**A picture containing rectangle

Description automatically generatedBỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỤ NHIÊN**

**ĐẠI HỌC QUỐC GIA TP HỒ CHÍ MINH**

**KHOA ĐIỆN TỬ - VIỄN THÔNG**



**A picture containing text, sign

Description automatically generated**A picture containing logo

Description automatically generated

**BÁO CÁO ĐỒ ÁN CUỐI KHÓA**

**MÔN PHƯƠNG PHÁP TÍNH VÀ MATLAB**

Học kỳ I – Năm học: 2022-2023

**Nguyễn Dũng MSSV: 20200121.**

**GVHD: Nguyễn Xuân Vinh.**

Thành phố Hồ Chí Minh, ngày 25 tháng 12 năm 2022

**MỤC LỤC**

I. PHẦN LÝ THUYẾT CÁC PHƯƠNG PHÁP SỬ DỤNG TRONG ĐỒ ÁN MÔN HỌC: 4

1. Chương 8: Giải gần đúng phương trình: 4

1.1. Các bước giải gần đúng phương trình : 4

1.2. Khoảng phân ly nghiệm: 4

1.3. Phương pháp lặp đơn: 5

1.4. Phương pháp tiếp tuyến (PP Newton-Raphson): 7

1.5. Phương pháp dây cung: 8

2. Chương 9: Giải hệ phương trình: 11

2.1. Dạng tổng quát của một hệ phương trình tuyến tính: 11

2.2. Sự tồn tại và duy nhất của nghiệm: 12

2.3. Phương pháp Gauss-Jordan: 12

3. Chương 10: Nội suy, hồi quy: 13

3.1. Các khái niệm: 13

3.2. Nội suy Newton (có mốc cách đều): 14

4. Chương 11 + 12: Đạo hàm, tích phân, Phương trình vi phân: 17

4.1. Bài toán tích phân: 17

4.2. Phương pháp hình thang: 18

II. PHẦN GIẢI CÁC BÀI TẬP THEO LÝ THUYẾT: 20

III. PHẦN THIẾT KẾ GIAO DIỆN: 34

1. Giới thiệu về App Designer trong Matlab: 34

2. Thiết kế giao diện: 35

2.1. Giao diện chính của giao diện: 36

2.2. Phần code thuật toán các phương pháp đã dùng để giải các bài tập ở phần II: 37

2.2.1 Giải phương trình: 37

2.2.2 Giải hệ phương trình: 40

2.2.3 Nội suy, Hồi quy: 40

2.2.4 Đạo hàm, tích phân, Phương trình vi phân: 41

2.3. Phần code giao diện: 42

3. Kết quả: 52

3.1. Giải phương trình: 53

3.1.1 Phương pháp lặp đơn: 57

3.1.2 Phương pháp tiếp tuyến (Newton – Raphson): 58

3.1.3 Phương pháp dây cung: 59

3.2. Giải hệ phương trình: 59

3.2.1 Phương pháp Gauss-Jordan: 62

3.3. Nội suy, Hồi quy: 64

3.3.1 Nội suy Newton: 67

3.4. Đạo hàm, tích phân, Phương trình vi phân: 67

3.4.1 Phương pháp hình thang: 70

4. Thiết lập một số lỗi trong giao diện: 70

5. Đánh giá sản phẩm: 74

5.1. Ưu điểm: 74

5.2. Nhược điểm: 74

6. Kết luận: 74

1. PHẦN LÝ THUYẾT CÁC PHƯƠNG PHÁP sử dụng trong đồ án môn học:
   1. Chương 8: Giải gần đúng phương trình:
      1. Các bước giải gần đúng phương trình :

Để tìm nghiệm gần đúng (thực) của phương trình ta thường tiến hành theo các bước sau:

1. Khẳng định phương trình có nghiệm trong một khoảng nào đó: **khoảng phân ly nghiệm .**
2. Chọn một xấp xỉ đầu , xem đó là nghiệm gần đúng đầu tiên thuộc khoảng đang xét .
3. Điều chỉnh dần sao cho càng gần tới nghiệm đúng α càng tốt.
4. Đánh giá sai số của nghiệm gần đúng tìm được so với nghiệm đúng.
   * 1. Khoảng phân ly nghiệm:

Xét phương trình

Nếu liên tục và đơn điệu trên khoảng đồng thời và trái dấu

**Định nghĩa:** Giả sử α là một nghiệm của phương trình (3.1). Ta nói [a, b] là khoảng phân ly nghiệm α của phương trình nếu trong khoảng đó phương trình chỉ có một nghiệm thực α duy nhất.

Diagram

Description automatically generated**Định lý 1:** Giả sử là hàm số liên tục trên tập . Khi đó là một khoảng phân ly nghiệm của phương trình (\*) nếu và đơn điệu trên

**Định lý 2:** Với giả thiết của định lý 1 thì là khoảng phàn ly nghiêm của phương trình nếu và tồn tại đồng thời giữ nguyên một dấu .

Từ định lý trên ta suy ra, nếu là khoảng phân ly nghiệm của phương trình thì mọi khoảng , sao cho cũng là khoảng phân ly nghiệm của phương trình trên (thu hep khoảng chứa nghiệm).

* + 1. Phương pháp lặp đơn:

Giả sử là khoảng phân ly nghiệm của phương trình .

Tiến hành viết lại phương trình dưới dạng: trong đó liên tục trong .

Chọn là điểm bất kỳ trong . Tính dãy lặp theo công thức sau (cho đến khi sai số thỏa mãn)

**Định lý:** Giả thuyết tồn tại số sao cho . Khi đó quá trình lặp hội tụ.

**Diagram, engineering drawing

Description automatically generated** **Minh họa hình học của phương pháp:**

**Sai số ước lượng:**

**Đánh giá:**

* Ưu điểm: xấp xỉ đầu không nhất thiết phải gần , có khả năng sửa sai, dễ lập trình.
* Nhược điểm: Khi gần 1 thì hội tụ rất chậm.

**Thuật toán:**

Từ phương trình đề cho:

**Bước 1:** Xác định sai số cho phép.

**Bước 2:** Xác định khoảng phân li nghiệm .

**Bước 3:** Tìm hàm lặp hội tụ (tức biến đổi x về vế trái, phần còn lại về vế phải chính là .

**Bước 4:** Chọn xấp xỉ ban đầu là .

**Bước 5:** Tính (lặp) giá trị ,với đến khi thì dừng.

**Bước 6:** Kết luận nghiệm = mới nhất và số lần lặp.

Sau lần lặp, được là nghiệm gần đúng của phương trình. Sai số được đánh giá theo công thức (hoặc khi thỏa điều kiện sai số tính toán sai số cho phép .

* + 1. Phương pháp tiếp tuyến (PP Newton-Raphson):

Chart, line chart

Description automatically generated**Ý tưởng chủ đạo của phương pháp này** là tìm cách thay phương trình , phi tuyến đối với , bằng một phương trình gần đúng, tuyến tính đối với .

Giả sử là khoảng phân ly nghiệm của phương trình liên tục với và không đổi dấu trên . Nghiệm đúng của phương trình là hoành độ giao điểm của đường cong với trục hoành .

**Nội dung phương pháp:**

Thay đường cong bằng tiếp tuyến kẻ từ hay coi hoành độ giao điểm của tiếp tuyến và là nghiệm gần đúng.

Đặt nếu tiếp tuyến kẻ từ , nếu tiếp tuyến kẻ từ . Phương trình tiếp tuyến tại có dạng như sau:

Ta lại có tiếp tuyến vẽ từ điểm , tương tự xấp xỉ tuyến tính tiếp theo có dạng:

Ta đó, ta có công thức tổng quát như sau:

**Điều kiện để thực hiện được phương pháp tiếp tuyến:**

Điều kiện để hội tụ là và không đổi dấu trên .

Xấp xỉ đầu được chọn là hay sao cho , nói cách khác cùng dấu với khi đó sẽ hội tụ.

**Đánh giá sai số:**

trong đó , hoặc

với .

