BÀI THỰC HÀNH MATLAB

Họ tên: Nguyễn Anh Tuấn
MSSV: 20200400
Ca học: 9

Câu 1:

File	Code	Giải thích

main.m	x_arr = [0.1 0.3 0.5 0.7 0.9];
	y_arr = [0.1002 0.3047 0.5236 0.7754 1.1198];
	f= @(x) asin(x);
	h = abs(x_arr(1,length(x_arr))-x_arr(1,length(x_arr)-1));
	
	val = 0.7;
	
	fprintf('Ket qua cua da thuc Taylor: %f\n',
	taylor(x_arr,y_arr,val,h));
	fprintf('Ket qua voi da thuc Lagrange: %f\n', lagrange(x_arr, y_arr, val));
		Tạo mảng x như đề
		Tạo mảng y như đề
		Tạo hàm f dưới dạng function_handle
		h là bước nhảy giữa các phần tử trong mảng x => ở đây lấy khoảng cách của phần tử đầu tiên và phần tử thứ 2 trong mảng
		
		In ra màn hình kết quả đa thức taylor với tham chiếu mảng x , y , value = 0.7 và bước nhảy h
		
		In ra màn hình kết quả đa thức lagrange với tham chiếu mảng x , y , value = 0.7 và bước nhảy h

taylor.m
(câu a)
	function answer = taylor(x,y,val,h)
	    n = length(x);
	    for i=1:n
	        if val == x(1,i)
	            answer = (y(1,i+1) - y(1,i)) / h;
	            break;
	        end
	    end
	end
		Khai báo hàm taylor với các tham chiếu tương ứng
		N nhận giá trị là kích thước mảng x
		Cho chạy mảng từ 1->n với điều kiện so sánh value = 0.7 với phần tử của biến ma trận x hàng 1 cột I nếu = nhau thì lấy hiệu tại vị trí i+1 trong mảng y và vị trí i trong mảng y chia cho bước nhảy h là ta đc kết quả đa thức, huỷ vòng lặp thông qua break

lagrange.m
(câu b)
	function answer = lagrange(x_arr, y_arr, var)
	    n = length(x_arr);
	    sub_arr = ones(1,n);
	    l = sym(sub_arr);
	    syms x;
	    for i = 1:length(x_arr)
	        for j = 1:length(x_arr)
	            if i ~= j
	                l(i) = l(i) * (x - x_arr(j))/(x_arr(i) - x_arr(j));
	            end
	        end
	    end
	    syms x;
	    func = diff(vpa(simplify((l*y_arr'))),x);
	    answer = func(var);
		Tạo hàm lagrange với tham chiếu mảng x vs mảng y
		N nhận kích thước mảng x
		Tạo mảng phụ 1 hàng n phần tử = 1
		Cấu trúc mảng phụ thành symbolic 1 hàng n cột lưu từ biến t
		Khai báo x symbolic
		Chạy vòng lặp từ i -> n lồng vào lặp từ j->n nhằm duyệt các phần tử trong x vs y
		So sánh nếu i khác j thì:
		Phần tử tại I của t = t(i) * (x-x_arr(i))/ (x_arr(i) - x_arr(j)) (tương tự như phép tính hàm l(x) tại index)
		
		Lúc này ta được mảng t gồm các l(x) từ index 1 đến n
		Theo công thức, tiếp tục nhân từng index của y với t, ta được hàm symbol cần tìm sau đó thông qua nvpa vs simplify để đơn giản hoá hàm, sau đó dùng hàm diff để đạo hàm-> được hàm func là hàm lagrange đã đạo hàm cấp 1

	`end`	**Answer nhận giá trị tại f'(0.7) truyền vào**

Câu a: Kết quả đa thức Taylor: **1.722**

Câu b: Kết quả đa thức Lagrange: **1.392**

Câu c: Kết quả chính xác đạo hàm :

$$(\arcsin(x))' = \frac{1}{\sqrt{1-x^2}} = \frac{1}{\sqrt{1-0.7^2}} \approx \mathbf{1,4003}$$

Nhận xét:

-Kết quả tính đạo hàm theo Lagrange gần đúng với kết quả chính xác hơn là Taylor

-Từ kết quả trên cho thấy được sự khách quan trong tính chính xác của công thức Taylor và Lagrange khi dùng để tính chung đạo hàm 1 hàm tại x xác định.

