ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA ĐIỆN TỬ - VIỄN THÔNG

------OOO------

BÁO CÁO ĐỒ ÁN

THỰC HÀNH PHƯƠNG PHÁP TÍNH

THIẾT KẾ GIAO DIỆN & LUẬN LÝ GIẢI QUYẾT CÁC BÀI TOÁN VỚI MATLAB APP DESIGNER

Giảng viên hướng dẫn

Thầy Huỳnh Quốc Thịnh

Thầy Nguyễn Xuân Vinh

Sinh viên thực hiện

Nguyễn Trần Hoàng Phúc – 22207128

Giao Hữu Lực – 23207080

Lê Quang Thông – 23207113

Đặng Bá Trần Trung – 23207121

TP. HỒ CHÍ MINH – 12/2025

MỤC LỤC

DANH MỤC HÌNH ẢNH

DANH MỤC BẢNG

CODE CELLS

MỞ ĐẦU

Ngày nay, khi các công cụ hỗ trợ học tập ngày càng được ứng dụng rộng rãi trong giảng dạy các môn kỹ thuật, việc kết hợp giữa lý thuyết và thực hành thông qua các ứng dụng trực quan đang trở thành một xu hướng truyền đạt kiến thức hiệu quả. Cách tiếp cận này không chỉ giúp người học dễ dàng tiếp thu kiến thức mà còn tăng khả năng vận dụng vào các bài toán thực tiễn.

Trong quá trình học tập môn *Phương pháp tính*, nhóm nhận thấy rằng các nội dung như tìm nghiệm gần đúng của phương trình, nội suy và hồi quy hàm số, tính đạo hàm và tích phân - vi phân không chỉ mang tính lý thuyết mà còn có ý nghĩa thực tiễn cao trong việc giải quyết các bài toán kỹ thuật. Tuy nhiên, để nắm vững và vận dụng hiệu quả các thuật toán của những phương pháp này, sinh viên cần có sự kết hợp chặt chẽ giữa lý thuyết, thực hành và khả năng trực quan hóa kết quả tính toán.

Từ thực tế đó, nhóm nhận thấy rằng việc xây dựng một ứng dụng hỗ trợ giải các bài toán phương pháp tính sẽ mang lại nhiều lợi ích thiết thực. Trước hết, quá trình thiết kế và cài đặt các thuật toán giúp các thành viên trong nhóm củng cố lại kiến thức đã học, hiểu rõ hơn nguyên lý hoạt động cũng như ưu và nhược điểm của từng phương pháp. Bên cạnh đó, việc tích hợp các phương pháp tính vào một ứng dụng có giao diện trực quan giúp việc tiếp cận và khai thác kiến thức trở nên thuận tiện và hiệu quả hơn.

Không chỉ phục vụ cho quá trình học tập của nhóm, ứng dụng còn có thể được sử dụng như một tài liệu tham khảo và công cụ hỗ trợ học tập cho sinh viên ở các khóa sau khi học môn *Phương pháp tính*. Thông qua ứng dụng, người học có thể thử nghiệm nhiều phương pháp khác nhau, so sánh kết quả và quan sát trực tiếp ảnh hưởng của từng thuật toán đến kết quả tính toán cuối cùng. Từ những cơ sở trên, nhóm tiến hành thực hiện báo cáo thực hành về quá trình xây dựng một ứng dụng giải các bài toán phương pháp tính bằng *MATLAB App Designer*, làm nền tảng cho việc trình bày chi tiết nội dung của đồ án ở các phần tiếp theo.

NỘI DUNG BÁO CÁO

CHƯƠNG 1: GIỚI THIỆU VỀ ĐỒ ÁN

1.1 Mục tiêu đồ án

Với cơ sở mong muốn xây dựng một ứng dụng giúp giải quyết các dạng bài toán đại số thuộc nội dung môn *Phương pháp tính* bằng công cụ App Designer của MATLAB, việc tìm hiểu đề tài sẽ xoay quanh một vài mục tiêu cụ thể.

Đầu tiên là giúp củng cố và hệ thống lại kiến thức về các phương pháp tính toán về việc tìm nghiệm gần đúng của phương trình đại số - hệ phương trình – phương trình vi phân, thuật toán nội suy và hồi quy, tính toán đạo hàm và tích phân. Bên cạnh đó, đây là cơ hội tìm hiểu và vận dụng được kĩ năng sử dụng MATLAB nói chung và App Designer nói riêng, đặc biệt là thiết kế giao diện (UI – User Interface) cũng như trực quan hóa được kết quả thông qua hiển thị số liệu và vẽ đồ thị. Cuối cùng, ứng dụng được xây dựng theo nguyên tắc tách biệt giữa phần giao diện và phần xử lý thuật toán nhằm thuận tiện cho việc chỉnh sửa, mở rộng chương trình và sử dụng soucre code như một tài liệu tham khảo cho việc học tập và giảng dạy trong tương lai.

1.2 Phạm vi và yêu cầu cần đạt được

Đồ án này tập trung xây dựng một ứng dụng MATLAB dạng App Designer gồm nhiều tab, phục vụ cho việc giải và minh họa các bài toán tính toán số cơ bản trong chương trình Toán ứng dụng/Kỹ thuật. Phạm vi đồ án bao gồm các bài toán: tìm nghiệm gần đúng của phương trình, nội suy và hồi quy dữ liệu, tính gần đúng đạo hàm và tích phân xác định bằng các phương pháp số thông dụng. Các thuật toán được cài đặt dựa trên cơ sở lý thuyết đã học, với dữ liệu đầu vào do người dùng cung cấp, không xét đến các bài toán quy mô lớn hay yêu cầu tối ưu hiệu năng cao.

Mục tiêu của đồ án là giúp nhóm củng cố và vận dụng kiến thức lý thuyết vào thực hành, hiểu rõ nguyên lý hoạt động, ưu nhược điểm của từng phương pháp tính toán số. Đồng thời, đồ án hướng tới việc xây dựng một ứng dụng có giao diện trực quan, dễ sử dụng, hỗ trợ người học trong việc khảo sát, so sánh kết quả và kiểm chứng thuật toán. Thông qua quá trình thực hiện, nhóm rèn luyện thêm kỹ năng lập trình MATLAB, thiết kế giao diện ứng dụng và làm việc nhóm thông qua công cụ quản lý mã nguồn Github.

CHƯƠNG 2: CƠ SỞ LÝ THUYẾT & THUẬT TOÁN

2.1 Bài toán tìm nghiệm gần đúng của phương trình

2.1.1 Phương pháp chia đôi

Phương pháp chia đôi (bisection method) sẽ dựa trên định lý *giá trị trung gian.* Giả sử cho hàm số $f(x)$ liên tục trên đoạn $[a; b]$ và:

$$f(a) \cdot f(b) < 0$$

thì sẽ tồn tại ít nhất một nghiệm $x_0 \in [a; b]$ để $f(x_0) = 0$ với $[a; b]$ được gọi là *khoảng phân ly nghiệm* của phương trình. Ý tưởng của phương pháp là với mỗi lần lặp, ta tính giá trị trung điểm của khoảng chứa nghiệm:

$$c = \frac{a + b}{2}$$

rồi liên tục chia đôi khoảng và kiểm tra dấu để thu hẹp khoảng chứa nghiệm:

$$f(a) \cdot f(c) < 0 \rightarrow \text{nghiệm nằm trong đoạn } [a; c]$$

$$f(b) \cdot f(c) < 0 \rightarrow \text{nghiệm nằm trong đoạn } [c; b]$$

Quá trình này được lặp lại cho đến khi đạt được độ chính xác mong muốn hoặc khi giá trị $|b - a|$ nhỏ hơn một sai số cho phép ϵ cho trước. Ưu điểm của phương pháp chia đôi là tính hội tụ chắc chắn và đơn giản cho việc lập trình, cũng như không phụ thuộc vào giá trị khởi đầu, chỉ cần điều kiện đổi dấu được thỏa mãn. Tuy nhiên, tốc độ hội tụ lại tương đối chậm so với các phương pháp khác như phương pháp tiếp tuyến (Newton – Raphson) hay phương pháp dây cung do độ dài khoảng nghiệm chỉ giảm một nửa sau mỗi lần lặp.

2.1.2 Phương pháp lặp đơn

Là một phương pháp để tìm nghiệm gần đúng của $f(x) = 0$ bằng cách đưa phương trình về dạng $x = g(x)$ với $g(x)$ liên tục trên $[a; b]$. Sau đó chọn điểm ban đầu $x_0 \in [a; b]$ và tính dãy lặp $\{x_n\}$ theo công thức:

$$x_n = g(x_{n-1})$$

$$(n = 1,2,3, \dots, k)$$

Quá trình lặp sẽ được thực thi đến khi $|f(x_n)|$ đạt độ chính xác mong muốn hoặc khi thỏa sai số ước lượng:

$$|x_n - a| \leq \frac{q}{1-q}|x_n - x_{n-1}|$$

với q được coi là *hệ số hội tụ* thỏa định lý $|g'(x)| \leq q \leq 1, \forall x \in [a;b]$. Khi điều kiện này được đảm bảo, dãy lặp sẽ hội tụ về nghiệm duy nhất của phương trình trong khoảng phân li. Ngược lại, phương pháp có thể hội tụ chậm hoặc phân kỳ. Việc lựa chọn hàm lặp $g(x)$ sao cho q nhỏ ($q \leq 1$) là yếu tố quan trọng ảnh hướng tới độ hiệu quả của thuật toán này, nên trong thực tế để đảm bảo điều kiện hội tụ thì giá trị q thường sẽ được ước lượng:

$$q = \max_{x \in [a,b]} |g'(x)|$$

Phương pháp lặp đơn có điểm mạnh là dễ cài đặt, công thức lặp đơn giản và không yêu cầu tính đạo hàm của hàm số gốc, cũng như điểm xấp xỉ ban đầu x_0 không cần phải gần a. Tuy nhiên, hạn chế lớn là phụ thuộc vào cách lựa chọn hàm lặp $g(x)$ và giá trị ban đầu x_0, bên cạnh đó khi hệ số hội tụ q càng tiến gần tới 1 thì tốc độ hội tụ sẽ rất chậm.