Trên thực tế thường dừng quá trình tính toán khi

(sai số cho phép)

**Nhận xét:**

Phương pháp tiếp tuyến (PP Newton-Raphson) còn gọi là phương pháp tuyến tính hóa, hội tụ nhanh hơn so với phương pháp chia đôi và phương pháp lặp đơn. Tuy nhiên việc kiểm tra điều kiện để áp dụng phương pháp này phức tạp hơn.

**Tóm tắt thuật toán:**

**Bước 1:** Tính , và xét dấu của chúng.

**Bước 2:** Chọn được chọn là hay sao cho cùng dấu với

**Bước 3:** Từ xấp xỉ đầu , tính

Sau lần lặp ta thu được là nghiệm gần đúng của phương trình.

* + 1. Phương pháp dây cung:

**Ý tưởng:** Trong phương pháp tiếp tuyến, ta đã thay cung đồ thị AB của hàm bởi tiếp tuyễn vẽ tại A hay B. Bây giờ ta thay cung AB thành dây cung vẽ từ A đến B.

Giả sử là khoảng phân ly nghiệm của phương trình liên tục với và không đổi dấu trên .

Trong phương pháp tiếp tuyến ta thay đường cong bằng tiếp tuyến của đường cong. Trong phương pháp dây cung, ta thay bằng dây cung nối 2 điểm và . Nghiệm đúng của phương trình là hoành độ giao điểm của dây cung với trục hoành .

**Phương trình dây cung:**

Chọn đặt . Lặp lại quá trình trên, ta có:

**Chart

Description automatically generated with medium confidenceMinh họa hình học:**

**Điều kiện hội tụ:**

Giả sử là khoảng phân ly nghiệm của phương trình . Hàm số liên tục và có đạo hàm ; đồng thời các đạo hàm đó giử nguyên dấu trên đoạn . Xấp xỉ ban đầu được chọn sao cho

Khi đó dãy tính theo công thức ở trên sẽ đơn điệu hội tụ tới khi

.

**Đánh giá sai số:**

Được đánh giá theo công thức sau:

trong đó,

Để thì

**Nhận xét:** Phương pháp dây cung, hội tụ khá nhanh so với phương pháp chia đôi và phương pháp lặp đơn nhưng chậm hơn phương pháp tiếp tuyến (PP Newton-Raphson). Tuy nhiên việc kiểm tra điều kiện để áp dụng phương pháp này khá phức tạp.

**Tóm tắt thuật toán:**

**Bước 1:** Tính và , xét dấu của chúng.

Hội tụ khi và không đổi dấu trong

**Bước 2:** Kiểm tra điều kiện:

Để chọn xấp xỉ ban đầu là hay .

**Bước 3:** Từ xấp xỉ ban đầu , tính

hoặc,

Sau lần lặp như phương pháp chia đôi ta sẽ thu được là ngiệm gần đúng của phương trình.

* 1. Chương 9: Giải hệ phương trình:
     1. Dạng tổng quát của một hệ phương trình tuyến tính:

Một hệ phương trình tuyến tính có thể có phương trình ẩn.

Pham vị môn học chỉ xét hệ phương trình có phương trình ẩn như sau:

Trong đó là hệ số của ẩn ở phương trình thứ , là vế phải của phương trình thứ . Giả sử, ta đã biết và , ta cần tìm các giá trị ẩn

Từ hệ phương trình, ta có được các ma trận:

được gọi là ma trận hệ số, và là ma trận vuông.

là các vector hệ số và vector của ẩn.

Từ đó, ta chuyển hệ phương trình trên thành dạng ma trận như sau:

* + 1. Sự tồn tại và duy nhất của nghiệm:

**Sự dụng phương pháp Cramer:**

Xét một hệ phương trình , trong đó là ma trận vuông cấp . Để khả nghịch thì .

Khi đó, nghiệm duy nhất của hệ phương trình được tính theo công thức hoặc sử dụng công thức Cramer:

Table

Description automatically generated Trong đó và là định thức khi thay bởi cột thứ bằng cột .

Nhược điểm của phương pháp này là khi càng lớn thì thực hiện càng nhiều phép tính.

(Nguồn: Slide/74)

Có rất nhiều phương pháp khác để tìm nghiệm gần đúng của hệ phương trình. Một trong số là **phương pháp Gauss-Jordan**.

* + 1. Phương pháp Gauss-Jordan:

**Phương pháp khử Gauss-Jordan** dùng cách khử dẫn các ẩn đưa hệ phương trình đã cho về một dạng ma trận đường chéo rồi giải hệ phương trình này, chúng ta không cần tính một định thức nào cả. Phương pháp này là loại trừ ẩn nên thực chất là biến đổi hệ số của hệ phương trình để được hệ tương đương sao cho trong một phương trình của hệ mới chỉ còn lại một ấn số . Rồi sau đó tiến hành giải hệ phương trình.

Xét về khối lượng tính toán, đối với hệ phương trình ẩn, người ta dựa vào các công thức tính để đếm các phép nhân, chia. Số lượng các phép tính cụ thể như sau:

* Nhân và chia:
* Cộng và trừ:

Calendar

Description automatically generated**Ưu điểm:** giảm được số phép tính toánkhi giải hệ phương trình.

**Chi tiết các bước tiến hành sẽ được trình bày rỏ ở mục II (giải bài tập).**

* 1. Chương 10: Nội suy, hồi quy:
     1. Các khái niệm:

Diagram

Description automatically generated**Nội suy:** là phương pháp ước tính giá trị của các điểm dữ liệu chưa biết trong phạm vi của một tập hợp rời rạc chứa một số điểm dữ liệu đã biết, tức dự báo giá trị của biến phụ thuộc từ giá trị của biến độc lập nằm trong khoảng biến thiên của kết quả quan sát cho nó, kết quả dự báo đáng tin cậy hơn.

**Ngoại suy:** là phương pháp ước tính giá trị (tương lai) chưa biết dựa vào các giá trị (quá khứ) đã biết

.

Chart, scatter chart

Description automatically generated**Hồi quy:** Tìm một đường biểu diễn thể hiện khuynh hướng thay đổi của dữ liệu.

* + 1. Nội suy Newton (có mốc cách đều):

Xét bài toán nội suy có , các nút cách đều nhau với

**Bảng sai phân hưu hạn:**

Sai phân tiến:

được gọi là sai phân tiến cấp một.

Sai phân tiến cấp 2 là sai phân tiến của sai phân tiến cấp một và được ký hiệu như sau:

Tổng quát, sai phân tiến của sai phân tiến cấp là sai phân tiến cấp , được ký hiệu:

Sai phân lùi:

được gọi là sai phân lùi cấp một.

Sai phân lùi cấp là sai phân lùi của sai phân lùi cấp và được ký hiệu như sau:

Mối quan hệ giữa sai phân tiến và sai phân lùi

Diagram, schematic

Description automatically generated Từ đó, ta có bảng sai phân như sau:

Nguồn: Slide/134.

Các công thức:

Đa thức nội suy Newton tiến có mốc nội suy cách đều:

Đa thức nội suy Newton lùi có mốc nội suy cách đều:

Diagram

Description automatically generated with medium confidence

**Đánh giá sai số:**

**Chú ý:** Nếu cần tìm giá trị gần đúng tại điếm (ngoài bảng), trong trường hơp này

Nếu ta sử dụng công thức tiến.  
Nếu ta sử dụng công thức lùi.

* 1. Chương 11 + 12: Đạo hàm, tích phân, Phương trình vi phân:
     1. Bài toán tích phân:

Già sử là hàm số liên tuc trên đoan có nguyên hàm là , khi đó giá trị của tích phân xác đinh trên đoan đươc tính theo công thức Newton-Leibniz:

Tuy nhiên trong nhiều trường hợp có là nguyên hàm của hàm không phải là hàm số sơ cấp hoặc có biểu thức quá phức tạp. Khi đó giá trị của tích phân chỉ tìm được là số gần đúng. Trong trường hợp hàm số cho trong dạng bảng số thì khái niệm nguyên hàm không còn ý nghĩa gì nữa.

