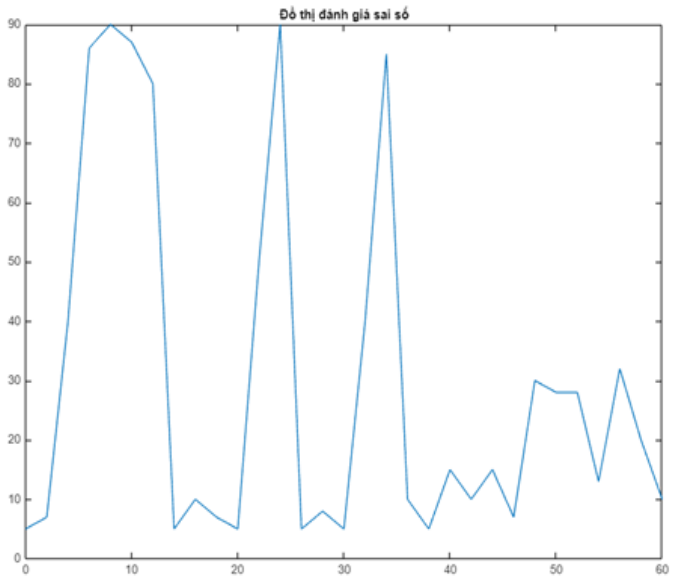
Hình 2.6

2.3.2.2 *Code matlab*

```
% Tao bang sai phan lui
x= input(' Nhap cac gia tri x');
y= input (' Nhap cac gia tri y');
p= input(' Nhap diem can tinh gan dung:');
```

```
n= length(x);
F= zeros(n,n);
F(:,1) =y;
for j=2:n
   for i=1:n-j+1
       F(i,j) = F(i+1,j-1) - F(i,j-1);
   end
end
F
% Thiet lap cong thuc
h = x(2)-x(1);
u= (p-x(n))/h;
A= y(n);G=u;
for k=1:n-1
   A=A+G*F(n-k,k+1);
   G= (u+k)/(k+1)*G;
end
% Ve do thi
plot(x, y);
title('Do thi danh gia sai so');
fprintf(' Gia tri cua Y(%f)=%f\n',p,A);
```

2.3.2.3 Kết quả



Hình 2.7 Đồ thị đánh giá sai số (Newton lùi)

2.3.2.4 Đánh giá độ chính xác

Nhìn chung giữa dữ liệu mẫu và dữ liệu nội suy bằng phương pháp Newton có dáng hình khá tương đồng về các điểm mốc dữ liệu.

Khi tính toán tại các mốc nội suy cụ thể cho kết quả có tỷ lệ chính xác chưa cao vì:

- Biến động nhanh chóng: CPU Usage có thể thay đổi nhanh chóng trong thời gian ngắn. Nếu có sự biến động quá lớn giữa các mẫu dữ liệu phương pháp nội suy Newton có thể không nắm bắt cho ra kết quả không chính xác.
- Sai số tích phân: Khi số lượng mốc nội suy lớn, sai số trong quá trình tính toán các sai phân có thể tích tụ và dẫn đến kết quả không chính xác.
- Sự không ổn định của phương pháp tính toán: Phương pháp nội suy Newton có thể trở nên không ổn định khi số lượng mốc nội suy quá lớn, dẫn đến kết quả không chính xác hoặc không đúng về mặt toán học.

2.3.2.5 Đánh giá phương pháp

- *Ưu điểm*: Dùng Newton để mô tả dữ liệu hoạt động của CPU trong một khoảng thời gian cho kết quả chính xác tại vài khoảng và sai số không nhiều.
- *Nhược điểm*: Với số lượng mốc nội suy lớn thường cho kết quả có tỷ lệ chính xác chưa cao, biện pháp: giảm mốc nội suy.