2.1.3 Phương pháp tiếp tuyến (Newton – Raphson)

Dựa trên ý tưởng xấp xỉ hàm số bằng tiếp tuyến tại một điểm gần nghiệm, phương pháp tiếp tuyến sẽ dùng đạo hàm của hàm số để tăng tốc độ hội tụ so với các phương pháp khác. Giả sử $[a;b]$ là khoảng phân ly nghiệm α của phương trình $f(x) = 0$ liên tục với $f'(x)$ và $f''(x)$ không đổi dấu trên $[a;b]$. Đặt $x_0 = a$ nếu tiếp tuyến kẻ từ điểm $A\big(a, f(a)\big)$, $x_0 = b$ nếu tiếp tuyến kẻ từ điểm $B\big(b, f(b)\big)$. Phương trình tiếp tuyến tại $\big(x_0, f(x_0)\big)$ có dạng:

$$y = f(x_0) + f'(x_0)(x_1 - x_0) \Rightarrow x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Tiếp tục vẽ tiếp tuyến tại $\big(x_1, f(x_1)\big)$, ta có xấp xỉ kế tiếp: $x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$. Suy ra công thức tính tổng quát:

$$x_n = x_{n-1} - \frac{f(x_{n-1})}{f'(x_{n-1})}$$

Quá trình lặp được thực hiện cho đến khi $|f(x_n)|$ đạt độ chính xác mong muốn hoặc độ chênh lệch giữa hai bước lặp liên tiếp nhỏ hơn một sai số ϵ cho trước:

$$|x_n - x_{n-1}| \leq \epsilon$$

Phương pháp tiếp tuyến còn được gọi là phương pháp tuyến tính hóa, sẽ hội tụ rất nhanh khi giá trị khởi đầu x_0được chọn đủ gần nghiệm chính xác. Trong trường hợp hội tụ, tức là khi $f(x_0)$ cùng dấu với $f''(x)$, phương pháp có tốc độ hội tụ bậc hai, nghĩa là sai số giảm rất nhanh sau mỗi bước lặp. Tuy nhiên, nhược điểm của phương pháp là phụ thuộc mạnh vào giá trị khởi đầu x_0 và cần phải tính đạo hàm của hàm số. Nếu chọn x_0 không phù hợp hoặc nếu $f'(x_n)$ gần bằng 0 trong quá trình lặp sẽ dẫn tới việc phương pháp có thể hội tụ chậm hoặc không hội tụ.

2.2 Bài toán nội suy hàm số

Trong nhiều bài toán thực tế, hàm số cần khảo sát không được biết dưới dạng biểu thức giải tích mà chỉ được xác định thông qua một tập hữu hạn các giá trị tại những điểm rời rạc.

2.2.1 Nội suy Lagrange

Lagrange là một phương pháp nội suy đa thức dùng để xây dựng một đa thức xấp xỉ đi qua tất cả các điểm dữ liệu đã cho. Phương pháp này cho phép xác định trực tiếp đa thức nội suy mà không cần giải hệ phương trình tuyến tính. Giả sử cho một tập hữu hạn $n + 1$:

x	x_0	x_1	x_2	...	x_{n-1}
y	$f(x_0)$	$f(x_1)$	$f(x_2)$	...	$f(x_{n-1})$

Đa thức nội suy Lagrange $P_n(x)$ đi qua toàn bộ các điểm dữ liệu đã cho và được sử dụng để xấp xỉ giá trị của hàm số tại các điểm nằm giữa các nút nội suy, $P_n(x)$ có dạng:

$$P_n(x) = \sum_{i=0}^{n} y_i L_i(x) = y_0 L_0(x) + y_1 L_1(x) + y_2 L_{2(x)} + \cdots + y_{n-1} L_{n-1}(x)$$

$$\text{với } L_i(x) = \frac{(x-x_0)(x-x_1)(x-x_2)\ldots(x_{x_n})}{(x_i-x_0)(x_i-x_1)(x_i-x_2)\ldots(x_i-x_n)}$$

Phương pháp nội suy Lagrange có ưu điểm là công thức rõ ràng, dễ hiểu và dễ dàng lập trình đối với số lượng điểm nội suy nhỏ. Tuy nhiên khi số điểm nội suy tăng lên, đa thức nội suy có thể dao động mạnh và việc bổ sung thêm điểm dữ liệu mới sẽ yêu cầu tính lại toàn bộ đa thức nội suy.

2.2.2 Nội suy Newton

Nếu trong phương pháp Lagrange, mỗi khi thêm mốc nội suy mới thì cần phải tính lại toàn bộ đa thức nội suy, dẫn tới tăng thời gian xử lý và sai số không mong muốn. Đa thức nội suy Newton sẽ khắc phục hạn chế này bằng việc sử dụng *tỉ hiệu* (divided difference) để xây dựng đa thức nội suy linh hoạt hơn trong thực tế. Giả sử cho tập $n+1$ điểm dữ liệu phân biệt:

x	x_0	x_1	x_2	...	x_{n-1}
y	$f(x_0)$	$f(x_1)$	$f(x_2)$	...	$f(x_{n-1})$

Đa thức nội suy Newton $P_n(x)$ đi qua toàn bộ các điểm dữ liệu đã cho và được sử dụng để xấp xỉ giá trị của hàm số tại các điểm nằm giữa các nút nội suy có dạng:

- Newton tiến: $P_n(x) = f(x_0) + f(x - x_0)(x - x_0) + \cdots + f(x_0, x_1, \dots, x_n)(x - x_{n-1})$

- Newton lùi: $P_n(x) = f(x_n) + (x - x_n)f(x_n, x_{n-1}) + \cdots + (x - x_1)f(x_n, \dots, x_0)$

- Tỉ hiệu cấp 1 tại x_i và x_j: $f(x_j, x_j) = \frac{f(x_j) - f(x_i)}{x_j - x_i}$

- Tỉ hiệu cấp 2 tại x_i, x_j, x_k: $f(x_i, x_j, x_k) = \frac{f(x_i, x_j) - f(x_j, x_k)}{x_i - x_k}$

… tương tự cho những tí hiệu cấp cao hơn.

Điểm mạnh của phương pháp nội suy Newton là dễ xử lý khi cần thêm điểm dữ liệu mới và thuận tiện trong tính toán, đặc biệt khi số lượng mốc nội suy tăng. Tuy nhiên, giống như nội suy Lagrange, phương pháp này cũng có thể gặp hiện tượng giá trị dao động khi bậc đa thức lớn hoặc khi các điểm nội suy phân bố không hợp lý.

2.3 Bài toán hồi quy hàm số

2.3.1 Hồi quy tuyến tính

Hồi quy tuyến tính được sử dụng để mô hình hóa mối quan hệ gần đúng giữa hai đại lượng thông qua một hàm tuyến tính, bài toán hồi quy tuyến tính đặt ra là tìm hàm số có dạng:

$$y = ax + b$$

sao cho đường thẳng này xấp xỉ tốt nhất các điểm dữ liệu đã cho trước. Để đánh giá mức độ phù hợp của mô hình, phương pháp *bình phương tối thiểu* được sử dụng nhằm tối thiểu hóa tổng bình phương sai số giữa giá trị thực nghiệm và giá trị dự đoán của mô hình. Hàm sai số được xác định:

$$S(a,b) = \sum_{i=1}^{n} (y_i - ax_i - b)^2$$

Từ điều kiện cực tiểu của hàm sai số $S(a,b)$, ta thu được công thức tính các hệ số a và b như sau:

$$a = \frac{n\sum_{i=1}^{n}(x_i y_i) - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n}(x_i)^2 - (\sum_{i=1}^{n} x_i)^2}$$

$$b = \frac{\sum_{i=1}^{n} y_i - a\sum_{i=1}^{n} x_i}{n}$$

Để đánh giá mức độ phù hợp của mô hình hồi quy với dữ liệu tính được, hệ số tương quan r^2 được sử dụng để phản ánh tỷ lệ biến thiên của dữ liệu được mô hình giải thích và được xác định bởi:

$$r^2 = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2 - \sum_{i=1}^{n}(y_i - ax_i - b)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \ (0 \le r^2 \le 1)$$

với r^2 càng gần 1 thì kết quả hồi quy càng khớpvới dữ liệu thực nghiệm, giúp đánh giá chất lượng xấp xỉ của mô hình và là tiêu chí quan trọng trong việc so sánh các phương pháp hồi quy khác nhau. Hồi quy tuyến tính là nền tảng cho nhiều phương pháp xấp xỉ và hồi quy nâng cao hơn, đồng thời được ứng dụng rộng rãi trong xử lý số liệu và phân tích dữ liệu thực nghiệm.

2.3.2 Hồi quy phi tuyến

a) Dạng $y = ae^{bx}$:

Trong nhiều bài toán thực tế, mối quan hệ giữa biến độc lập và biến phụ thuộc không tuân theo quy luật tuyến tính. Một trong những mô hình phi tuyến thường gặp là hàm mũ cơ số e:

$$y = ae^{bx}$$

Lấy logarith tự nhiên hai vế ta được:

$$ln(y) = ln(a) + bln(e^x) = ln(a) + bx$$

Đặt: $Y = ln(y), A = ln(a)$, $B = b$ và $X = x$, ta được dạng hàm tuyến tính:

$$Y = A + BX$$

Quá trình giải hệ phương trình, tìm các hệ số A và B sẽ tương tự với phương pháp hồi quy tuyến tính đã được trình bày ở trên.

b) Dạng $y = ax^b$:

Một dạng mô hình phi tuyến thường gặp khác là mô hình hàm lũy thừa có dạng:

$$y = ax^b$$

với $a \neq 0, x > 0$

Hàm số này thường được sử dụng để mô tả các hiện tượng có quan hệ tăng/giảm theo quy luật tỷ lệ, ví dụ: quan hệ giữa dòng điện và điện áp trong một số linh kiện, quan hệ hình học hoặc các bài toán thực nghiệm kỹ thuật.

Lấy logarith cơ số 10 hai vế:

$$log(y) = log(a) + blog(x)$$

Đặt: $Y = log(y), A = log(a), B = b$ và $X = log(x)$, thu được dạng hàm tuyến tính:

$$Y = A + BX$$

Quá trình giải hệ phương trình, tìm các hệ số A và B sẽ tương tự với phương pháp hồi quy tuyến tính đã được trình bày ở trên.

2.4 Bài toán tính gần đúng đạo hàm

Trong nhiều trường hợp hàm số $f(x)$ không có biểu thức rõ ràng hoặc có các giá trị x, y nhưng việc xây dựng hàm $y = f(x)$ khó khăn. Khi đó, việc tính đạo hàm chính xác là không dễ dàng, ta cần sử dụng các phương pháp số để xấp xỉ đạo hàm tại một điểm cho trước. Một trong những cách tính đó là áp dụng công thức Taylor, giả sử hàm số $f(x)$ có đạo hàm liên tục xung quanh điểm x. Khi đó, các giá trị của hàm tại $(x \pm h)$ có thể được biểu diễn thông qua khai triển Taylor:

- Công thức Taylor tại $(x + h)$ (xấp xỉ tiến):

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

với giá trị sai số xấp xỉ bậc $\boldsymbol{O(h)}$

- Công thức Taylor tại $(x-h)$ (xấp xỉ lùi):

$$f'(x) = \frac{f(x) - f(x-h)}{h}$$

với giá trị sai số xấp xỉ bậc $\boldsymbol{O(h)}$

- Công thức Taylor tại trung tâm:

$$f'(x) = \frac{f(x+h) - f(x-h)}{2h}$$

với giá trị sai số xấp xỉ bậc $\boldsymbol{O(h^2)}$

Qua các công thức xấp xỉ đạo hàm dựa trên khai triển Taylor, có thể thấy phương pháp xấp xỉ đạo hàm theo chiều tiến và lùi khá dễ hiểu, dễ áp dụng nhưng độ chính xác không cao nếu bước nhảy h còn lớn. Phương pháp xấp xỉ trung tâm cho kết quả chính xác hơn do sử dụng giá trị hàm ở hai đầu của điểm cần tính, tuy nhiên không áp dụng được ở các điểm biên. Vì vậy, trong thực tế cần lựa chọn phương pháp phù hợp tùy vào dữ liệu và yêu cầu độ chính xác của bài toán.