Vậy nên chúng ta có một số cách giải gần đúng như sau:

**Các phương pháp Newton ‐ Cotes** (khoảng lấy tích phân được chia đều: phương pháp hình thang, Simpson).

**Phương pháp Gauss:** các điểm chia được chọn để đạt độ chính xác cao nhất.

**Ý tưởng của phương pháp Newton – Cotes:** Chia thành đoạn bằng nhau có chiều dài và kí hiệu . Sau đó ta xấp xỉ hàm bằng đa thức bậc đi qua các nút. Thay các biểu thức hàm phức tạp hay các bảng dữ liệu bằng các hàm xấp xỉ dễ lấy tích phân.

trong đó,

Chart, histogram

Description automatically generated

* + 1. Phương pháp hình thang:

Phân đoạn thành đoạn con bằng nhau: với . Nối các điểm và bằng các đường thẳng

Chart

Description automatically generated

Khi đó: Giá trị tích phân là tổng diện tích các tích phân từng phần hình thang

Diagram

Description automatically generatedTiến hành xét trên từng hình thang:

**Công thức tính phân hình thang:**

**Sai số địa phương (Sai số từng phần):**

**Công thức hình thang tổng quát:**

**Sai số toàn phần:**

Với

Dựa trên công thức sai số toàn phần, để tăng độ chính xác, ta tiến hành chia bước nhảy càng nhỏ, tưng ứng với tăng số đoạn lên.

1. Phần giải các bài tập theo lý thuyết:

**Chương 8: Giải gần đúng phương trình**

**Bài tập 3.2 (bài số 1.) trang 103**: Bằng phương pháp lăp đơn, phương pháp Newton, và phương pháp dây cung tìm nghiệm gần đúng của các phương trình sau sao cho đạt sai số tuyệt đối < 0.01

**Phương pháp lặp đơn:**

* Xác định khoảng phân ly nghiệm:

Ta có: liên tục trên .

Nên suy ra khoảng phân ly nghiệm của là

* Xác định hàm lặp:

Chọn

Kiểm tra điều kiện hội tụ:

Ta có:

Thõa mãn điều kiện hàm lặp hội tụ.

* Xấp xỉ giá trị ban đầu
* Tìm nghiệm dựa vào công thức hàm lặp:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 1 | 3 | 3.4482 |
| 2 | 3.4482 | 3.5928 |
| 3 | 3.5928 | 3.6371 |
| 4 | 3.6371 | 3.6505 |
| 5 | 3.6505 | 3.6545 |

Theo đề bài, ta có sai số tuyệt đối mà

Dừng quá trình lặp.

Vậy nghiệm gần đúng của phương trình là

**Phương pháp Newton – Raphson:**

* Xác định khoảng phân ly nghiệm:

Ta có: liên tục trên .

Nên suy ra khoảng phân ly nghiệm của là

* Kiểm tra điều kiện hội tụ:

Ta có:

Thõa mãn điều kiện hội tụ.

* Xấp xỉ giá trị ban đầu . Vì thõa mãn điều kiện
* Tìm nghiệm dựa trên công thức tiếp tuyến:

|  |  |  |
| --- | --- | --- |
|  |  |  |
| 0 | 4 | 3.6944 |
| 1 | 3.6944 | 3.6567 |
| 2 | 3.6567 | 3.6562 |

Theo đề bài, ta có sai số tuyệt đối mà

Dừng quá trình lặp.

Vậy nghiệm gần đúng của phương trình là

**Phương pháp dây cung:**

* Xác định khoảng phân ly nghiệm:

Ta có: liên tục trên .

Nên suy ra khoảng phân ly nghiệm của là

* Kiểm tra điều kiện hội tụ:

Ta có:

Thõa mãn điều kiện hội tụ.

* Xấp xỉ giá trị ban đầu . Vì thõa mãn điều kiện
* Tìm nghiệm theo công thức dây cung:

Ta có bảng sau:

|  |  |  |  |
| --- | --- | --- | --- |
| Số lần lặp |  |  |  |
| 1 |  |  |  |
| 2 |  |  |  |
| 3 |  |  |  |
| 4 |  |  |  |

Theo đề bài, ta có sai số tuyệt đối mà

Dừng quá trình lặp.

Vậy nghiệm gần đúng của phương trình là .

**Chương 9: Giải hệ phương trình**

**Bài tập 4.1 (bài số 4,5,6) trang 158:** Giải hê phương trình , bằng phương pháp Gauss-Jordan, trong đó:

Các được toán được làm tròn tới 4 chữ số ở phần thập phân.

4.

Ghép hai ma trận:



Tuyến tính hóa hàng 1: Chia hàng đầu tiên cho hệ số đầu tiên = 1.7, ta có:



Triệt tiêu hệ số ở hàng 2,3,4 bằng cách:

Lấy hàng 2 – (hàng 1)\*0.2

Lấy hàng 3 – (hàng 1)\*0.1

Lấy hàng 4 – (hàng 1)\*0.3



Tiếp theo, ta tuyến tính hóa hàng 2 bằng cách chia hàng 2 cho 1.8765



Triệt tiêu ở hàng 1,3,4 bằng cách:

Lấy hàng 1 – (Hàng 2)\*0.1176

Lấy hàng 3 – (Hàng 2)\*(-0.1118)

Lấy hàng 4 – (Hàng 2)\*(0.1647)



Tiếp theo, ta tuyến tính hóa hàng 3 bằng cách chia hàng 3 cho 1.2874



Triệt tiêu ở hàng 1, 2, 4 bằng cách:

Lấy hàng 1 – (Hàng 3)\*0.0658

Lấy hàng 2 – (Hàng 3)\*(-0.0596)

Lấy hàng 4 – (Hàng 3)\*(-0.4078)



Tiếp theo, ta tuyến tính hóa hàng 4 bằng cách chia hàng 4 cho 3.9034



Triệt tiêu ở hàng 1, 2, 3 bằng cách:

Lấy hàng 1 – (Hàng 4)\*0.1870

Lấy hàng 2 – (Hàng 4)\*0.0689

Lấy hàng 3 – (Hàng 4)\*(-0.3168)



Kết luận nghiệm của là 

5.

Ghép hai ma trận:



Tuyến tính hóa hàng 1: Chia hàng đầu tiên cho hệ số đầu tiên = 6.1, ta có:



Triệt tiêu hệ số ở hàng 2,3,4 bằng cách:

Lấy hàng 2 – (hàng 1)\*1.7

Lấy hàng 3 – (hàng 1)\*1.0

Lấy hàng 4 – (hàng 1)\*9.1



Tiếp theo, ta tuyến tính hóa hàng 2 bằng cách chia hàng 2 cho 1.1262



Triệt tiêu ở hàng 1,3,4 bằng cách:

Lấy hàng 1 – (Hàng 2)\*0.2787

Lấy hàng 3 – (Hàng 2)\*0.8213

Lấy hàng 4 – (Hàng 2)\*5.5638



Tiếp theo, ta tuyến tính hóa hàng 3 bằng cách chia hàng 3 cho 0.2370



Triệt tiêu ở hàng 1, 2, 4 bằng cách:

Lấy hàng 1 – (Hàng 3)\*(-0.0394)

Lấy hàng 2 – (Hàng 3)\*0.7294

Lấy hàng 4 – (Hàng 3)\*(-4.7497)



Tiếp theo, ta tuyến tính hóa hàng 4 bằng cách chia hàng 4 cho -22.9283



Triệt tiêu ở hàng 1, 2, 3 bằng cách:

Lấy hàng 1 – (Hàng 4)\*0.2064

Lấy hàng 2 – (Hàng 4)\*3.2459

Lấy hàng 3 – (Hàng 4)\*2.3232



Kết luận nghiệm của là 

6.