2.4 Phương pháp Lagrange

2.4.1 Sơ đồ giải thuật

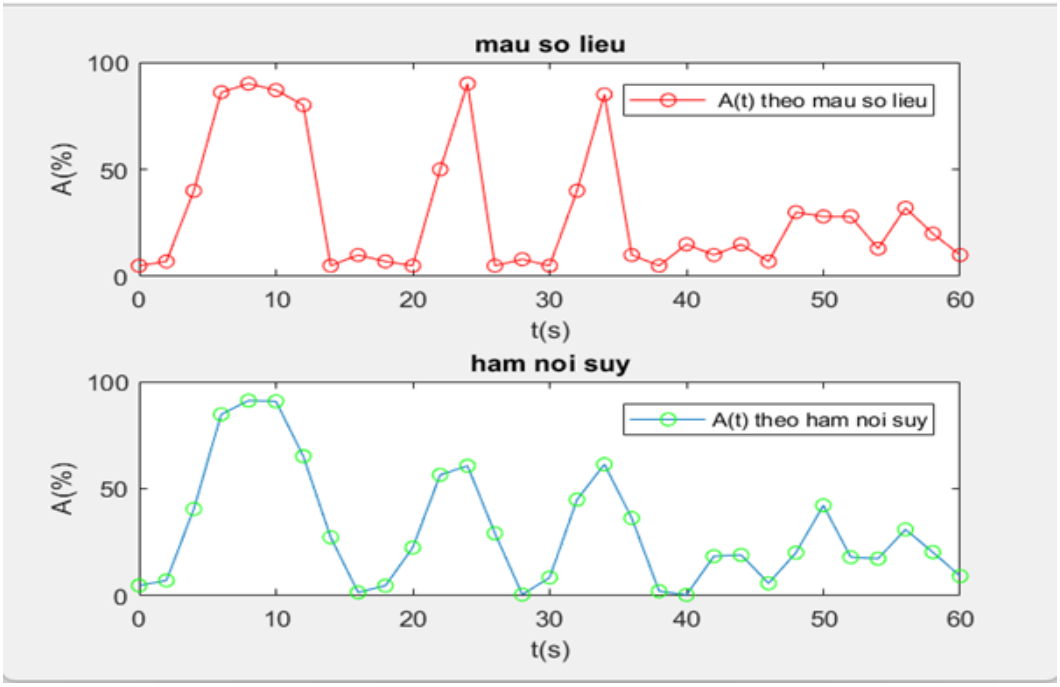
Hình 2.8

2.4.2 Code matlab

```
% Du lieu
t = [0 2 4 6 8 10 12 14 16 18 20 22 24 26 28 30 32 34 36 38 40 42 44
    46 48 50 52 54 56 58 60];
A = [5 7 40 86 90 87 80 5 10 7 5 50 90 5 8 5 40 85 10 5 15 10 15 7
    30 28 28 13 32 20 10];

% Giam bac cua da thuc
degree = 20; % Doi gia tri nay theo bac mong muon
P = polyfit(t, A, degree);

% Tao mot mang thoi gian moi de ve do thi muot ma hon
t_new = linspace(min(t), max(t), 31);

% Tinh toan gia tri cua da thuc noi suy tai cac diem moi
```

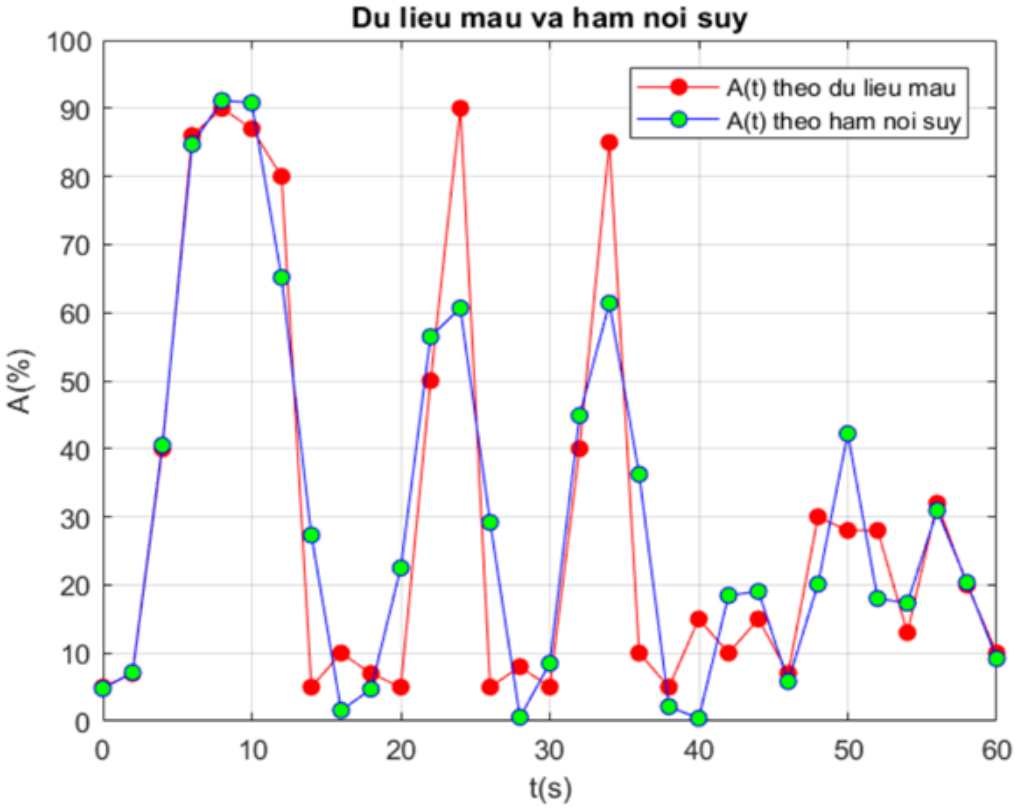
```
A_interp = polyval(P, t_new);
% Ve do thi
subplot(2, 1, 1); % Ve do thi dau tien
plot(t, A, 'o-', 'DisplayName', ' A(t) theo mau so
    lieu','MarkerEdgeColor', 'red','color','red');

xlabel('t(s)');
ylabel('A(%)');
title('mau so lieu');
legend('show');

% Ve do thi thu 2
subplot(2, 1, 2); % Ve do thi thu 2
plot(t_new, abs(A_interp),'o-', 'DisplayName','A(t) theo ham noi
    suy','MarkerEdgeColor', 'green');

xlabel('t(s)');
ylabel('A(%)');
title('ham noi suy');
legend('show');

% Noi suy tai mot diem cu the
x = 35; % Diem can noi suy
interpolated_value = lagrange_interpolation(t, A, x);
fprintf('gia tri noi suy tai diem %f la: %f\n', x,
    interpolated_value);

function interpolated_value = lagrange_interpolation(t, A, x)
    n = length(t);
    interpolated_value = 0;
    for i = 1:n
        term = A(i);
        for j = 1:n
            if j ~= i
                term = term * (x - t(j)) / (t(i) - t(j));
            end
        end
        interpolated_value = interpolated_value + term;
    end
end
```