2.5 Bài toán tính gần đúng tích phân

Ý tưởng chung của các phương pháp tích phân số là chia đoạn $[\boldsymbol{a, b}]$ thành nhiều đoạn nhỏ, sau đó thay thế đồ thị hàm số bằng các hình học đơn giản như đoạn thẳng hoặc đa thức để tính gần đúng diện tích dưới đường cong. Độ chính xác của kết quả phụ thuộc vào số đoạn chia và phương pháp được sử dụng. Trong phạm vi đồ án, các phương pháp được xét bao gồm phương pháp hình thang, phương pháp Simpson 1/3 và phương pháp Simpson 3/8, là những phương pháp phổ biến, dễ cài đặt và cho độ chính xác.

2.5.1 Phương pháp hình thang

Phương pháp hình thang là một trong những phương pháp tích phân số đơn giản nhất. Ý tưởng chính là chia nhỏ khoảng lấy tích phân $[a, b]$ thành các hình thang. Hình thang cong được thay thế gần đúng bởi hình thang. Như vậy, tích phân gần đúng là tổng các diện tích hình thang nhỏ. Cách này tương đương với việc lấy tích phân của hàm nội suy bậc 1 của phương pháp nội suy Newton tiến với khoảng cách đều.

$$I = \int_a^b f(x)dx \approx \frac{\mathrm{h}}{2}\left[\mathrm{f(a)} + \mathrm{f(b)} + 2\sum_{\mathrm{i=1}}^{\mathrm{N-1}} \mathrm{f(a + ih)}\right]$$

Công thức hình thang chia khoảng [a,b] thành N đoạn con bằng nhau:

$$h = \frac{b-a}{N}$$

Phương pháp hình thang có ưu điểm là dễ hiểu và vận dụng, cài đặt, có thể áp dụng cả khi hàm số được cho dưới dạng tập hợp giá trị rời rạc. Tuy nhiên, do chỉ sử dụng xấp xỉ tuyến tính nên độ chính xác không cao đối với các hàm có độ cong lớn, trừ khi số đoạn chia N đủ lớn.

2.5.2 Phương pháp Simpson 1/3

Do việc tính gần đúng tích phân là việc lấy tích phân của hàm nội suy nên hàm nội suy càng chính xác sẽ cho kết quả gần đúng có sai số nhỏ hơn. Trong công thức Simpson, việc tính gần đúng tích phân là việc lấy tích phân hàm nội suy bậc 2 của phương pháp nội suy Newton xấp xỉ tiến với khoảng cách đều:

$$I = \int_a^b f(x)dx \approx \frac{h}{3}\left[f(a) + f(b) + 4\sum_{i=1}^{N-1} f(a+ih) + 2\sum_{i=2}^{N-1} f(a+ih)\right]$$

Công thức Simpson 1/3 chia đoạn $[a, b]$ thành N đoạn con bằng nhau với N chẵn:

$$h = \frac{b-a}{N}$$

Phương pháp Simpson 1/3 có độ chính xác cao hơn phương pháp hình thang, đặc biệt với các hàm có dạng cong đều. Tuy nhiên, phương pháp yêu cầu số đoạn chia phải là số chẵn, nên không linh hoạt bằng phương pháp hình thang.

2.5.3 Phương pháp Simpson 3/8

Phương pháp Simpson 3/8 cũng dựa trên ý tưởng xấp xỉ hàm số bằng đa thức bậc ba, cho độ chính xác cao hơn trong một số trường hợp nhất định:

$$I = \int_a^b f(x)dx \approx \frac{h}{3}\left[f(a) + f(b) + 3\sum_{i\neq 0(mod3)}^{N-1} f(a+ih) + 2\sum_{i=0(mod3)}^{N-1} f(a+ih)\right]$$

Phương pháp Simpson 3/8 cho kết quả chính xác hơn hình thang và trong một số trường hợp có thể tốt hơn Simpson 1/3. Tuy nhiên, điều kiện chia đoạn khắt khe hơn và cách tính phức tạp hơn, nên ít được sử dụng độc lập mà thường kết hợp với Simpson 1/3.

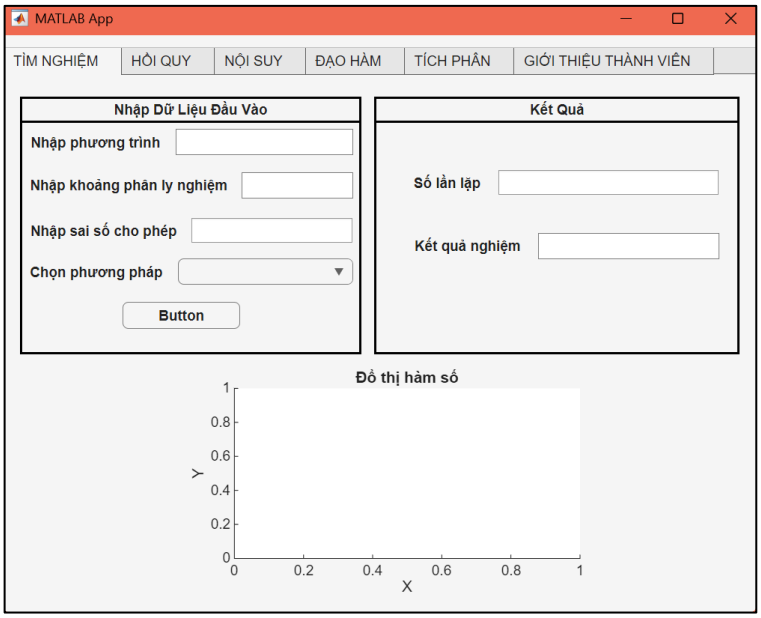
Mỗi phương pháp đều có những ưu điểm và hạn chế riêng, phù hợp với từng bài toán cụ thể. Phương pháp hình thang phù hợp cho các bài toán đơn giản hoặc dữ liệu rời rạc, trong khi các phương pháp Simpson cho độ chính xác cao hơn khi hàm số liên tục và đủ trơn. Việc lựa chọn phương pháp phụ thuộc vào yêu cầu độ chính xác, dạng dữ liệu đầu vào và khả năng chia đoạn của bài toán.

CHƯƠNG 3: GIAO DIỆN & HOẠT ĐỘNG CỦA ỨNG DỤNG

Chương này sẽ tập trung vào việc tìm hiểu giao diện và cách thức hoạt động của ứng dụng đã được xây dựng. Mục tiêu của chương là giúp hiểu rõ cách các thuật toán lý thuyết được hiện thực hóa trong chương trình, cũng như cách người dùng tương tác với ứng dụng để thực hiện các phép tính giải bài toán.Bằng việc kết hợp *MATLAB App Designer*, ứng dụng được tổ chức theo dạng nhiều tab khác nhau về chức năng. Mỗi tab sẽ tương ứng với một vấn đề bài toán cụ thể, cho phép người dùng nhập dữ liệu đầu vào, lựa chọn phương pháp tính toán và quan sát kết quả thông qua kết quả số và đồ thị minh họa.

3.1 Giao diện tổng thể của ứng dụng

Ứng dụng được thiết kế theo bố cục nhiều tab khác nhau, trong đó mỗi tab tương ứng với một nhóm bài toán riêng biệt.



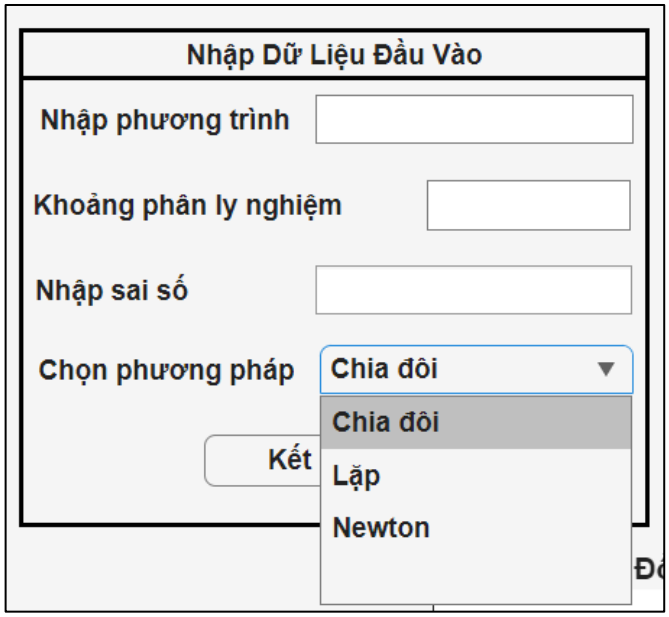
Hình 1. Giao diện ứng dụng

Trong mỗi tab, giao diện bao gồm các khu vực chính: nhập dữ liệu đầu vào, lựa chọn phương pháp tính toán, khung hiển thị kết quả và đồ thị minh họa. Các khu vực này sẽ giúp người dùng thuận tiện trong việc theo dõi quá trình xử lý từ dữ liệu ban đầu cho đến kết quả cuối cùng và hình ảnh trực quan tương ứng.

3.2 Thuật toán và luận lý

3.2.1 Nghiệm của phương trình

Với bài toán tìm nghiệm gần đúng của phương trình phi tuyến $f(x) = 0$, tab tìm nghiệm của ứng dụng sẽ cho phép nhập vào phương trình $f(x)$ cần tính nghiệm, nhập khoảng phân ly nghiệm [a b] và sai số tối đa ϵ và lựa chọn phương pháp tính toán theo yêu cầu:



Hình 2. Giao diện nhập dữ liệu

a) **Phương pháp chia đôi:** đảm bảo hội tụ khi hàm số liên tục và đổi dấu trên đoạn được xét.

- Các bước thực hiện:

1. Nhập hàm số $f(x)$ và khoảng phân ly nghiệm [a b]
2. Kiểm tra điều kiện đổi dấu của hàm số trên [a b]
3. Tính trung điểm

$$c = \frac{a + b}{2}$$

4. Nếu $|f(c)|$ nhỏ hơn sai số cho phép thì nhận c là nghiệm gần đúng.
5. Ngược lại, thu hẹp khoảng về [a c] hoặc [c b] sao cho hàm số đổi dấu.
6. Lặp lại cho đến khi đạt độ chính xác mong muốn.