Ghép hai ma trận:



Tuyến tính hóa hàng 1: Chia hàng đầu tiên cho hệ số đầu tiên = 3.2, ta có:



Triệt tiêu hệ số ở hàng 2,3,4 bằng cách:

Lấy hàng 2 – (hàng 1)\*1.7

Lấy hàng 3 – (hàng 1)\*1.0

Lấy hàng 4 – (hàng 1)\*9.1



Tiếp theo, ta tuyến tính hóa hàng 2 bằng cách chia hàng 2 cho 



Triệt tiêu ở hàng 1,3,4 bằng cách:

Lấy hàng 1 – (Hàng 2)\*2.40625

Lấy hàng 3 – (Hàng 2)\*(-1.30625)

Lấy hàng 4 – (Hàng 2)\*(-13.796875)



Tiếp theo, ta tuyến tính hóa hàng 3 bằng cách chia hàng 3 cho 



Triệt tiêu ở hàng 1, 2, 4 bằng cách:

Lấy hàng 1 – (Hàng 3)\* 

Lấy hàng 2 – (Hàng 3)\* ()

Lấy hàng 4 – (Hàng 3)\*( )



Tiếp theo, ta tuyến tính hóa hàng 4 bằng cách chia hàng 4 cho 



Triệt tiêu ở hàng 1, 2, 3 bằng cách:

Lấy hàng 1 – (Hàng 4)\*2.38751

Lấy hàng 2 – (Hàng 4)\*( )

Lấy hàng 3 – (Hàng 4)\*(3.83241)



Kết luận nghiệm của là 

**Chương 10: Nội suy, hồi quy**

**Bài tập 5.7 trang 190:** Cho giá trị của hàm số

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  | 4364.11 | 4425.17 | 4486.69 | 4548.69 | 4611.16 |

Dùng nội suy Newton tiến, tính gần đúng giá trị của y tại

Đầu tiên, ta có bảng sai phân:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |
|  |  | … | … | … | … | … |
| … | … |  | … | … | … | … |
|  |  | … | 0.47 | … | … | … |
| … | … |  | … | -0.01 | … | … |
|  |  | … | 0.46 | … | 0.03 | … |
| … | … |  | … | 0.02 | … | -0.06 |
|  |  | … | 0.48 | … | -0.03 | … |
| … | … |  | … | -0.01 | … | … |
|  |  | … | 0.47 | … | … | … |
| … | … |  | … | … | … | … |
|  |  | … | … | … | … | … |

Theo công thức nội suy Newton tiến:

Với

Với

**Chương 11 + 12: Đạo hàm, tích phân, Phương trình vi phân**

**Bài tập 6.3 trang 221:** Tính gần đúng tích phân xác định

Bằng công thức hình thang tổng quát với . Đánh giá sai số.

Ta có ta tiến hành phân hoạch đoạn thành đoạn bằng nhau với , ta được bảng sau:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |  |  |

Áp dụng công thức hình thang tổng quát:

Từ đó,

Sai số toàn phần

1. Phần thiết kế giao diện:
   1. Giới thiệu về App Designer trong Matlab:

App Designer cho phép tạo ra các ứng dụng chuyên nghiệp mà không cần phải là nhà phát triển phần mềm chuyên nghiệp. Kéo và thả các thành phần trực quan để bố trí thiết kế giao diện người dùng đồ họa (GUI) của bạn và sử dụng trình chỉnh sửa tích hợp để nhanh chóng lập trình hành vi cho nó.

**Các đối tượng giao diện**

Graphical user interface, application, Teams

Description automatically generatedGiao diện chính:

**Khu vực thiết kế giao diện**

**Tùy chỉnh thuộc tính của đối tượng**

Text

Description automatically generated

**Khu vực viết code để lập trình hành vi cho các đối tượng**

**Một số cú pháp nhập xuất dữ liệu, vẽ đồ thị:**

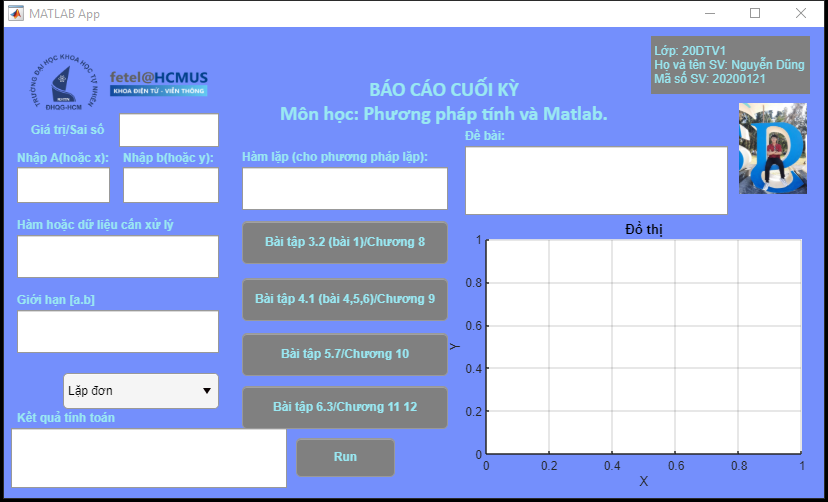
Variable = app.ComponentName.Properties %Cú pháp lấy dữ liệu.

app.ComponentName.Properties = %Variable Cú pháp xuất dữ liệu ra

plot(app.UIAxes,x,y) %Vẽ đồ thị.

* 1. Thiết kế giao diện:

**Yêu cầu:** Thiết kế giao diện để tính toán và hiện thị kết quả ra app các bài tập đã làm ở phần II.

* + 1. Giao diện chính của giao diện:

**Hiện thị kết quả**

**Vẽ đồ thị**

**Đề bài**

**Nhập dữ liệu**

**Thông tin người làm**

**Logo Trường, Khoa**

Giao diện bao gồm một số thành phần chính sau:

* Bên trái giao diện là các ô Edit Field Text để nhập dữ liệu đầu vào, và chọn phương pháp để tính toán tương ứng.
* Ở giữa là các nút ẩn để import dữ liệu các bài tập đã làm ở trên mục II.
* Bên trái là ô Text Area để hiện thị đề bài và một vùng để vẽ đồ thị.
* Bên dưới cùng là nút Run để chạy app, tính toán và hiển thị kết quả.
* Ngoài ra trên giao diện còn có Logo Trường và Khoa, Thông tin và ảnh người làm.

Một số Component Browser (tham chiếu tới những đối tượng trong giao diện để viết code hành vi cho nó).

Table

Description automatically generated

Graphical user interface, application

Description automatically generated

* + 1. Phần code thuật toán các phương pháp đã dùng để giải các bài tập ở phần II:
       1. Giải phương trình:

**Phương pháp lặp đơn: lap.m**

function [nghiem solanlap]=lap(fp,a,b,saiso)

syms x

fd1 = str2func(['@(x)' char(diff((fp(x))))]);

if (abs(fd1(a))>=1)||(abs(fd1(b))>=1)

disp('Không hội tụ')

return

end

x0 = a;

x1 = fp(x0);

solanlap=1;

while abs(x1-x0)>= saiso

x0 = x1;

x1 = fp(x0);

solanlap=solanlap+1;

end

nghiem = x1;

end

**Phương pháp tiếp tuyến (Newton – Raphson): tieptuyen.m**

function [nghiem solanlap]=tieptuyen(f,a,b,saiso)

syms x;

fd1 = str2func(['@(x)' char(diff(f(x)))]);

fd2 = str2func(['@(x)' char(diff(f(x),2))]);

nostop = 1;

temp1 = double(solve(diff(f(x))));

temp2 = double(solve(diff(fd1(x))));

%Kiem tra f'(x) co doi dau trong khoang phan li nghiem hay khong

if ~isempty(temp1)

for i=1:length(temp1)

if (temp1(i)<=b) && (temp1(i)>=a)

disp('Ham f'' doi dau trong khoang phan li nghiem');

nostop=0;

nghiem = 'khong xac dinh';

solanlap = 'khong xac dinh';

break;

end

end

end

%Kiem tra f"(x) co doi dau trong khoang phan li nghiem hay ko

if ~isempty(temp2)

for i=1:length(temp2)

if (temp2(i)<=b) && (temp2(i)>=a)

disp('Ham f" doi dau trong khoang phan li nghiem');

nostop=0;

nghiem = 'khong xac dinh';

solanlap = 'khong xac dinh';

break;

end

end

end

if (nostop) %thuat toan phuong phap tiep tuyen.