2.4.3 Kết quả



Hình 2.9 Đồ thị hàm nội suy Lagrange so với mẫu số liệu

2.4.3.1 Đánh giá độ chính xác



Hình 2.10 Dữ liệu mẫu và hàm nội suy

Nhìn chung, đồ thị đa thức nội suy Lagrange mang dáng hình tương đối của hàm CPU Usage gốc với độ lệch giá trị có thể xem là khá cao tính từ thời điểm từ giây thứ 10 trở đi.

Với đa thức nội suy Lagrange, giá trị cần ước tính xấp xỉ trả về chưa được ổn định (chẳng hạn tại $t = 24s$, $t = 28s$, $t = 50s$). Lý do là vì sự thay đổi đột ngột giá trị % CPU Usage gốc dẫn đến dao động hàm lớn, khả năng bất liên tục của hàm số xảy ra theo chiều hướng tăng.

Dù nội suy Lagrange là phương pháp đơn giản về mặt ý tưởng nhưng mỗi khi bổ sung them một số điểm quan sát mới thì ta phải xây dựng lại toàn bộ đa thức nội suy. Thế nên việc lựa chọn và phân bố điểm quan sát ảnh hưởng trực tiếp đến độ chính xác của phép nội suy Lagrange.

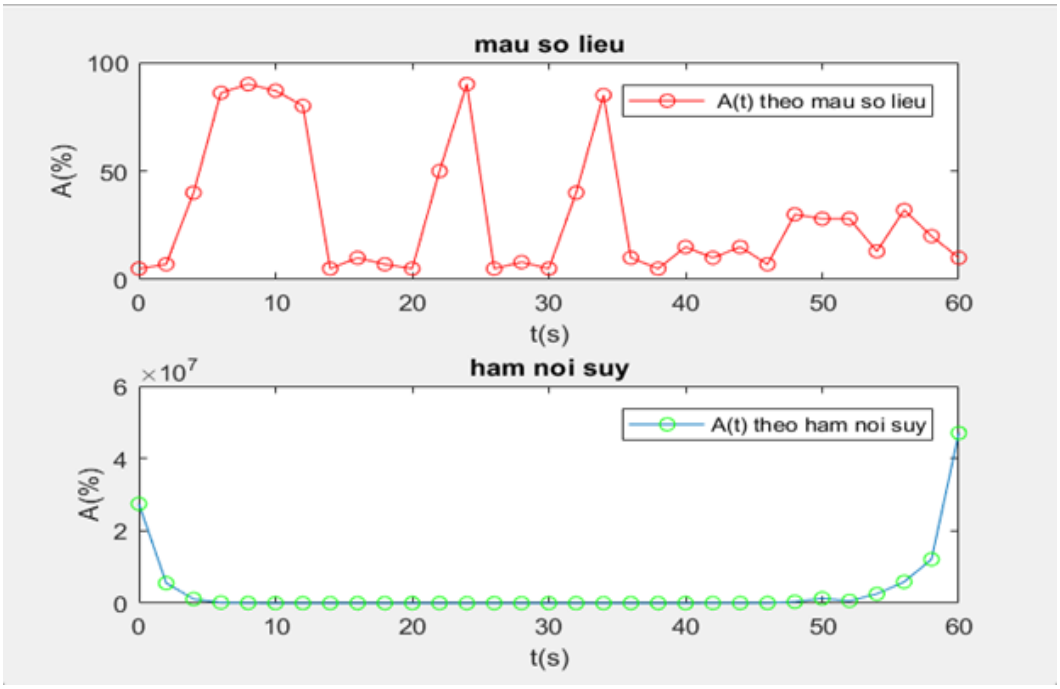
Mốc thời gian tham chiếu (s)	CPU usage thực tế (%)	CPU usage nội suy (%)	Sai số giữa thực tế đối với nội suy (%)
25	47	46.0233	2.0781
29	6	12.0440	100.7333
33	66	76.9881	16.6486
41	11	4.1744	62.0509
43	14	23.4962	67.8300