- Cài đặt thuật toán:

```
function [root, iter] = Bisection(f, a, b, eps)

iter = 0;

if f(a)*f(b) > 0
    error('f(a) và f(b) phải trái dấu');
end

while (b - a)/2 > eps
    c = (a + b)/2;

    if f(c) == 0
        break;
    elseif f(a)*f(c) < 0
        b = c;
    else
        a = c;
    end
    iter = iter + 1;
end

root = (a + b)/2;
end
```

Code cell 1. Phương pháp chia đôi

b) **Phương pháp lặp đơn:** dựa vào việc xây dựng hàm lặp $x = g(x)$ phù hợp và đảm bảo điều kiện hội tụ.

- Các bước thực hiện:

1. Nhập hàm số $f(x)$, hàm lặp $g(x)$, giá trị khởi đầu x_0 và sai số ϵ
2. Kiểm tra điều kiện hội tụ của phương pháp với hàm lặp đã chọn:

$$q = \max_{x\in[a;b]} |g(x)| < 1$$

3. Thực hiện phép lặp theo công thức:

$$x_{k+1} = g(x_k)$$

4. Điều kiện dừng:

$$|x_{k+1} - x_k| < \epsilon$$

- Cài đặt thuật toán:

```
function [root, iter] = FixedPoint(g, x0, eps)

iter = 0;

while (1)
    x1 = g(x0);
    iter = iter + 1;

    if abs(x1 - x0) < eps
        break;
    end
    x0 = x1;
end

root = x1;
end
```

Code cell 2. Phương pháp lặp đơn

c) **Phương pháp tiếp tuyến (Newton – Raphson):** cho tốc độ hội tụ nhanh khi chọn giá trị ban đầu x_0 phù hợp.

- Các bước thực hiện:

1. Nhập hàm số $f(x)$, đạo hàm $f'(x)$, giá trị khởi đầu x_0 và sai số ϵ
2. Thực hiện lặp theo công thức:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

3. Điều kiện dừng:

$$|x_{k+1} - x_k| < \epsilon$$

- Cài đặt thuật toán:

```
function [root, iter] = NewtonMethod(f, df, x0, eps)

iter = 0;
while (1)
    if df(x0) == 0
        error('Đạo hàm bằng 0 – Newton không hội tụ');
    end

    x1 = x0 - f(x0)/df(x0);
    iter = iter + 1;

    if abs(x1 - x0) < eps
        break;
    end
    x0 = x1;
end

root = x1;
end
```

Code cell 3. Phương pháp tiếp tuyến

3.2.2 Nội suy

a) **Nội suy Lagrange:** cho phép xây dựng trực tiếp đa thức nội suy mà không cần giải hệ phương trình.

- Các bước thực hiện:

1. Nhập các giá trị đầu vào $x_i$và y_i, cùng với giá trị x cần tính nội suy
2. Với mỗi $i = 0,1, \dots, n$, xây dựng đa thức cơ sở $L_i(x)$theo công thức:

$$L_i(x) = \frac{(x - x_0)(x - x_1)(x - x_2) \dots (x_{x_n})}{(x_i - x_0)(x_i - x_1)(x_i - x_2) \dots (x_i - x_n)}$$

3. Nhân mỗi $L_i(x)$ với giá trị tương ứng y_i, cộng dồn để thu được đa thức nội suy $P_n(x)$:

$$P_n(x) = y_0 L_0(x) + y_1 L_1(x) + y_2 L_{2(x)} + \cdots + y_{n-1} L_{n-1}(x)$$

4. Xuất kết quả nội suy tại điểm x yêu cầu.

- Cài đặt thuật toán:

```
function [Px, poly_str] = lagrange_interp(X, Y, x_interp)
n = length(X);

if length(Y) ~= n
   error('Vectors X and Y must have the same length.');
end

if length(unique(X)) ~= n
   error('Duplicate x-values are not allowed in interpolation.');
end

Px = 0;

for i = 1:n
    Li = 1;
    for j = 1:n
        if i ~= j
           Li = Li *(x_interp - X(j))/(X(i) - X(j));
        end
    end
    Px = Px + Y(i)*Li;
end
```

Code cell 4. Nội suy Lagrange

b) **Nội suy Newton:**

- Các bước thực hiện:

1. Nhập các giá trị đầu vào $x_i$và y_i $(i = 0, 1, 2, 3, \dots, n)$
2. Lập bảng tỷ hiệu từ dữ liệu ban đầu và tính các hệ số
3. Xây dựng đa thức nội suy $P\_n(x)$ theo dạng Newton

- Cài đặt thuật toán:

```
function [Px, poly_str] = newton_interp(X, Y, x_interp)
n = length(X);

if length(Y) ~= n
   error('Vectors X and Y must have the same length.');
end

if length(unique(X)) ~= n
   error('Duplicate x-values found, resulting in zero division.');
end

D = zeros(n, n);
D(:, 1) = Y';

for j = 2:n
    for i = j:n
        denominator = X(i) - X(i-j+1);
        if denominator == 0
           error('Error in divided difference calculation: Zero denominator.');
        end
        D(i, j) = (D(i, j-1) - D(i-1, j-1))/denominator;
    end
end
```

Code cell 5. Nội suy Newton

3.2.3 Hồi quy

Trong tab hồi quy, người dùng nhập bảng giá trị $(x_i; y_i)$, chọn mô hình hồi quy để ứng dụng tính toán các tham số của hàm hồi quy theo phương pháp bình phương tối thiểu. Đồng thời, đồ thị hàm hồi quy và dữ liệu thực nghiệm được hiển thị, giúp dễ dàng so sánh và đánh giá mức độ phù hợp của mô hình thông qua các chỉ tiêu như sai số và hệ số tương quan r^2.

a) **Mô hình hồi quy tuyến tính:** $y = ax + b$ là mô hình hồi quy cơ bản và được sử dụng phổ biến để mô tả mối quan hệ gần đúng giữa biến phụ thuộc $y$và biến độc lập $x$thông qua một hàm bậc nhất.

- Các bước thực hiện:

1. Nhập dữ liệu các cặp giá trị $(x_i; y_i)$
2. Tính các tổng cần sử dụng:

$$\sum_{i=0}^{n} x_i ; \sum_{i=0}^{n} y_i ; \sum_{i=0}^{n} x_i^2 ; \sum_{i=0}^{n} x_i y_i$$

3. Tính hệ số a và b của đường hồi quy $y = ax + b$ theo công thức bình phương tối thiểu:

$$a = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}$$

$$b = \frac{\sum_{i=1}^{n} y_i - a\sum_{i=1}^{n} x_i}{n}$$

4. Xác định hệ số tương quan r^2 để đánh giá mức độ khớp của mô hình:

$$r^2 = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2 - \sum_{i=1}^{n}(y_i - ax_i - b)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \ (0 \le r^2 \le 1)$$

- Cài đặt thuật toán:

```
function [a0, a1, r2] = HoiQuyTuyenTinh(x,y)

n = length(x);
sum_x = 0;
sum_y = 0;
sum_xy = 0;
sum_x_binh=0;
sum_Sr = 0;
sum_St = 0;

for i=1:n
   sum_xy=sum_xy+ x(i)*y(i);
   sum_x = sum_x + x(i);
   sum_y = sum_y + y(i);
   sum_x_binh = sum_x_binh + (x(i))^2;
end

binh_sumx = sum_x^2;
a1 = (n*sum_xy - sum_x * sum_y) / (n*sum_x_binh - binh_sumx);
y_trungbinh = sum_y / n;
x_trunbinh = sum_x /n;
a0 = y_trungbinh - a1*x_trunbinh;

for i=1:n
    sum_Sr = sum_Sr + (y(i) - a0 -a1*x(i))^2;
    sum_St = sum_St + (y(i) - y_trungbinh)^2;
end
r2 = (sum_St - sum_Sr)/sum_St;
end
```

Code cell 6. Hồi quy tuyến tính

b) **Mô hình hồi quy phi tuyến $y = ae^{bx}$ và $y = ax^b$:**

- Các bước thực hiện:

1. Nhập dữ liệu đầu vào $(x_i; y_i)$, với $y_i > 0$.
2. Biến đổi dữ liệu:

$y = ae^{bx}$	$y = ax^b$
$\rightarrow lny = lna + bx$ $\rightarrow Y = A + bx$	$\rightarrow lny = lna + blnx$ $\rightarrow Y = A + bX$

3. Áp dụng phương pháp bình phương tối thiểu cho mô hình tuyến tính
4. Tính hệ số và xác định mô hình hồi quy

- Cài đặt thuật toán:

```
function [a, b, r2] =
HoiQuyPhiTuyen_Ham_e_mu(x,y)

n = length(x);
X = log(x);
Y = log(y);
sum_X = 0;
sum_Y = 0;
sum_XY = 0;
sum_X_binh=0;

for i=1:n
   sum_XY=sum_XY+ X(i)*Y(i);
   sum_X = sum_X + X(i);
   sum_Y = sum_Y + Y(i);
   sum_X_binh = sum_X_binh + (X(i))^2;
end

binh_sumX = sum_X^2;
A1 = (n*sum_XY - sum_X *
sum_Y)/(n*sum_X_binh - binh_sumX);
Y_trungbinh = sum_Y/n;
X_trungbinh = sum_X/n;
A0 = Y_trungbinh - A1*X_trungbinh;
a = exp(A0);
b=A1;
st=0;
sr=0;
for i=1:n
    st = st + (Y(i) - Y_trungbinh)^2;
    sr = sr + (Y(i) - (A0 + A1 * X(i)))^2;
end
r2 = (st - sr)/st;
end
```

Code cell 7. Hồi quy phi tuyến hàm mũ

```
function [a, b, r2] =
HoiQuyPhiTuyen_HamMu(x,y)

n = length(x);
X = log10(x);
Y = log10(y);
sum_X = 0;
sum_Y = 0;
sum_XY = 0;
sum_X_binh=0;

for i=1:n
   sum_XY=sum_XY+ X(i)*Y(i);
   sum_X = sum_X + X(i);
   sum_Y = sum_Y + Y(i);
   sum_X_binh = sum_X_binh + (X(i))^2;
end

binh_sumX = sum_X^2;
A1 = (n*sum_XY - sum_X * sum_Y)/(n*sum_X_binh
- binh_sumX);
Y_trungbinh = sum_Y / n;
X_trungbinh = sum_X /n;
A0 = Y_trungbinh - A1*X_trungbinh;
a = 10^A0;
b=A1;
st=0;
sr=0;
for i=1:n
    st = st + (Y(i) - Y_trungbinh)^2;
    sr = sr + (Y(i) - (A0 + A1 * X(i)))^2;
end
r2 = (st - sr)/st;
end
```

Code cell 8. Hồi quy phi tuyến hàm e mũ

3.2.4 Đạo hàm:

Các công thức sai phân dựa trên khai triển Taylor được sử dụng để tính gần đúng đạo hàm.