x0 = a;

while f(x0)\*fd2(x0)<=0

if f(x0)==0

nghiem = x0;

solanlap = 0;

return

else

x0 = b;

end

end

nghiem=x0-(f(x0)/fd1(x0));

solanlap=0;

while abs(nghiem-x0)>=saiso

x0=nghiem;

nghiem = x0-(f(x0)/fd1(x0));

solanlap=solanlap+1;

end

nghiem = double(nghiem);

end

end

**Phương pháp dây cung: secant.m**

function [X,n]=secant(fun,xO,x1,err)

if nargin< 4

err=le-5;

end

n=0;

while abs(x1-xO)>=err

x2 = xO;

xO = x1;

x1 = x1+(x1-x2)/(feval (fun,x2)/feval(fun,x1)-1);

n=n+1;

end

X=x1;

end

* + - 1. Giải hệ phương trình:

**Phương pháp Gauss – Jordan: gauss\_jordan.m**

function nghiem = gauss\_jordan (A,B)

AB = [A B'];

for k = 1:length(B)

AB(k,:) = AB(k,:)./AB(k,k);

for m = 1:length(B)

if m~=k

AB(m,:) = AB(m,:) - AB(k,:).\*AB(m,k);

end

end

end

c = size(AB);

nghiem = round(AB(:,c(2)),4);

end

* + - 1. Nội suy, Hồi quy:

**Nội suy Newton: noisuy\_newton.m**

function [dathuc Yn] = noisuy\_newton(x\_array,y\_array,No)

syms t d

mid = mean(x\_array);

h = abs(x\_array(2) - x\_array(1));

d = 1;

n = 0;

if No < mid %Newton tiến

P(t) = y\_array(1) + 0\*t;

for m = 1:length(x\_array)-1

for k = 1:length(y\_array)-1

y\_array(k) = y\_array(k+1) - y\_array(k);

end

d = (t - m + 1)\*d;

P = P + (d/factorial(m))\*y\_array(1);

y\_array(length(y\_array) - n) = 0;

n = n + 1;

end

a = (No - x\_array(1))/h;

dathuc = simplify(P);

dathuc = str2func(['@(t)' char(dathuc)]);

Yn = dathuc(a);

else %Newton lùi

P(t) = y\_array(length(y\_array)) + 0\*t;

for m = 1:length(x\_array)-1

for k = 1:length(y\_array)-1

y\_array(k) = y\_array(k+1) - y\_array(k);

end

d = (t + m - 1)\*d;

y\_array(length(y\_array) - m);

P = P + (d/factorial(m))\*y\_array(length(y\_array) - m);

y\_array(length(y\_array) - n) = 0;

n = n + 1;

end

a = (No - x\_array(length(x\_array)))/h;

dathuc = simplify(P);

dathuc = str2func(['@(t)' char(dathuc)]);

Yn = dathuc(a);

end

end

* + - 1. Đạo hàm, tích phân, Phương trình vi phân:

**Tính tích phân theo phương pháp hình thang: tichphanhinhthang.m**

function y = tichphanhinhthang(fx,a,b,N)

h = (b - a)/N;

k = 1:N-1;

y = (h/2).\*(fx(a)+fx(b)+2.\*sum(fx(a+k.\*h)));

end

* + 1. Phần code giao diện:

% Callbacks that handle component events

methods (Access = private)

% Button pushed function: RunButton

function RunButtonPushed(app, event)

try

app.KetQuaTinhToan.Value = '';

switch app.PhuongPhap.Value

case 'Lặp đơn'

fx = app.Data.Value;

fx = str2func(['@(x)' fx]);

fLap = app.Hamlap.Value;

fLap = str2func(['@(x)' fLap]);

Seg = app.GioiHan.Value;

if (Seg == "")

msgbox({'Không để trống!',' Vui lòng nhập khoảng giới hạn'},'Error','error');

else

Seg =str2num(Seg);

saiso = app.GiaTri.Value;

if (saiso == "")

msgbox({'Không để trống!',' Vui lòng nhập giá trị sai số'},'Error','error');

else

saiso = str2num(saiso);

[nghiem solanlap] = lap(fLap,Seg(1),Seg(2),saiso);

x = 0:0.01:10;

y = fx(x);

plot(app.UIAxes,x,y);

text(app.UIAxes,nghiem,0,'x');

app.KetQuaTinhToan.Value = ['Nghiệm gần đúng của phương trình là:' num2str(nghiem) ' và Số lần lặp là: ' num2str(solanlap)];

end

end

case 'Tiếp tuyến'

fx = app.Data.Value;

fx = str2func(['@(x)' fx]);

Seg = app.GioiHan.Value;

if (Seg == "")

msgbox({'Không để trống!',' Vui lòng nhập khoảng giới hạn'},'Error','error');

else

Seg =str2num(Seg);

saiso = app.GiaTri.Value;

if (saiso == "")

msgbox({'Không để trống!',' Vui lòng nhập giá trị sai số'},'Error','error');

else

saiso = str2num(saiso);

[nghiem solanlap] = tieptuyen(fx,Seg(1),Seg(2),saiso);

x = 0:0.01:10;

y = fx(x);

plot(app.UIAxes,x,y);

text(app.UIAxes,nghiem,0,'x');

app.KetQuaTinhToan.Value = ['Nghiệm gần đúng của phương trình là:' num2str(nghiem) ' và Số lần lặp là: ' num2str(solanlap)];

end

end

case 'Dây cung'

fx = app.Data.Value;

fx = str2func(['@(x)' fx]);

Seg = app.GioiHan.Value;

if (Seg == "")

msgbox({'Không để trống!',' Vui lòng nhập khoảng giới hạn'},'Error','error');

else

Seg =str2num(Seg);

saiso = app.GiaTri.Value;

if (saiso == "")

msgbox({'Không để trống!',' Vui lòng nhập giá trị sai số'},'Error','error');

else

saiso = str2num(saiso);

[nghiem solanlap] = secant(fx,Seg(1),Seg(2),saiso);

x = 0:0.01:10;

y = fx(x);

plot(app.UIAxes,x,y);

text(app.UIAxes,nghiem,0,'x');

app.KetQuaTinhToan.Value = ['Nghiệm gần đúng của phương trình là:' num2str(nghiem) ' và Số lần lặp là: ' num2str(solanlap)];

end

end

case 'Gauss-Jordan'

A = app.Nhap\_A.Value;

b = app.Nhap\_b.Value;

if (A == "") || (b == "")

msgbox({'Không để trống!','Vùi lòng nhập dữ liệu để giải hệ phương trình.'},'Error','error');

else

A = str2num(A);

b = str2num(b);

nghiem = (gauss\_jordan(A,b))';

nghiem = num2str(nghiem);

nghiem = "{ " + nghiem + " }";

app.KetQuaTinhToan.Value = ['Nghiệm của hệ phương trình là: ' num2str(nghiem)];

end

case 'Newton'

x = app.Nhap\_A.Value;

y = app.Nhap\_b.Value;

if (x == "") || (y == "")

msgbox({'Không để trống!','Vùi lòng nhập dữ liệu để tìm đa thức nội suy.'},'Error','error');

else

x = str2num(x);

y = str2num(y);

No = app.Data.Value;

if (No == "")

msgbox("Vùi lòng nhập giá trị nội suy",'Error','error');

else

No = str2num(No);

[dathuc Yn] = noisuy\_newton(x,y,No);

j = 1;

a = 0:0.01:(x(length(x)) - x(1))/(x(2) - x(1));

for i = 0:0.01:(x(length(x)) - x(1))/(x(2) - x(1))

b(j) = dathuc(i);

j = j + 1;

end

plot(app.UIAxes,a,b);

dathuc = func2str(dathuc);