$$\%\ \text{Sai số} = \left| \frac{CPU\,usage_{thucte} - CPU\,usage_{noisuy}}{CPU\,usage_{thucte}} \right| \times 100\%$$

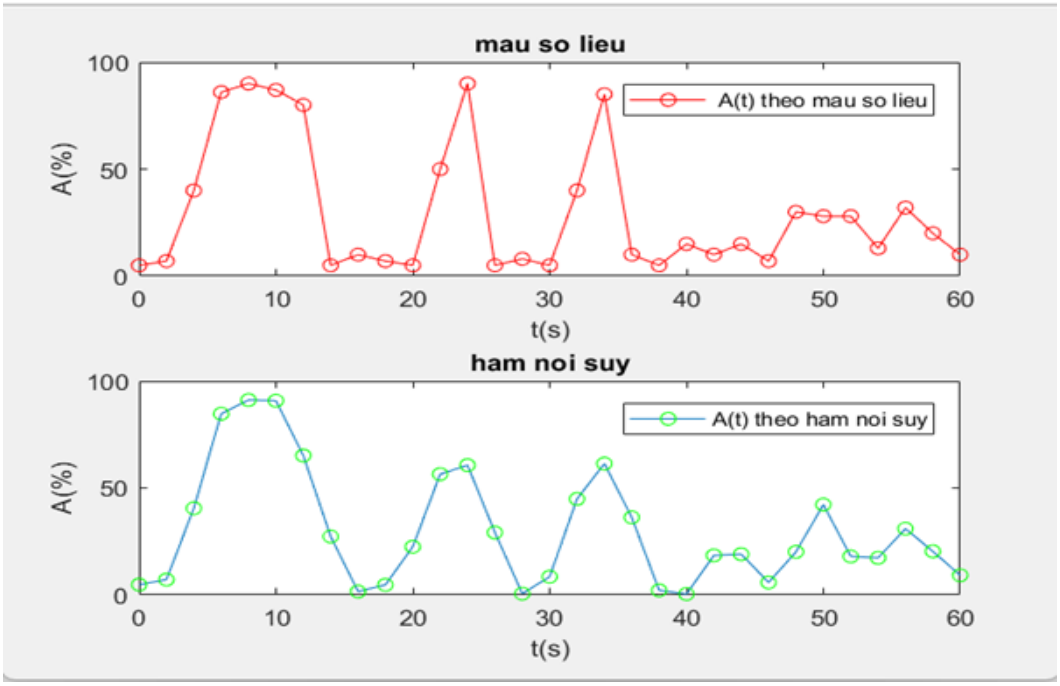
2.4.3.2 Đánh giá phương pháp

- *Ưu điểm*: tính chính xác nhanh chóng các hàm nội suy có bậc thấp (< 10).
- *Nhược điểm*: khi tính toán cách hàm bậc cao, đa thức nội suy Lagrane sẽ có tỷ lệ chính xác thấp hơn so với các phương pháp còn lại. Cụ thể, với bảng số liệu trên, hàm Lagrange theo lý thuyết sẽ chọn bậc của đa thức là 30. Nhưng khi ta chọn bậc đa thức là 30, hàm đa thức nội suy sẽ sai so với các phương pháp khác.