- Các bước thực hiện:

1. Chọn điểm x cần tính đạo hàm và bước nhảy h
2. Tính các giá trị tại $f(x)$, $f(x+h)$, $f(x-h)$
3. Áp dụng các công thức sai phân để tính xấp xỉ đạo hàm tại x:

$$\text{Newton xấp xỉ tiến: } f'(x) \approx \frac{f(x+h)-f(x)}{h}$$

$$\text{Newton xấp xỉ lùi: } f'(x) = \frac{f(x)-f(x-h)}{h}$$

$$\text{Newton xấp xỉ trung tâm: } f'(x) = \frac{f(x+h)-f(x-h)}{2h}$$

- Cài đặt thuật toán:

```
function result = tinh_newton_tien(f, x0, h, sai_so)

if strcmp(sai_so, 'O(h)')
   result = (f(x0 + h) - f(x0))/h;
else
   fx0 = f(x0);
   fx1 = f(x0 + h);
   fx2 = f(x0 + 2*h);
   result = (-fx2 + 4*fx1 - 3*fx0)/(2*h);
end
end
```

Code cell 9. Newton tiến

```
function result = tinh_newton_lui(f, x0, h, sai_so)

if strcmp(sai_so, 'O(h)')
   result = (f(x0) - f(x0 - h))/h;
else
   fx0 = f(x0);
   fx_minus1 = f(x0 - h);
   fx_minus2 = f(x0 - 2*h);
   result = (3*fx0 - 4*fx_minus1 + fx_minus2)/(2*h);
end
end
```

Code cell 10. Newton lùi

```
function result = tinh_trung_tam(f, x0, h, sai_so)

if strcmp(sai_so, 'O(h)')
   result = (f(x0 + h) - f(x0 - h))/(2*h);
else
   term1 = -f(x0 + 2*h);
   term2 = 8 * f(x0 + h);
   term3 = -8 * f(x0 - h);
   term4 = f(x0 - 2*h);
   result = (term1 + term2 + term3 + term4)/(12*h);
end
end
```

Code cell 11. Newton trung tâm

3.2.5 Tích phân

a) **Phương pháp hình thang:** xấp xỉ đồ thị của hàm số trên mỗi đoạn con bằng một đoạn thẳng, từ đó diện tích dưới đường cong được xấp xỉ bằng tổng diện tích các hình thang.

- Các bước thực hiện:

1. Chia đoạn $[a; b]$ thành N đoạn con bằng nhau
2. Tính các giá trị $f(x_i)$ tại các nút $x_i = a + ih$
3. Áp dụng công thức hình thang để tính gần đúng giá trị tích phân:

$$I = \int_a^b f(x)dx \approx \frac{\mathrm{h}}{2}\left[\mathrm{f}(a) + f(b) + 2\sum_{\mathrm{i}=1}^{\mathrm{N}-1}\mathrm{f}(a + ih)\right]$$

- Cài đặt thuật toán:

```
function I = hinhthang_xy(x, y)
n = length(x);
I = 0;

for i = 1:(n - 1)
    I = I + (y(i) + y(i + 1))*(x(i + 1) - x(i))/2;
end
end
```

Code cell 12. Phương pháp hình thang (x;y)

```
function I = hinhthang_f(f, a, b, N)

h = (b - a)/N;
s = f(a) + f(b);

for k = 1:(N - 1)
    xk = a + k*h;
    s = s + 2*f(xk);
end

I = (h/2)*s;
end
```

Code cell 13. Phương pháp hình thang f(x)

b) **Phương pháp Simpson 1/3:** xấp xỉ hàm số bằng các đa thức bậc hai trên từng cặp đoạn con liên tiếp.

1. Chia đoạn $[a; b]$ thành N đoạn con (với N chẵn)
2. Tính giá trị $f(x_i)$ tại các nút chia
3. Tính tổng các giá trị tại vị trí lẻ và chẵn
4. Áp dụng công thức Simpson 1/3 để tính gần đúng giá trị tích phân:

$$I = \int_a^b f(x)dx \approx \frac{h}{3}\left[f(a) + f(b) + 4\sum_{i\ lẻ}^{N-1} f(a+ih) + 2\sum_{i\ chẵn}^{N-1} f(a+ih)\right]$$

- Cài đặt thuật toán:

```
function I = simpson13(f, a, b, N)

if mod(N, 2) ~= 0
   uialert(app.UIFigure, "N phải là số chẵn để áp dụng Simpson 1/3", "Lỗi");
end

h = (b - a)/N;
sum1 = 0;
sum2 = 0;

for k = 1:2:(N - 1)
    sum1 = sum1 + f(a + k*h);
end

for k = 2:2:(N - 2)
    sum2 = sum2 + f(a + k*h);
end

I = (h/3)*(f(a) + f(b) + 4*sum1 + 2*sum2);
end
```

Code cell 14. Phương pháp Simpson 1/3

c) **Phương pháp Simpson 3/8:** xấp xỉ hàm số bằng đa thức bậc ba trên từng nhóm ba đoạn con liên tiếp.

1. Chia đoạn $[a; b]$ thành N đoạn con (với N chia hết cho 3)
2. Tính giá trị $f(x_i)$ tại các nút chia
3. Tính tổng các giá trị tại vị trí chia hết cho 3 và không chia hết cho 3
4. Áp dụng công thức Simpson 3/8 để tính gần đúng giá trị tích phân:

$$I = \int_a^b f(x)dx \approx \frac{h}{3}\left[f(a) + f(b) + 3\sum_{i\neq 0(mod3)}^{N-1} f(a+ih) + 2\sum_{i=0(mod3)}^{N-1} f(a+ih)\right]$$

- Cài đặt thuật toán:

```
function I = simpson38(f, a, b, N)
if mod(N, 3) ~= 0
   uialert(app.UIFigure, "N phải chia hết cho 3 để áp dụng Simpson 3/8", "Lỗi");
end

h = (b - a)/N;
sum3 = 0;
sum2 = 0;

for k = 1:(N - 1)
    xk = a + k*h;
    if mod(k, 3) == 0
        sum2 = sum2 + f(xk);
    else
        sum3 = sum3 + f(xk);
    end
end

I = (3*h/8)*(f(a) + f(b) + 3*sum3 + 2*sum2);
end
```

Code cell 15. Phương pháp Simpson 3/8

CHƯƠNG 4: THỬ NGHIỆM & KIỂM TRA KẾT QUẢ

4.1 Chức năng tìm nghiệm gần đúng của phương trình

4.1.1 Phương pháp chia đôi

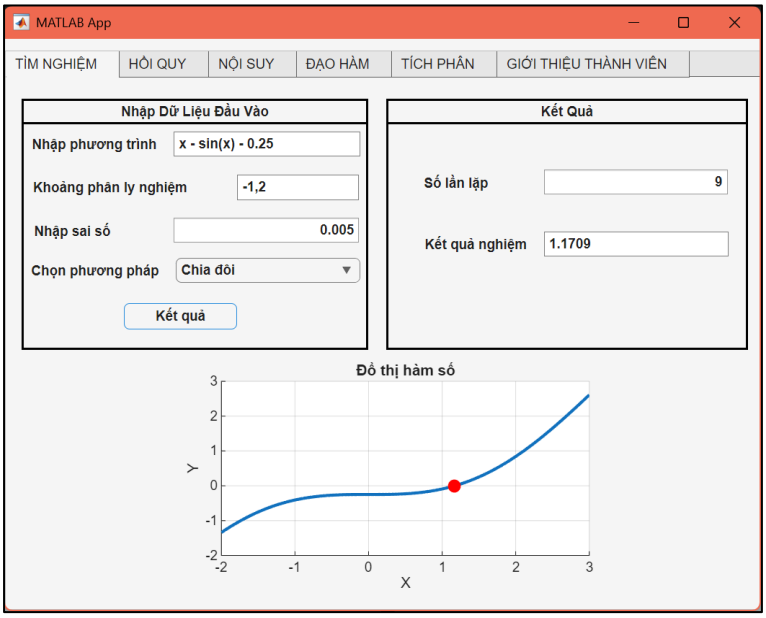
- Hàm số:

$$f(x) = x - sin(x) - 0.25$$

- Khoảng phân ly nghiệm: $[-1,2]$

- Sai số: $\epsilon = 0.005$

- Kết quả:



Số lần lặp: 9 lần
Nghiệm gần đúng ≈ 1.1709

Bảng 1. Test case tìm nghiệm

4.1.2 Phương pháp lặp đơn

- Hàm số:

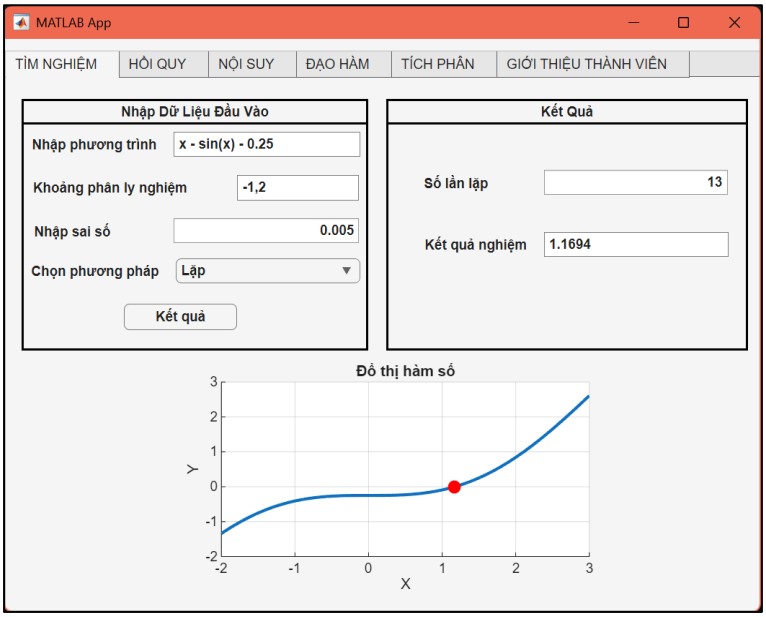
$$f(x) = x - sin(x) - 0.25$$

- Hàm lặp $x = g(x)$:

$$x = g(x) = sin(x) + 0.25$$

- Sai số: $\epsilon = 0.005$

- Kết quả:



Số lần lặp: 13 lần
Nghiệm gần đúng ≈ 1.1694

Bảng 2. Test case tìm nghiệm 2

4.1.3 Phương pháp tiếp tuyến (Newton – Raphson)

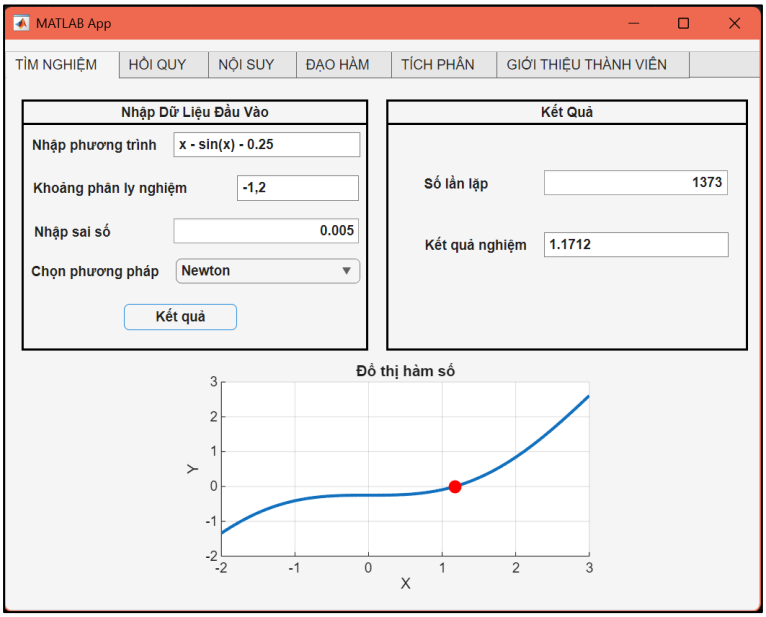
- Hàm số:

$$f(x) = x - sin(x) - 0.25$$

- Khoảng phân ly nghiệm: $[-1,2]$

- Sai số: $\epsilon = 0.005$

- Kết quả:



Số lần lặp: 1373 lần
Nghiệm gần đúng ≈ 1.1712

Bảng 3. Test case tìm nghiệm 3

4.2 Chức năng nội suy và hồi quy hàm số

4.2.1 Hồi quy tuyến tính hàm số

- Dữ liệu đầu vào x_i:

$$\boldsymbol{x = [1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9]}$$

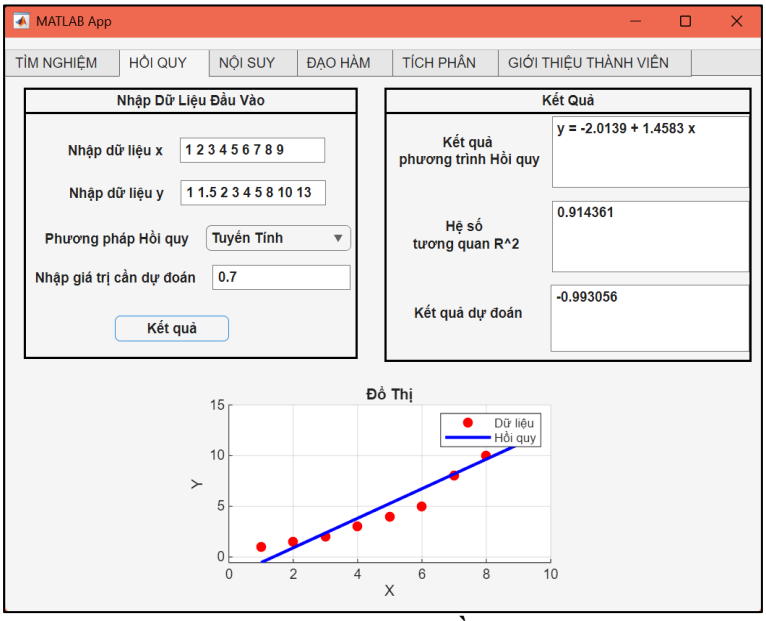
- Dữ liệu đầu vào y_i:

$$y = [1\ 1.5\ 2\ 3\ 4\ 5\ 8\ 10\ 13]$$

- Giá trị cần dự đoán:

$$x_{predict} = 0.7$$

- Kết quả:



Phương trình hồi quy

$$y = -2.0139 + 1.4583x$$

Kết quả dự đoán

$$y(0.7) \approx -0.993056$$

Bảng 4. Test case hồi quy 1

4.2.2 Hồi quy phi tuyến (hàm mũ):

- Dữ liệu đầu vào x_i:

$$x = [2.5\ 3.5\ 5\ 6\ 7.5\ 10\ 12.5\ 15\ 17.5\ 20]$$

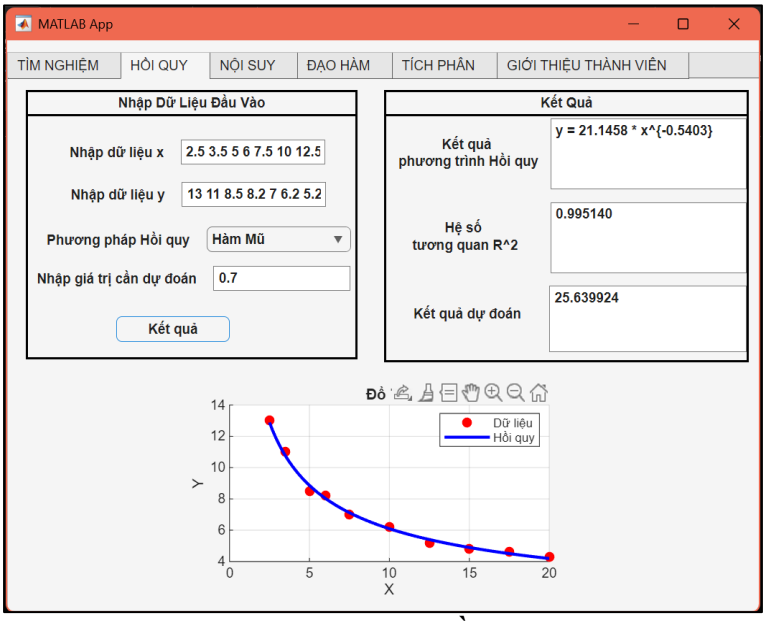
- Dữ liệu đầu vào y_i:

$$y = [13\ 11\ 8.5\ 8.2\ 7\ 6.2\ 5.2\ 4.8\ 4.6\ 4.3]$$

- Giá trị cần dự đoán:

$$x_{predict} = 0.7$$

- Kết quả:



Phương trình hồi quy $y = ax^b$

$$y = 21.1458x^{-0.5403}$$

Kết quả dự đoán

$$y(0.7) \approx 25.639924$$

Bảng 5. Test case hồi quy 2

4.2.3 Hồi quy phi tuyến (hàm e mũ):

- Dữ liệu đầu vào x_i:

$$\text{x} = [2.5\ 3.5\ 5\ 6\ 7.5\ 10\ 12.5\ 15\ 17.5\ 20]$$

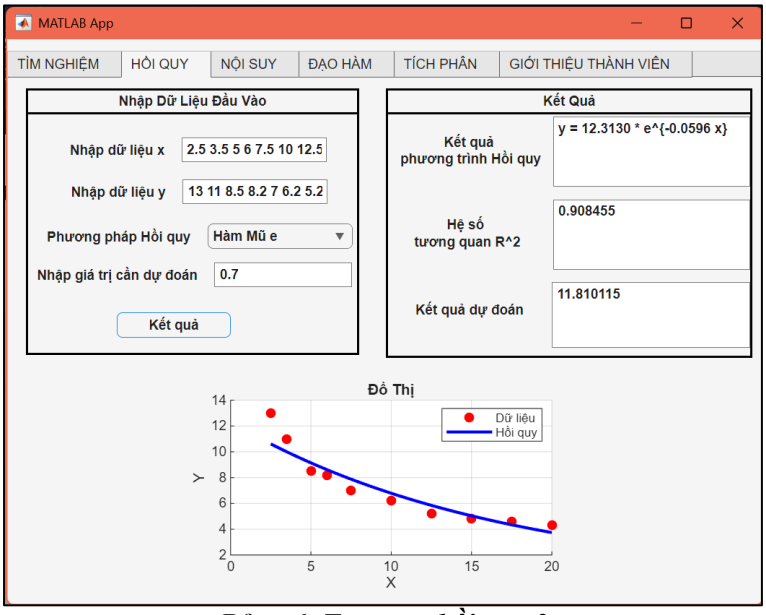
- Dữ liệu đầu vào y_i:

$$y = [13\ 11\ 8.5\ 8.2\ 7\ 6.2\ 5.2\ 4.8\ 4.6\ 4.3]$$

- Giá trị cần dự đoán:

$$x_{predict} = 0.7$$

- Kết quả:



Phương trình hồi quy $y = ae^{bx}$

$$y = 12.3130e^{-0.0596x}$$

Kết quả dự đoán

$$y(0.7) \approx 11.810115$$

Bảng 6. Test case hồi quy 3

4.3 Chức năng tính gần đúng đạo hàm

4.3.1 Đạo hàm điểm

- Dữ liệu đầu vào x_i:

$$\text{x} = [0.1\ 0.3\ 0.5\ 0.7\ 0.9]$$

- Dữ liệu đầu vào y_i:

$$y = [0.1002\ 0.3047\ 0.5236\ 0.7754\ 1.1198]$$

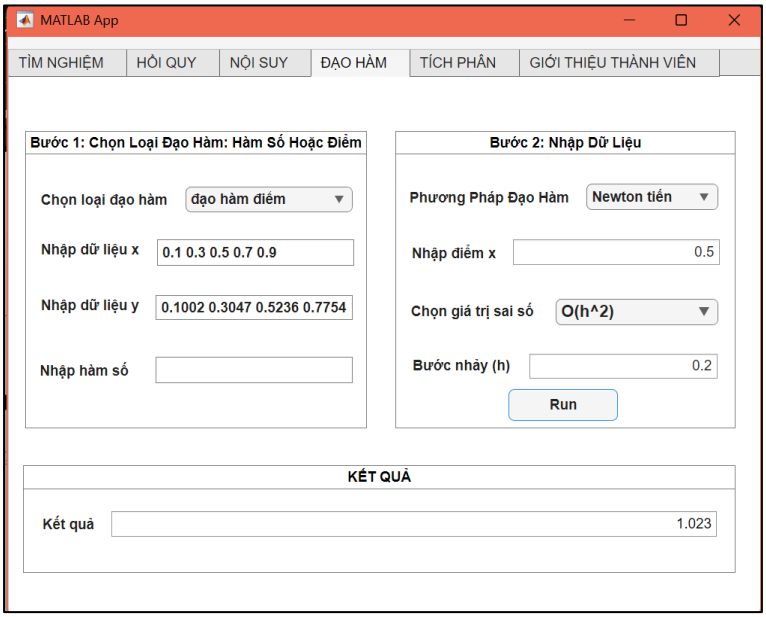
- Điểm cần xấp xỉ đạo hàm:

$$x_0 = 0.5$$

- Bậc sai số:

$$O(h^2)$$

- Kết quả:



Kết quả đạo hàm tại $x_0 = 0.5$

$f'(x_0) \approx 1.023$

Bảng 7. Test case đạo hàm 1

4.3.2 Đạo hàm hàm số

- Hàm số:

$$f(x) = x^3 sin(x)$$

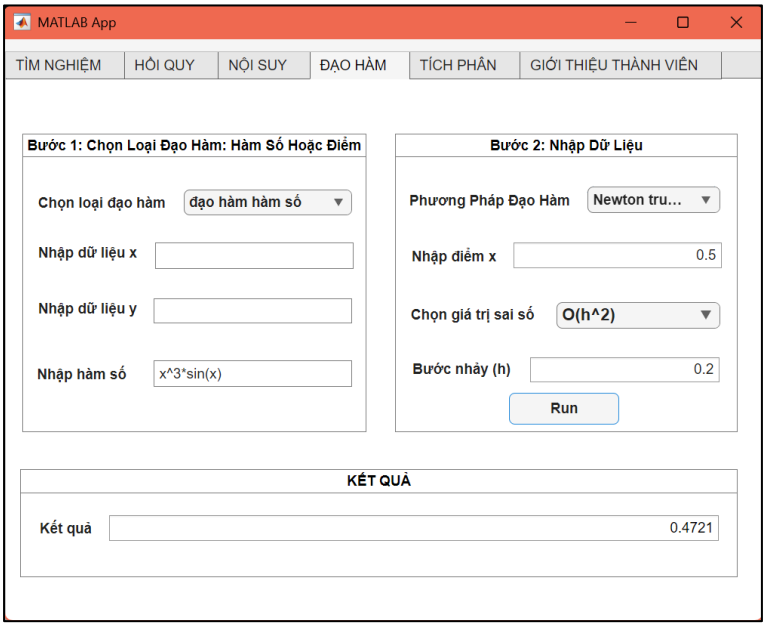
- Điểm cần xấp xỉ đạo hàm:

$$x_0 = 0.5$$

- Bậc sai số:

$$O(h^2)$$

- Kết quả:



Kết quả đạo hàm tại $x_0 = 0.5$
$f'(x_0) \approx 0.4721$

Bảng 8. Test case đạo hàm 2

4.4 Chức năng tính gần đúng tích phân

4.4.1 Phương pháp hình thang (mảng rời rạc x_i và y_i)

- Dữ liệu đầu vào x_i:

$$\mathrm{x} = [0.1\ 0.3\ 0.5\ 0.7\ 0.9]$$

- Dữ liệu đầu vào y_i:

$$y = [0.1002\ 0.3047\ 0.5236\ 0.7754\ 1.1198]$$

- Cận dưới và cận trên:

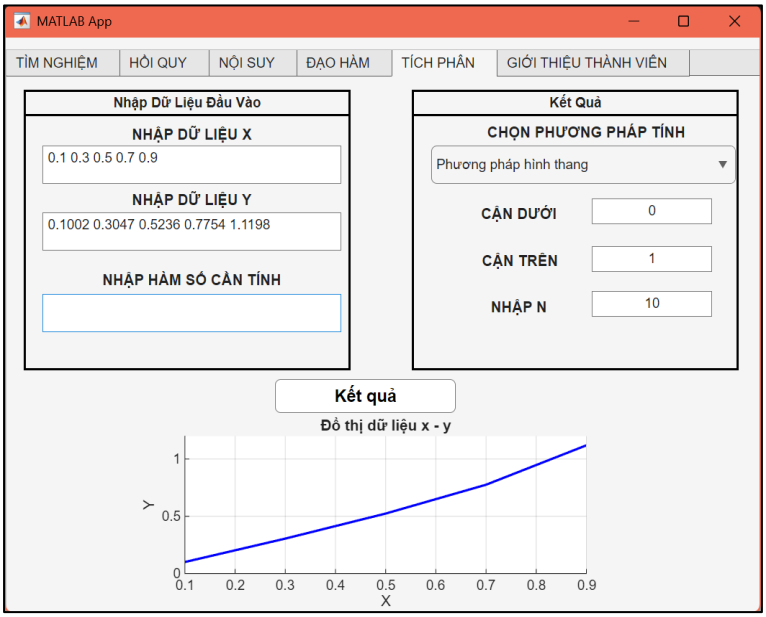
$$Cận\ dưới\ = 0$$

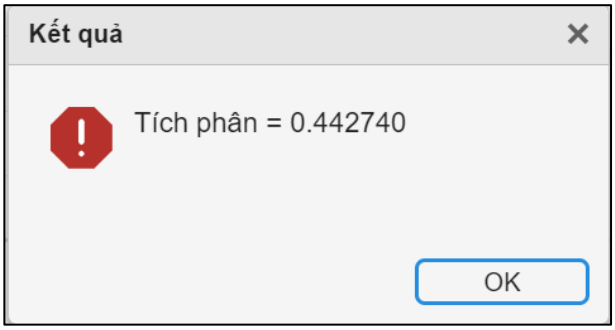
$$Cận\ trên = 1$$

- Số đoạn con N:

$$N = 10$$

- Kết quả:





Bảng 9. Test case tích phân 1

4.4.2. Phương pháp hình thang (hàm số)

- Hàm số:

$$f(x) = x^3 sin(x)$$

- Cận dưới và cận trên:

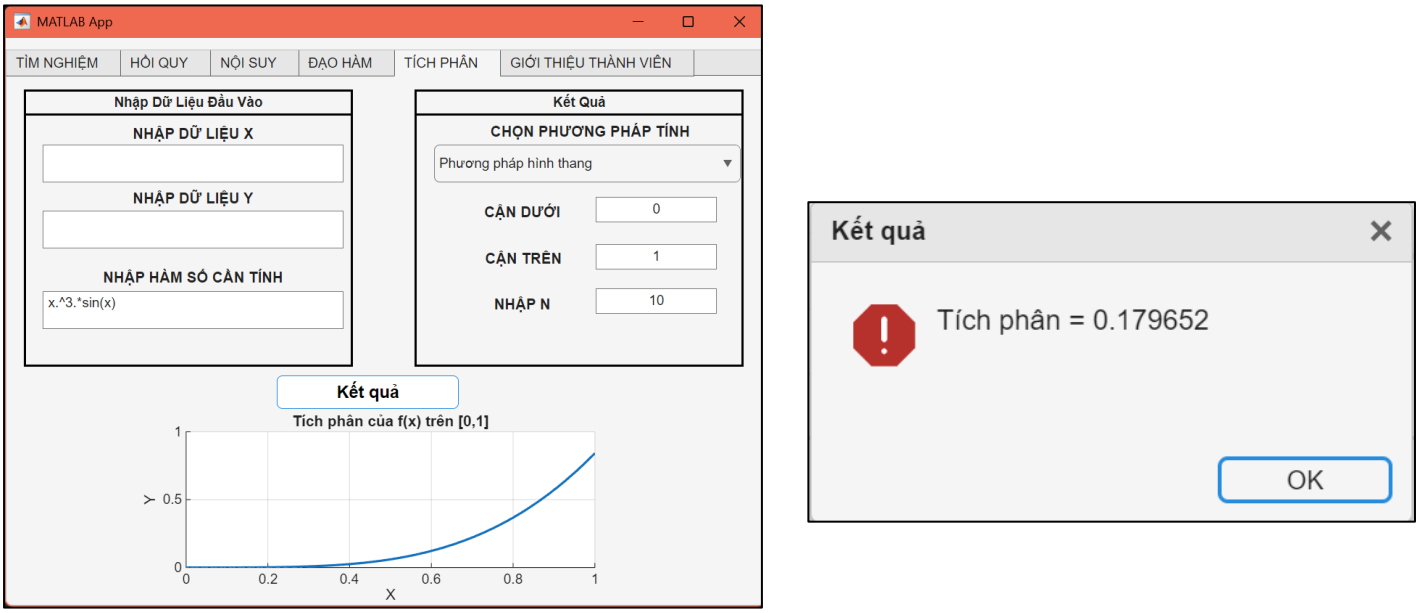
$$Cận\ dưới\ =\ 0$$

$$Cận\ trên = 1$$

- Số đoạn con N:

$$N = 10$$

- Kết quả:



Bảng 10. Test case tích phân 2

4.4.3 Phương pháp Simpson 1/3

- Hàm số:

$$f(x) = x^3 sin(x)$$

- Cận dưới và cận trên:

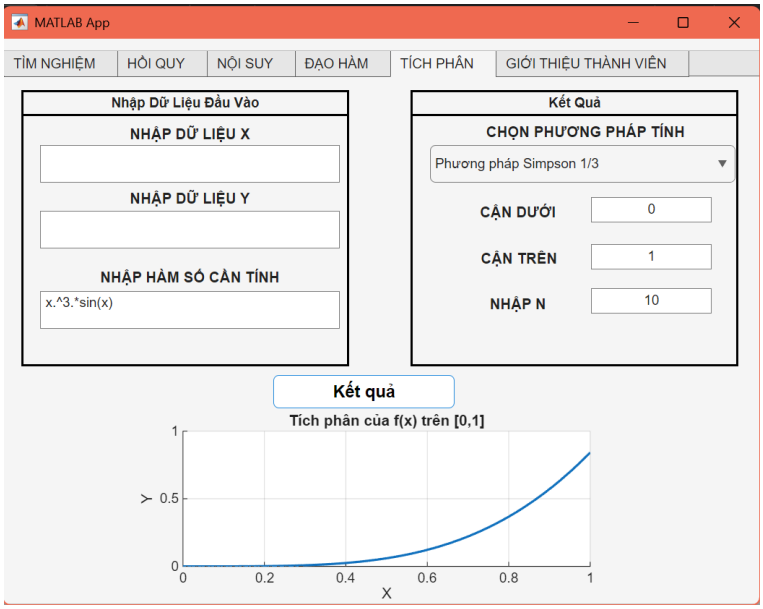
$$Cận\ dưới\ =\ 0$$

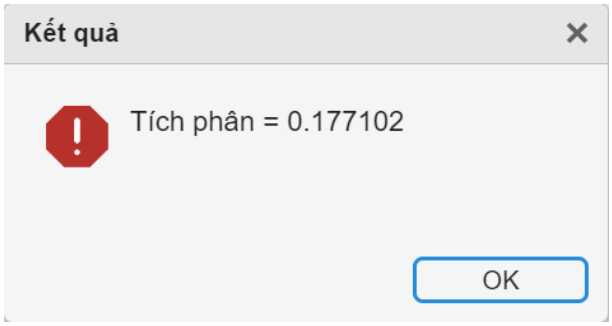
$$Cận\ trên = 1$$

- Số đoạn con N:

$$N = 10$$

- Kết quả:





Bảng 11. Test case tích phân 3

4.4.4 Phương pháp Simpson 3/8

- Hàm số:

$$f(x) = x^3 sin(x)$$

- Cận dưới và cận trên:

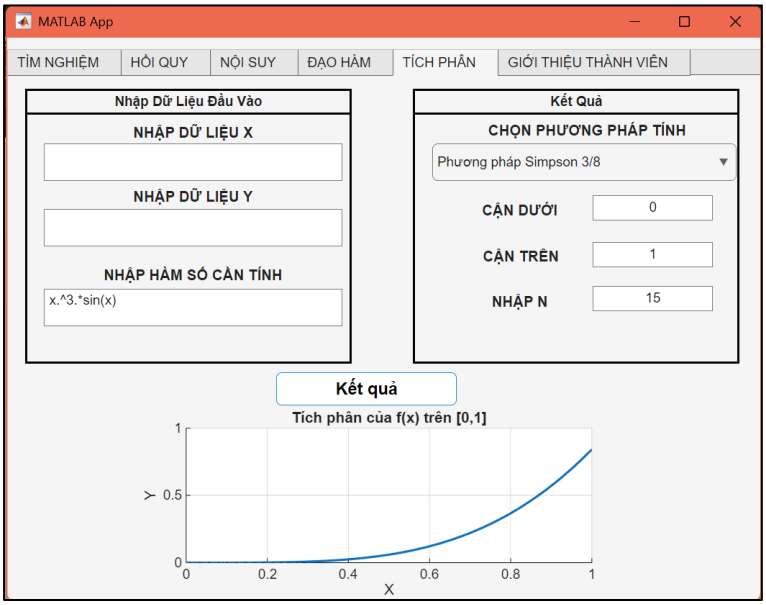
$$Cận\ dưới\ =\ 0$$

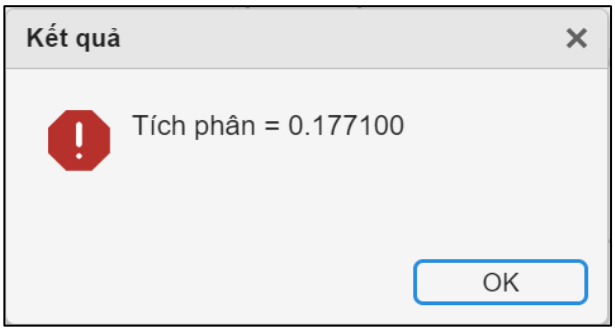
$$Cận\ trên = 1$$

- Số đoạn con N:

$$N = 15$$

- Kết quả:





Bảng 12. Test case tích phân 4

CHƯƠNG 5: NHỮNG HẠN CHẾ & HƯỚNG PHÁT TRIỂN

5.1 Hạn chế cần khắc phục

Mặc dù ứng dụng đã thực hiện được các chức năng chính như tìm nghiệm phương trình, nội suy, hồi quy, tính gần đúng đạo hàm và tích phân, tuy nhiên trong quá trình xây dựng và thử nghiệm, nhóm vẫn gặp một số hạn chế nhất định như sau:

Thứ nhất, khả năng nhập hàm còn bị giới hạn. Ứng dụng hiện tại chỉ hỗ trợ các biểu thức toán học cơ bản, chưa xử lý tốt các hàm phức tạp như hàm chứa căn bậc hai, giá trị tuyệt đối hoặc các hàm từng đoạn trong một số tab (đặc biệt là tab tìm nghiệm theo phương pháp lặp đơn).

Thứ hai, chưa có cơ chế kiểm tra lỗi đầu vào đầy đủ. Khi người dùng nhập sai định dạng dữ liệu (ví dụ nhập ký tự không phải số, hoặc chọn số bước N không phù hợp với phương pháp Simpson), ứng dụng có thể báo lỗi hoặc cho kết quả không mong muốn.

Thứ ba, độ chính xác phụ thuộc nhiều vào tham số người dùng chọn, chẳng hạn như số bước chia N, sai số cho phép ε hay khoảng xét nghiệm ban đầu. Ứng dụng chưa có chức năng tự động gợi ý hoặc tối ưu các tham số này.

Cuối cùng, giao diện vẫn còn đơn giản, chưa có nhiều hướng dẫn trực tiếp trên từng tab, có thể gây khó khăn cho người dùng mới khi sử dụng lần đầu.

5.2 Đề xuất hướng phát triển trong tương lai

Trong thời gian tới, nếu có điều kiện tiếp tục phát triển, ứng dụng có thể được mở rộng và cải tiến theo các hướng khác nhau. Thứ nhất, mở rộng khả năng xử lý biểu thức toán học, cho phép nhập các hàm phức tạp hơn như căn, logarit nhiều biến, hàm từng đoạn và kiểm tra miền xác định của hàm số trước khi tính toán. Thứ hai, bổ sung cơ chế kiểm tra và cảnh báo lỗi thông minh, giúp người dùng nhận biết lỗi ngay khi nhập dữ liệu không hợp lệ, đồng thời đưa ra gợi ý điều chỉnh phù hợp.

Về hình thức cần cải thiện độ thân thiện của giao diện, bổ sung mô tả ngắn cho từng phương pháp, minh họa thuật toán bằng hình ảnh hoặc sơ đồ để người dùng dễ hiểu hơn. Bên cạnh đó, việc tối ưu thuật toán và tự động lựa chọn tham số, ví dụ tự đề xuất số bước chia phù hợp cho Simpson hoặc tự kiểm tra điều kiện hội tụ của phương pháp lặp cũng cần được bổ sung thêm. Cuối cùng, ứng dụng có thể được phát triển

thêm theo hướng đóng gói thành phần mềm độc lập hoặc tích hợp thêm các phương pháp số nâng cao để phục vụ tốt hơn cho việc học tập và nghiên cứu.

CHƯƠNG 6: PHÂN CÔNG NHIỆM VỤ & ĐÁNH GIÁ KẾT QUẢ

6.1 Phân công nhiệm vụ

Nhóm thực hiện đồ án gồm 4 thành viên, mỗi thành viên đảm nhận một phần công việc cụ thể nhằm đảm bảo tiến độ và chất lượng của bài báo cáo.

Sinh viên Nguyễn Trần Hoàng Phúc:

- Nghiên cứu, tổng hợp cơ sở lý thuyết, thuật toán và code cho chủ đề **"Tìm nghiệm gần đúng của phương trình".**
- Thiết kế giao diện và quá trình hoạt động của tab **"Tìm nghiệm gần đúng của phương trình".**

Sinh viên Giao Hữu Lực:

- Nghiên cứu, tổng hợp cơ sở lý thuyết, thuật toán và code cho chủ đề "**Tính gần đúng tích phân".**
- Thiết kế giao diện và quá trình hoạt động của tab "**Tính gần đúng tích phân".**
- Chịu trách nhiệm viết và trình bày báo cáo của dồ án.

Sinh viên Đặng Bá Trần Trung:

- Nghiên cứu, tổng hợp cơ sở lý thuyết, thuật toán và code cho chủ đề "**Hồi quy hàm số"** + "**Tính gần đúng đạo hàm của hàm số".**
- Thiết kế giao diện và quá trình hoạt động của tab "**Hồi quy hàm số"** + "**Tính gần đúng đạo hàm của hàm số".**
- Chịu trách nhiệm thiết kế giao diện ứng dụng và tổng hợp nội dung.

Sinh viên Lê Quang Thông:

- Nghiên cứu, tổng hợp cơ sở lý thuyết, thuật toán và code cho chủ đề **"Nội suy hàm số".**
- Thiết kế giao diện và quá trình hoạt động của tab "**Nội suy".**

6.2 Đánh giá kết quả và sự đóng góp

Dựa trên khối lượng công việc và mức độ hoàn thành, nhóm tiến hành tự đánh giá mức độ đóng góp của các thành viên như sau:

STT	Thành viên	Nội dung đóng góp	Tỷ lệ đóng góp (%)
1	Nguyễn Trần Hoàng Phúc	Giao diện, thuật toán phần "**Tìm nghiệm gần đúng"**	25%
2	Giao Hữu Lực	Viết báo cáo, giao diện	25%
3	Đặng Bá Trần Trung	Tổng hợp tài liệu, giao diện	25%
4	Lê Quang Thông	Giao diện, thuật toán phần "**Nội suy hàm số"**	25%
Tổng			**100%**

Bảng 13. Đóng góp từng cá nhân

Nhóm đánh giá rằng các thành viên đã phối hợp làm việc nghiêm túc, đúng tiến độ và hỗ trợ lẫn nhau trong quá trình thực hiện đề tài. Các nhiệm vụ được phân công rõ ràng và hoàn thành đúng kế hoạch đề ra. Mặc dù vẫn còn một số hạn chế về mặt thời gian và phạm vi kiến thức, nhưng nhóm đã cố gắng hoàn thành đầy đủ các yêu cầu của đề tài và đạt được mục tiêu ban đầu.

KẾT LUẬN

Trong đồ án này, nhóm đã xây dựng thành công một ứng dụng MATLAB dạng nhiều tab, hỗ trợ giải quyết các bài toán số học quan trọng như: tìm nghiệm phương trình, nội suy, hồi quy, tính gần đúng đạo hàm và tích phân. Ứng dụng cho phép người dùng nhập dữ liệu linh hoạt, lựa chọn phương pháp phù hợp và trực quan hóa kết quả thông qua đồ thị.

Thông qua quá trình thực hiện, nhóm không chỉ củng cố kiến thức lý thuyết về các phương pháp số mà còn rèn luyện kỹ năng lập trình MATLAB, thiết kế giao diện bằng App Designer và tổ chức mã nguồn theo hướng có cấu trúc. Việc kiểm thử bằng nhiều test case giúp đánh giá được độ chính xác, ưu nhược điểm của từng phương pháp. Mặc dù ứng dụng vẫn còn một số hạn chế nhất định về khả năng xử lý hàm phức tạp và tối ưu tham số, nhưng nhìn chung đồ án đã đáp ứng được các yêu cầu đề ra và có tính ứng dụng trong học tập. Đây là nền tảng để nhóm tiếp tục mở rộng và hoàn thiện ứng dụng trong các hướng phát triển sau này.

Người viết báo cáo

Giao Hữu Lực

PHỤ LỤC

A. NỀN TẢNG GITHUB HỖ TRỢ QUẢN LÝ SOURCE CODE

A.1 Mục đích cần sử dụng nền tàng GitHub

Trong quá trình thực hiện đồ án, nhóm sử dụng GitHub để quản lý mã nguồn và phối hợp làm việc giữa các thành viên. Việc sử dụng Github giúp nhóm lưu trữ và quản lý phiên bản code tập trung tại một nơi. Bên cạnh đó có thể xem danh sách file được tải lên và theo dõi lịch sử chỉnh sửa của từng thành viên, giúp thuận tiện cho việc tổng hợp và hoàn thiện ứng dụng.

Hình 3. GITHUB

Các thành viên trong nhóm phân chia công việc và cập nhật mã nguồn lên GitHub theo từng phần chức năng. Mỗi lần chỉnh sửa đều được commit với nội dung rõ ràng, giúp việc tổng hợp và kiểm tra code diễn ra thuận lợi. Việc sử dụng Github không chỉ hỗ trợ hoàn thành đồ án hiệu quả hơn mà còn giúp nhóm làm quen với quy trình làm việc nhóm và quản lý mã nguồn trong các dự án thực tế.

A.2 Thông tin repository

- Nền tảng: GitHub
- Tên repository: MATLAB Project 2025
- Hình thức: Repository dành riêng cho đồ án
- Nội dung lưu trữ: file ‘.mlapp’ của ứng dụng, các file ‘.m’ chứa thuật toán của các phương pháp tính (chia đôi, lặp đơn, simpson,…) và tài liệu báo cáo, hình ảnh minh họa.
- Link GitHub: https://github.com/HuuLuc/MATLAB-Project-2025