Yn = num2str(Yn);

if contains(dathuc,'@(t)')

dathuc = strrep(dathuc,'@(t)','');

end

app.KetQuaTinhToan.Value = ['Đa thức nội suy Newton là: ' dathuc ' và Giá trị nội suy là ' Yn];

end

end

case 'Hình thang'

fx = app.Data.Value;

fx = str2func(['@(x)' fx]);

Seg = app.GioiHan.Value;

if (Seg == "")

msgbox({'Không để trống!',' Vui lòng nhập khoảng giới hạn'},'Error','error');

else

Seg = str2num(Seg);

N = app.GiaTri.Value;

if (N == "")

msgbox({'Không để trống!',' Vui lòng nhập giá trị số đoạn con.'},'Error','error');

else

N = str2num(N);

if (N < 1)

msgbox({'Dùng sai phương pháp','Vui lòng chọn phương pháp phù hợp để giải!'},'Error','error');

else

I = tichphanhinhthang(fx,Seg(1),Seg(2),N);

I = num2str(I);

x = 0:0.01:10;

y = fx(x);

plot(app.UIAxes,x,y);

line(app.UIAxes,[Seg(1) Seg(1)], [0 fx(Seg(1))],'Color','black','linewidth',2);

line(app.UIAxes,[Seg(2) Seg(2)], [0 fx(Seg(2))],'Color','black','linewidth',2);

app.KetQuaTinhToan.Value = ['Kết quả tích phân là: ' I];

end

end

end

otherwise

end

catch error

switch (error.identifier)

case {'MATLAB:badsubscript','MATLAB:m\_illegal\_character','MATLAB:dimagree','MATLAB:catenate:dimensionMismatch'}

msgbox({'Dùng sai phương pháp','Vui lòng chọn phương pháp phù hợp để giải!'},'Error','error');

case 'MATLAB:m\_incomplete\_statement'

msgbox({'Không để trống','Vùi lòng nhập hàm/hàm lặp hoặc dữ liệu cần xử lý'},'Error','error');

otherwise

msgbox({error.identifier,error.message},'Error','error');

end

end

end

% Button pushed function: BaiTap3\_2

function BaiTap3\_2ButtonPushed(app, event)

app.Data.Value = 'x.^3 - 12\*x - 5';

app.Hamlap.Value = '(5+12.\*x).^(1/3)';

app.GioiHan.Value = '[3 4]';

app.Nhap\_A.Value = 'Không nhập';

app.Nhap\_b.Value = 'Không nhập';

app.GiaTri.Value = '0.01';

app.KetQuaTinhToan.Value = '';

app.DeBai.Value = 'Bằng phương pháp lăp đơn, phương pháp Newton, và phương pháp dây cung tìm nghiệm gần đúng của phương trình x.^3 -12.\*x - 5 = 0 sao cho đạt sai số tuyệt đối < 0.01';

plot(app.UIAxes,0,0);

end

% Button pushed function: BaiTap4\_1

function BaiTap4\_1ButtonPushed(app, event)

app.Nhap\_A.Value = '[1.7 0.2 0.1 0.3;0.2 1.9 -0.1 0.2;0.1 -0.1 1.3 -0.4;0.3 0.2 -0.4 4.1]';

app.Nhap\_b.Value = '[1.6 0.3 -1.2 0.7]';

app.Data.Value = 'Không nhập';

app.Hamlap.Value = 'Không nhập';

app.GioiHan.Value = 'Không nhập';

app.GiaTri.Value = 'Không nhập';

app.KetQuaTinhToan.Value = '';

app.DeBai.Value = 'Giải hê phương trình Ax = b bằng phương pháp Gauss-Jordan vơi A, b đã được nhập sẳn.'

plot(app.UIAxes,0,0);

end

% Button pushed function: BaiTap5\_7

function BaiTap5\_7ButtonPushed(app, event)

app.Nhap\_A.Value = '[58 58.34 58.68 59.02 59.36 59.7]';

app.Nhap\_b.Value = '[4303.52 4364.11 4425.17 4486.69 4548.69 4611.16]';

app.Data.Value = '58.17';

app.Hamlap.Value = 'Không nhập';

app.GioiHan.Value = 'Không nhập';

app.GiaTri.Value = 'Không nhập';

app.KetQuaTinhToan.Value = '';

app.DeBai.Value = 'Xây dưng công thức nội suv Newton với dãy x, y đã được nhập sẳn, tại x = 58.17';

plot(app.UIAxes,0,0);

end

% Button pushed function: BaiTap6\_3

function BaiTap6\_3ButtonPushed(app, event)

app.Data.Value = 'sqrt(x)';

app.GioiHan.Value = '[1,2]';

app.GiaTri.Value = '10';

app.Hamlap.Value = 'Không nhập';

app.Nhap\_A.Value = 'Không nhập';

app.Nhap\_b.Value = 'Không nhập';

app.KetQuaTinhToan.Value = '';

app.DeBai.Value = 'Tính gần đúng tích phân xác định: fx = sqrt(x), a = 1, b = 2, N =10';

plot(app.UIAxes,0,0);

end

end

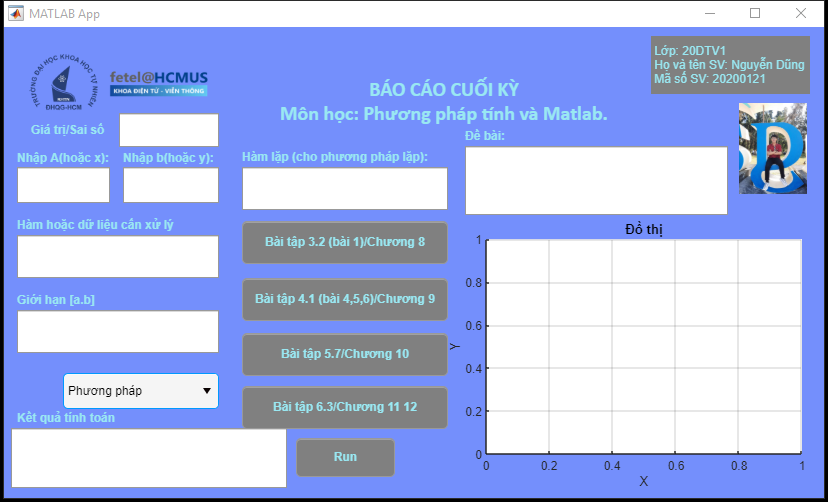
* 1. Kết quả:

Tiền hành chạy app: Ấn nút Run (trong tab DESIGNER).

Graphical user interface, application, Word

Description automatically generated

Ấn Run

sau đó, sẽ xuất hiện giao diện chính như sau:

Tiến hành thực hiện các bước như bên dưới để tính toán và hiển thị kết quả trên App

* + 1. Giải phương trình:
* Bước 1: Nhấn vào nút **“Bài tập 3.2 (bài 1)/Chương 8”** để load dữ liệu bài tập lên. Các dữ liệu cần cho tính toán như:
* Giá trị/Sai số:
* Hàm lặp (cho phương pháp lặp).
* Hàm hoặc dữ liệu cần xử lý (Hàm cần giải nghiệm).
* Khoảng giới hạn

Ngoài ra còn hiện đề bài ở khung “Đề bài” để tiện quan sát.

Graphical user interface, application

Description automatically generated

Ấn vào

Graphical user interface, application

Description automatically generated Ta thu được hình như sau:

* Bước 2: **Chọn phương pháp phù hợp**: trong phần này chỉ xét 3 phương pháp:
* Phương pháp lặp đơn.
* Phương pháp tiếp tuyến (Newton – Raphson).
* Graphical user interface, application

  Description automatically generatedPhương pháp dây cung.
* Bước 3: **Ấn nút Run** để tính toán, vẽ đồ thị, và hiện thị kết quả trên giao diện.