⇒ *Giải pháp: giảm bậc và khoảng chia của giá trị.*



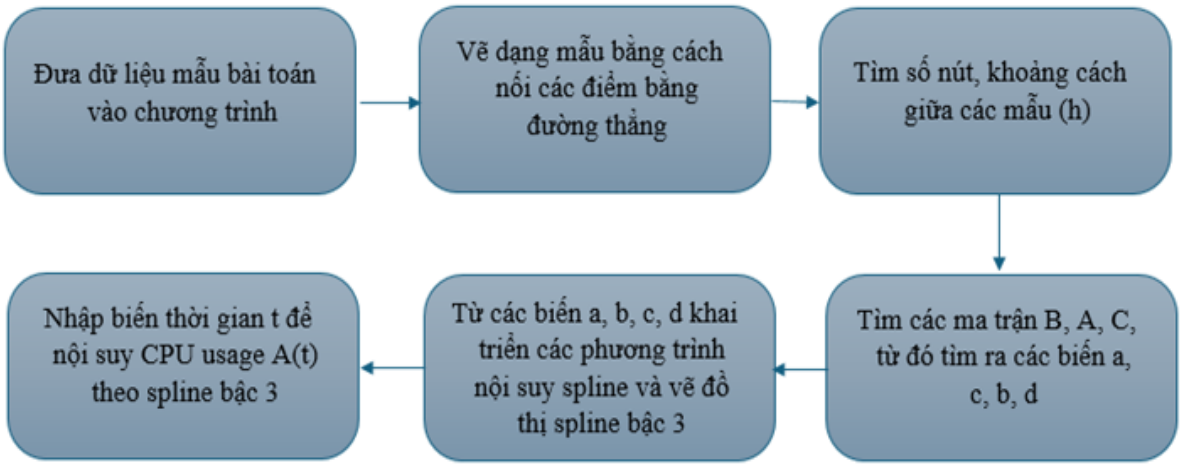
Hình 2.11 Áp dụng theo lý thuyết



Hình 2.12 Khi giảm bậc còn 20 và khoảng chia là 31

2.5 Phương pháp nội suy spline bậc 3

2.5.1 Sơ đồ giải thuật



Hình 2.13

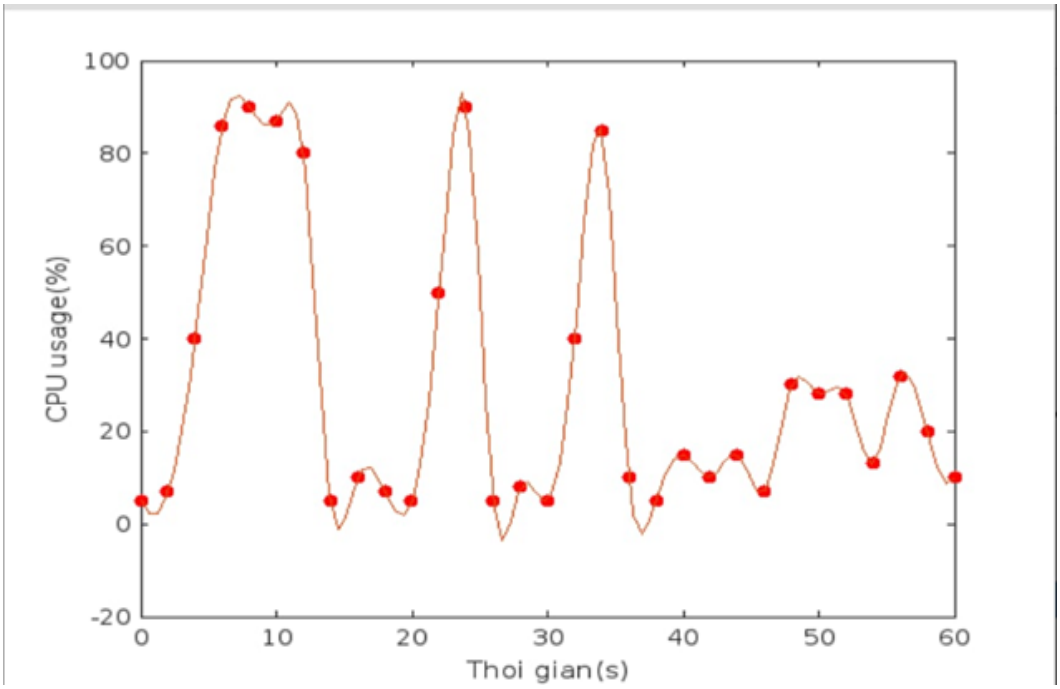
2.5.2 Code matlab

```
clc ;
clear;
close all;
%input t~x A~y(x)
t = [0 2  4  6  8  10 12 14 16 18 20 22 24 26 28 30 32 34
   36 38 40 42 44 46 48 50 52 54 56 58 60];
A = [5 7  40 86 90 87 80 5  10 7  5  50 90 5  8  5  40 85
   10 5  15 10 15 7  30 28 28 13 32 20 10];
%so phan tu N
N = length(t);
figure('name','Mau','color','white');
plot(t,A,'.r','markersize',20);
xx=linspace(t(1),t(N));
pp=spline(t,A);
yy=ppval(pp,xx);
xlabel('Thoi gian(s)');
ylabel('CPU usage(%)');
hold on
plot(t,A);
%so nut n
n= N-1;
x=zeros(1,N);
a=zeros(1,N);
%-> a
```

```
x=t;
a=A;
h=zeros(1,n);
%tim h
for i= 1:n
    h(i)=x(i+1)- x(i);
end
%ma tran b
matrixB = zeros(1,n);
for i = 1:n-1
    matrixB(i+1)= 3*((a(i+2)- a(i+1))/h(i+1)-(a(i+1)- a(i))/h(i));
end
%ma tran a
xl=zeros(1,N);
xl(1)=1;
xz=zeros(1,N);
xu=zeros(1,N);
for i=1:n-1
    xl(i+1)= 2*(x(i+2)-x(i))-h(i)*xu(i);
    xu(i+1)= h(i+1)/xl(i+1);
    xz(i+1)=(matrixB(i+1)-h(i)*xz(i))/xl(i+1);
end
xl(N)=1;
xz(N)=0;

b=zeros(1,N);
c=zeros(1,N);
d=zeros(1,N);
% -> c b d
for i= 0:n-1
    j=n-1-i;
    c(j+1)=xz(j+1)-xu(j+1)*c(j+2);
    b(j+1)=(a(j+2)-a(j+1))/h(j+1) - h(j+1)*(c(j+2)+2*c(j+1))/3;
    d(j+1)=(c(j+2)-c(j+1))/(3*h(j+1));
end
%hien ket qua
for i=1:n
fprintf('t tu %5.2fs den %5.2fs \n',x(i),x(i+1));
fprintf('CPU usage(t)=%5.2f + %5.2f(t-%5.2f) + %5.2f(t-%5.2f)^2 +
   %5.2f(t-%5.2f)^3 \n',a(i),b(i),x(i),c(i),x(i),d(i),x(i));
fprintf('\n')
end
```

```
figure('name','Spline3','color','white');
plot(t,A,'.r','markersize',20);
xx=linspace(t(1),t(N));
pp=spline(t,A);
yy=ppval(pp,xx);
xlabel('Thoi gian(s)');
ylabel('CPU usage(%)');
hold on
plot(xx,yy);
sym z;
z=-1;
while (z <= x(1) || z >= x(N))
fprintf('Nhap gia tri can noi suy trong khoang tu %5.2fs den %5.2fs:
    ',x(1),x(N));
z = input('');
end
for i= 1:n
    if z >= x(i) && z<=x(i+1)
        s = a(i)+b(i).*(z-x(i))+c(i).*(z-x(i)).^2+d(i).*(z-x(i)).^3;
    end
end
fprintf('CPU usage(t) = %5.2f',s);
```

2.5.3 Kết quả



Hình 2.14 Đường nội suy Spline bậc 3

2.5.3.1 Các phương trình nội suy tính toán được từ Matlab

- t từ 0.00s đến 2.00s

$$CPUusage(t) = 5.00 + -2.31(t-0.00) + 0.00(t-0.00)^2 + 0.83(t-0.00)^3$$

- t từ 2.00s đến 4.00s

$$CPUusage(t) = 7.00 + 7.63(t-2.00) + 4.97(t-2.00)^2 + -0.27(t-2.00)^3$$

- t từ 4.00s đến 6.00s

$$CPUusage(t) = 40.00 + 24.30(t-4.00) + 3.36(t-4.00)^2 + -2.01(t-4.00)^3$$

- t từ 6.00s đến 8.00s

$$CPUusage(t) = 86.00 + 13.68(t-6.00) + -8.68(t-6.00)^2 + 1.42(t-6.00)^3$$

- t từ 8.00s đến 10.00s

$$CPUusage(t) = 90.00 + -4.00(t-8.00) + -0.16(t-8.00)^2 + 0.71(t-8.00)^3$$

- t từ 10.00s đến 12.00s

$$CPUusage(t) = 87.00 + 3.83(t-10.00) + 4.08(t-10.00)^2 + -3.87(t-10.00)^3$$

- t từ 12.00s đến 14.00s

$$CPUusage(t) = 80.00 + -26.31(t-12.00) + -19.15(t-12.00)^2 + 6.78(t-12.00)^3$$

- t từ 14.00s đến 16.00s

$$CPUusage(t) = 5.00 + -21.59(t-14.00) + 21.51(t-14.00)^2 + -4.73(t-14.00)^3$$

- t từ 16.00s đến 18.00s

$$CPUusage(t) = 10.00 + 7.66(t-16.00) + -6.88(t-16.00)^2 + 1.15(t-16.00)^3$$

- t từ 18.00s đến 20.00s

$$CPUusage(t) = 7.00 + -6.06(t-18.00) + 0.02(t-18.00)^2 + 1.26(t-18.00)^3$$

- t từ 20.00s đến 22.00s

$$CPUusage(t) = 5.00 + 9.08(t-20.00) + 7.55(t-20.00)^2 + -0.42(t-20.00)^3$$

- t từ 22.00s đến 24.00s

$$CPUusage(t) = 50.00 + 34.23(t - 22.00) + 5.02(t - 22.00)^2 + -6.07(t - 22.00)^3$$