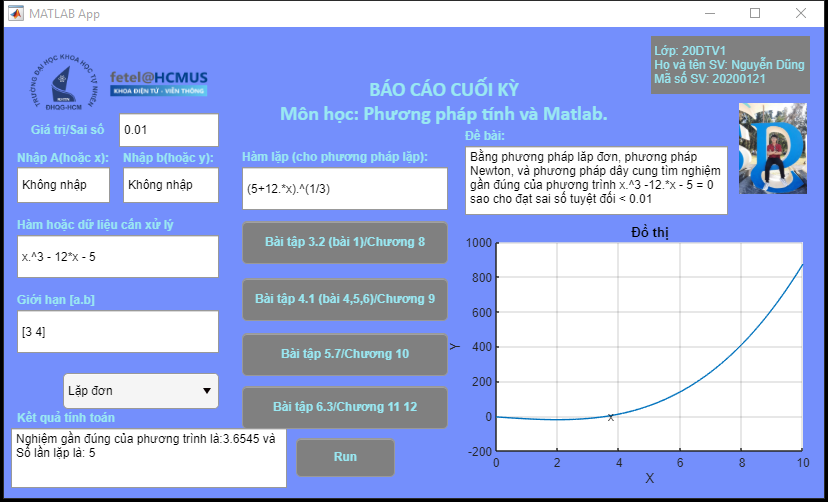
Graphical user interface, application

Description automatically generated

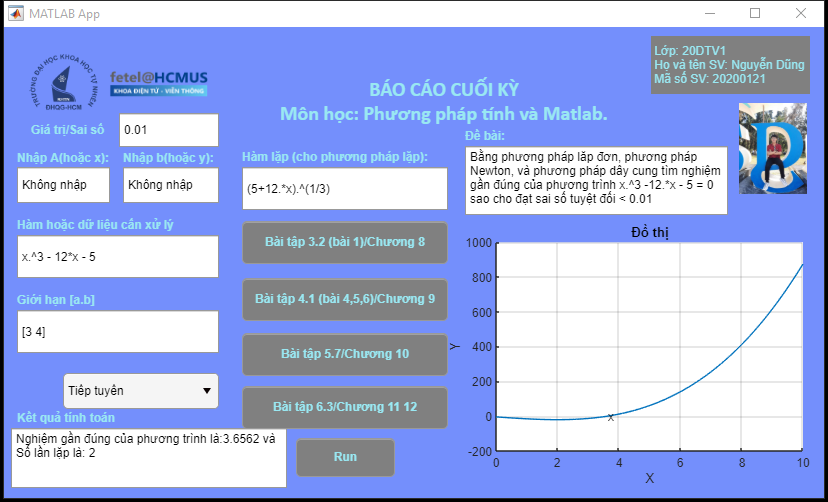
Ấn Run

Chú ý: Khi load dữ liệu có những Field hiển thị “không nhập”, tức có nghĩa trường đó không ảnh hưởng đến kết quả tính toán.

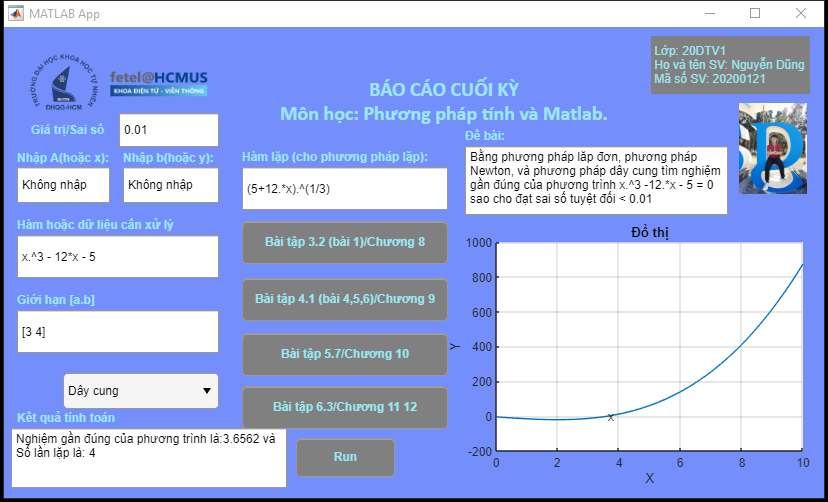
Kết quả các hình tương ứng với từng phương pháp.

* + - 1. Phương pháp lặp đơn:

So sánh với kết quả tính toán bằng lý thuyết ở mục II, ta thấy kết quả tương đối giống nhau, mặc dù cũng có sai số nhưng không đáng kể.

* + - 1. Phương pháp tiếp tuyến (Newton – Raphson):

So sánh với kết quả tính toán bằng lý thuyết ở mục II, ta thấy kết quả tương đối giống nhau, mặc dù cũng có sai số nhưng không đáng kể.

* + - 1. Phương pháp dây cung:

So sánh với kết quả tính toán bằng lý thuyết ở mục II, ta thấy kết quả tương đối giống nhau, mặc dù cũng có sai số nhưng không đáng kể.

* + 1. Giải hệ phương trình:
* Bước 1: Nhấn vào nút **“Bài tập 4.1 (bài 4,5,6)/Chương 9”** để load dữ liệu bài tập lên. Các dữ liệu cần cho tính toán như:
* Ma trận A.
* Ma trận b.

Ngoài ra còn hiện đề bài ở khung “Đề bài” để tiện quan sát.

Graphical user interface, application

Description automatically generated

Ấn vào

Ấn vào

Graphical user interface, application

Description automatically generatedTa thu được hình như sau:

* Graphical user interface, application

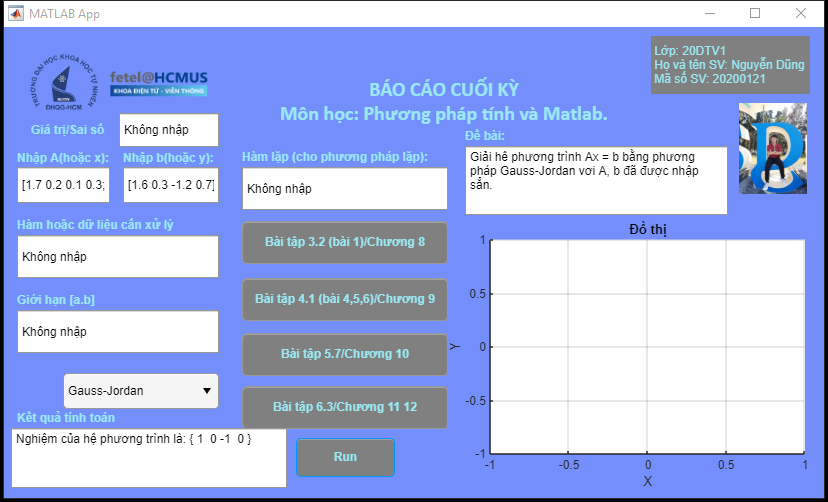
  Description automatically generatedBước 2: **Chọn phương pháp phù hợp**: trong phần này chỉ xét một phương pháp: Phương pháp Gauss-Jordan.
* Graphical user interface, application

  Description automatically generatedBước 3: **Ấn nút Run** để tính toán và hiện thị kết quả trên giao diện.

Ấn Run

Chú ý: Khi load dữ liệu có những Field hiển thị “không nhập”, tức có nghĩa trường đó không ảnh hưởng đến kết quả tính toán.

Kết quả hình tương ứng với phương pháp.

* + - 1. Phương pháp Gauss-Jordan:

So sánh với kết quả tính toán bằng lý thuyết ở mục II, ta thấy kết quả tương đối giống nhau, mặc dù cũng có sai số nhưng không đáng kể.

Graphical user interface, application

Description automatically generatedTrong phần giải hệ phương trình, chúng ta còn hai bài nữa, ta cần nhập tương ứng ma trận , ma trận sau đó nhấn ấn Run để hiển thị ra kết quả như sau:

Graphical user interface, application

Description automatically generated

So sánh với kết quả lý thuyết, ta thấy nó tương đối giống nhau, mặc dù có sai số nhưng không đáng kể.