- t từ 24.00s đến 26.00s

$$CPUusage(t) = 90.00 + -18.50(t - 24.00) + -31.39(t - 24.00)^2 + 9.70(t - 24.00)^3$$

- t từ 26.00s đến 28.00s

$$CPUusage(t) = 5.00 + -27.71(t - 26.00) + 26.79(t - 26.00)^2 + -6.09(t - 26.00)^3$$

- t từ 28.00s đến 30.00s

$$CPUusage(t) = 8.00 + 6.35(t - 28.00) + -9.75(t - 28.00)^2 + 2.91(t - 28.00)^3$$

- t từ 30.00s đến 32.00s

$$CPUusage(t) = 5.00 + 2.30(t - 30.00) + 7.73(t - 30.00)^2 + -0.07(t - 30.00)^3$$

- t từ 32.00s đến 34.00s

$$CPUusage(t) = 40.00 + 32.43(t - 32.00) + 7.33(t - 32.00)^2 + -6.15(t - 32.00)^3$$

- t từ 34.00s đến 36.00s

$$CPUusage(t) = 85.00 + -12.02(t - 34.00) + -29.56(t - 34.00)^2 + 8.41(t - 34.00)^3$$

- t từ 36.00s đến 38.00s

$$CPUusage(t) = 10.00 + -29.34(t - 36.00) + 20.90(t - 36.00)^2 + -3.74(t - 36.00)^3$$

- t từ 38.00s đến 40.00s

$$CPUusage(t) = 5.00 + 9.38(t - 38.00) + -1.54(t - 38.00)^2 + -0.32(t - 38.00)^3$$

- t từ 40.00s đến 42.00s

$$CPUusage(t) = 15.00 + -0.66(t - 40.00) + -3.48(t - 40.00)^2 + 1.28(t - 40.00)^3$$

- t từ 42.00s đến 44.00s

$$CPUusage(t) = 10.00 + 0.78(t - 42.00) + 4.19(t - 42.00)^2 + -1.67(t - 42.00)^3$$

- t từ 44.00s đến 46.00s

$$CPUusage(t) = 15.00 + -2.44(t-44.00) + -5.80(t-44.00)^2 + 2.51(t-44.00)^3$$

- t từ 46.00s đến 48.00s

$$CPUusage(t) = 7.00 + 4.48(t-46.00) + 9.26(t-46.00)^2 + -2.88(t-46.00)^3$$

- t từ 48.00s đến 50.00s

$$CPUusage(t) = 30.00 + 7.02(t-48.00) + -7.99(t-48.00)^2 + 1.99(t-48.00)^3$$

- t từ 50.00s đến 52.00s

$$CPUusage(t) = 28.00 + -1.05(t-50.00) + 3.96(t-50.00)^2 + -1.72(t-50.00)^3$$

- t từ 52.00s đến 54.00s

$$CPUusage(t) = 28.00 + -5.84(t-52.00) + -6.36(t-52.00)^2 + 2.76(t-52.00)^3$$

- t từ 54.00s đến 56.00s

$$CPUusage(t) = 13.00 + 1.89(t-54.00) + 10.22(t-54.00)^2 + -3.21(t-54.00)^3$$

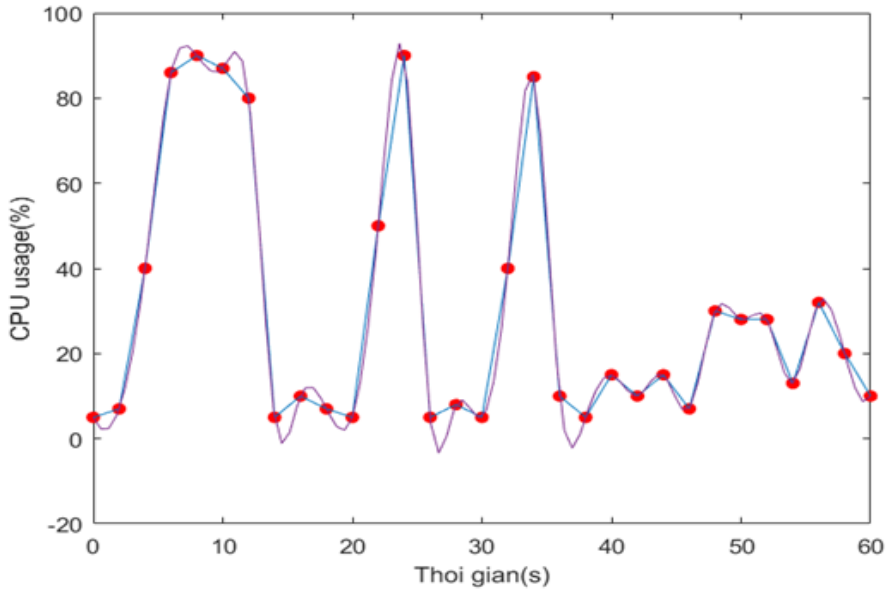
- t từ 56.00s đến 58.00s

$$CPUusage(t) = 32.00 + 4.28(t-56.00) + -9.03(t-56.00)^2 + 1.94(t-56.00)^3$$

- t từ 58.00s đến 60.00s

$$CPUusage(t) = 20.00 + -8.51(t-58.00) + 2.63(t-58.00)^2 + -0.44(t-58.00)^3$$

2.5.3.2 Tính toán sai số và nhận xét, đánh giá



Hình 2.15 Chênh lệch giữa mẫu và nội suy

* Chọn ngẫu nhiên các giá trị t { 9 21 35 47 53 } để tham chiếu:

- Nhập giá trị cần nội suy trong khoảng từ 0.00s đến 60.00s: 9
  $\Rightarrow$ CPU usage(t) = 86.54
- Nhập giá trị cần nội suy trong khoảng từ 0.00s đến 60.00s: 21
  $\Rightarrow$ CPU usage(t) = 21.21
- Nhập giá trị cần nội suy trong khoảng từ 0.00s đến 60.00s: 35
  $\Rightarrow$ CPU usage(t) = 51.83
- Nhập giá trị cần nội suy trong khoảng từ 0.00s đến 60.00s: 47
  $\Rightarrow$ CPU usage(t) = 17.87
- Nhập giá trị cần nội suy trong khoảng từ 0.00s đến 60.00s: 53
  $\Rightarrow$ CPU usage(t) = 18.57

Mốc thời gian tham chiếu (s)	CPU usage thực tế (%)	CPU usage nội suy (%)	Sai số giữa thực tế đối với nội suy (%)
9	88	86,54	1,6591
21	16	21,21	32,5625
35	10	51,83	418,3000
47	28	17,87	36,2857
53	20	18,87	5,6500

$$\%\ \text{Sai số} = \left| \frac{CPU\,usage_{thucte} - CPU\,usage_{noisuy}}{CPU\,usage_{thucte}} \right| \times 100\%$$

Nhận xét: Khi sử dụng Matlab tính toán một số mốc nội suy. Ta thấy rằng, ở mỗi khoảng nội suy sẽ cho ra kết quả sai số giữa thực tế và nội suy là khác nhau và chênh lệch tương đối lớn ở một số khoảng.

Đánh giá: Dùng phương pháp spline bậc 3 để mô tả dữ liệu hoạt động của CPU Usage trong một khoảng thời gian cho ra kết quả chính xác ở vài khoảng và sai số không quá nhiều.

CHƯƠNG 3: TỔNG KẾT

3.6 Kiến thức

- Nắm vững kiến thức về các phương pháp nội suy và cách ứng dụng nó vào trong đời sống.
- Biết cách sử dụng và làm quen với chương trình matlab và latex.
- Biết thêm các kiến thức về đo dữ liệu, phân tích hoạt động của CPU usage.

3.7 Kỹ năng

Sau một khoảng thời gian làm việc cùng nhau, các thành viên đã tích lũy được các kỹ năng:

- Kỹ năng làm việc nhóm.
- Kỹ năng giao tiếp.
- Tinh thần tự giác cao và có trách nhiệm.

3.8 Hạn chế

Ngoài những lợi ích mà buổi làm việc nhóm mang lại, cũng còn có những mặt hạn chế:

- Chưa hiểu rõ được các nội dung của sách tham khảo.
- Chưa tìm hiểu đủ sâu và chi tiết đề tài.
- Các buổi gặp mặt trực tiếp còn ít.

TÀI LIỆU THAM KHẢO

[1] M. J. White, “What is cpu usage, and how to fix high cpu usage,” 2024.

[2] C. C. Steven and P. C. Raymond, *Numerical Methods for Engineers 7th Edit*. McGraw-Hill Education, 2 Penn Plaza, New York, NY 10121, 2015.

[3] T. Hiếu, *Tài liệu tiếng việt phương pháp tính*.