* + 1. Nội suy, Hồi quy:
* Bước 1: Nhấn vào nút **“Bài tập 5.7 /Chương 10”** để load dữ liệu bài tập lên. Các dữ liệu cần cho tính toán như:
* Hàm hoặc dữ liệu cần xử lý (Điểm cần nội suy).
* Ma trận , Ma trận (để tìm đa thức nội suy).

Graphical user interface, application

Description automatically generatedNgoài ra còn hiện đề bài ở khung “Đề bài” để tiện quan sát.

Ấn vào

Ta thu được hình như sau:

Graphical user interface, application

Description automatically generated

* Graphical user interface, application

  Description automatically generatedBước 2: **Chọn phương pháp phù hợp**: trong phần này chỉ xét 1 phương pháp: Phương pháp nội suy Newton
* Graphical user interface, application

  Description automatically generatedBước 3: **Ấn nút Run** để tính toán, vẽ đồ thị và hiện thị kết quả trên giao diện.

Ấn Run

Chú ý: Khi load dữ liệu có những Field hiển thị “không nhập”, tức có nghĩa trường đó không ảnh hưởng đến kết quả tính toán.

Kết quả hình tương ứng với phương pháp.

* + - 1. Graphical user interface

         Description automatically generatedNội suy Newton:

Do kích thước của khung “Kết quả tính toán” không đủ để hiển thị ra đa thức nội suy Newton, nên em sẽ ghi lại như sau (chú ý đây là đa thức theo biến , theo công thức

So sánh với kết quả tính toán bằng lý thuyết ở mục II, ta thấy kết quả tương đối giống nhau, mặc dù cũng có sai số nhưng không đáng kể.

* + 1. Đạo hàm, tích phân, Phương trình vi phân:
* Bước 1: Nhấn vào nút **“Bài tập 6.3 /Chương 11 12”** để load dữ liệu bài tập lên. Các dữ liệu cần cho tính toán như:
* Giá trị, sai số (Số N đoạn con).
* Hàm hoặc dữ liệu cần xử lý (Hàm cần tính tích phân).
* Hai cận , để tính tích phân (Khoảng giới hạn )

Ngoài ra còn hiện đề bài ở khung “Đề bài” để tiện quan sát.

Graphical user interface, application

Description automatically generated

Ấn vào

Graphical user interface, application

Description automatically generatedTa thu được hình như sau:

* Graphical user interface, application

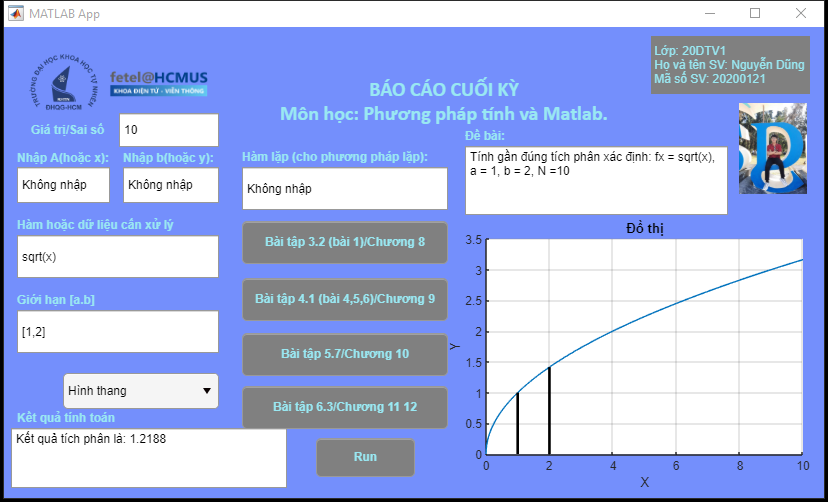
  Description automatically generatedBước 2: **Chọn phương pháp phù hợp**: trong phần này chỉ xét 1 phương pháp: Phương pháp hình thang.
* Graphical user interface, application

  Description automatically generatedBước 3: **Ấn nút Run** để tính toán, vẽ đồ thị và hiện thị kết quả trên giao diện.

Ấn Run

Chú ý: Khi load dữ liệu có những Field hiển thị “không nhập”, tức có nghĩa trường đó không ảnh hưởng đến kết quả tính toán.

Kết quả hình tương ứng với phương pháp.

* + - 1. Phương pháp hình thang:

So sánh với kết quả tính toán bằng lý thuyết ở mục II, ta thấy kết quả tương đối giống nhau, mặc dù cũng có sai số nhưng không đáng kể.

* 1. Thiết lập một số lỗi trong giao diện:

Trong phần code giao diện, chúng ta sử dụng khối lệnh

try  
*statements*  
catch *exception*  
*statements*  
end

thực thi các câu lệnh trong khối try và bắt những lỗi trong khối catch. Cách tiếp cận này cho phép chúng ta ghi đè lỗi mặc định cho một tập hợp các câu lệnh chương trình (xử lý lỗi). Nếu bất kỳ câu lệnh nào trong khối try tạo ra lỗi, chương trình sẽ ngay lập tức chuyển sang khối catch, khối này chứa các câu lệnh xử lý lỗi. Vậy nên chúng ta sữ dụng khối lệnh này để ra các hộp thoại báo lỗi khi xuất hiện lỗi ở chương trình.

Khảo sát một số lỗi đã thiết lập:

Graphical user interface, application

Description automatically generated **Lỗi dùng sai phương pháp:**

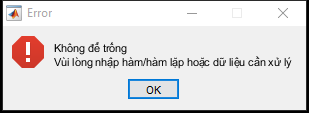
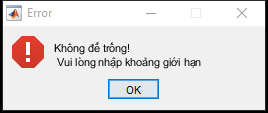
Ở đây, ta đang muốn tính tích phân nhưng lại chọn phương pháp tìm nghiệm (phương pháp lặp đơn) Lỗi, xuất hiện hộp thoại như trên hình.

Đối với các phần khác cũng vậy, khi chọn sai phương pháp thì sẽ xuất hiện hộp thoại thông báo lỗi.

**Nhập thiếu dữ liệu đầu vào:**

Graphical user interface, application

Description automatically generatedGiả sử trong quá trình tìm nghiệm, đã chọn đúng phương pháp, ta để trống giá trị sai số thì xuất hiện hộp thoại báo lỗi như hình.

Tương tự với các trường hợp khác như để trống “Hàm hoặc dữ liệu cần xử lý”, “Giới hạn [a, b]” và “Hàm lặp” lần lượt sẽ xuất hiện các hộp thoại như sau:

Graphical user interface, application

Description automatically generatedTrong phần giải hệ phương trình, khi chọn đúng phương pháp, nếu nhập thiếu dữ liệu đầu vào cũng sẽ thông báo lỗi.

Tương tự như các phần khác khi nhập thiếu dữ liệu đầu vào cũng sẽ thông báo lỗi.

* 1. Đánh giá sản phẩm:
     1. Ưu điểm:
* Giao diện cho ra kết quả như mong đợi (giống với kết quả lý thuyết đã làm ở mục 2).
* Áp dụng được các thuật toán cơ bản trong Phương pháp tính để giải nghiệm gần đúng, giải hệ phương trình, nội suy Newton và tính tích phân.
* Giao diện đầy đủ các thành phần cơ bản như yêu cầu…
  + 1. Nhược điểm:
* Trong quá trình tính toán, xử lý và đưa ra kết quả còn chậm (do trong code thuật toán có nhiều vòng lặp).
* Hiển thị kết quả chưa được đẹp.
  1. Kết luận:

Qua quá trình nghiên cứu, em đã hoàn thành được sản phẩm theo yêu cầu đề ra, mặc dù cũng còn nhiều thiếu sót trong báo cáo cũng như trong phần thiết kế giao diện, em mong quý thầy cô có thể bỏ qua. Lời cuối cùng, em xin cảm ơn thầy Nguyễn Xuân Vinh đã truyền tải cho em rất nhiều kiến thức để em có thể hoàn thành được đồ án này. Một lần nữa, em xin chân thành cảm ơn thầy.

**HẾT.**