Câu 2 + Câu 3:

File	Code	Giải thích

main.m	`f = @(x) x.^3.*sin(x)+x.*cos(x);`
	`a = 0; b = 1; n = 10;`
	`fprintf('Tich phan hinh thang cua ham f la: %f \n', tichphanhinhthang(f,a,b,n));`
	`fprintf('Tich phan Simpson cua ham f la: %f \n', tichphanSimpson(f,a,b,n));`
		**Tạo hàm function_handle f(x)
		Khai báo a, b, n tương ứng
		In ra màn hình kết quả tích phân hình thang hàm f
		In ra màn hình kết quả tích phân Simpson hàm f**

tichphanhinhthang.m	`function answer = tichphanhinhthang(f,a,b,n)`
	`    h = (b-a)/n;`
	`    answer = 0;`
	`    for i=1:n-1`
	`        answer = answer + f(a+i*h);`
	`    end`
	`    answer = (answer*2+f(a)+f(b))*h/2;`
	`end`
		**Khai báo hàm tichphanhinhthang
		h nhận giá trị là (b-a)/n
		answer tạm thời nhận giá trị 0
		chạy vòng lặp từ 1->n-1 sau đó answer nhận giá trị liên tục cộng với f(a+i*h)
		Sau đó x2 lần rồi cộng với f(a) vs f(b) rồi nhân với h/2**

tichphanSimpson.m	`function answer = tichphanSimpson(f,a,b,n)`
	`    h = (b-a)/n;`
	`    sum1 = 0;`
	`    for i = 2:2:n-1`
	`        sum1 = sum1 + f(a+i*h);`
	`    end`
	
	`    sum1 = sum1 * 2;`
	`    sum2 = 0;`
	`    for i = 1:2:n-1`
	`        sum2 = sum2 + f(a+i*h);`
	`    end`
	
	`    sum2 = sum2 * 4;`
	
	`answer=(sum1+sum2+f(a)+f(b))*h/3;`
	`end`
		**Khai báo hàm simpson
		h nhận giá trị là (b-a)/n
		gọi biến sum1 tạm thời = 0
		chạy vòng lặp từ 2 -> n-1 bước nhảy 2 nhằm nhận phần tử chẵn trong mảng sau đó sum1 nhận giá trị liên tục cộng với f(a+i*h)
		sum1 được x2 giá trị
		sum2 tạm thời = 0
		chạy vòng lặp từ 1 -> n-1 bước nhảy 2 nhằm nhận phần tử lẻ trong mảng sau đó sum2 nhận giá trị liên tục cộng với f(a+i*h)
		sum2 được x4 giá trị
		answer trả về kq = (sum1+sum2+f(a)+f(b)+*h/3**

Câu 4:

a. ***Tính chính xác tích phân của hàm số $f(x) = x^3 \sin x$ trong khoảng [0,1]. So sánh với giá trị gần đúng ở câu a và b rồi nhận xét. Gợi ý: tìm hiểu lệnh `integral`***

File	Code	Giải thích

main.m	`f = @(x) x.^3.*sin(x);`
	`a = 0; b = 1; n = 10;`
	`fprintf('Gia tri tich phan chinh xac cua ham f la: ~ 0.177099\n');`
	`fprintf('Tich phan hinh thang cua ham f la: %f \n', tichphanhinhthang(f,a,b,n));`
	`fprintf('Tich phan Simpson cua ham f la: %f \n', tichphanSimpson(f,a,b,n));`
	
	`fprintf('Tich phan Integral cua ham f la: %f \n', integral(f,a,b));`
		Tạo hàm function_handle f(x)
		Khai báo a, b, n tương ứng
		In ra màn hình kết quả tích phan chính xác tính tay
		In ra màn hình kết quả tích phân hình thang hàm f
		In ra màn hình kết quả tích phân Simpson hàm f
		
		In ra màn hình kết quả tích phân Integral hàm f

Nhận xét: Giá trị chính xác (tính tay) và giá trị theo hàm Integral là giống nhau, còn ở giá trị tích phân hàm simpson vs tích phân hình thang theo N = 10 thì có sự chênh lệch, cụ thể:

```
Gia tri tich phan chinh xac cua ham f la:  0.177099

Tich phan hinh thang cua ham f la:   0.179652

Tich phan Simpson cua ham f la:   0.177102

Tich phan Integral cua ham f la:   0.177099
```

Ở đây ta thấy độ chênh lệch sai số không quá $2.5 * 10^{-3}$

b. ***Viết chương trình tính sự chênh lệch của 2 phương pháp hình thang và Simpson so với tích phân chính xác `integral` ở ba trường hợp N = 1, N = 10 và N = 50 (in ra chi tiết giá trị chênh lệch). Từ đó rút ra kết luận phương pháp nào có độ chính xác cao hơn?***

Code	Giải thích

`f = @(x) x.^3.*sin(x);`
`a = 0; b = 1;`
 	
`correct = integral(f,a,b);`
	Tạo hàm f function_handle tương ứn
	khai báo a,b
	Biến correct nhận kq chính xác từ hàm integral tham chiếu f , a,b

`thang1 = abs(tichphanhinhthang(f,a,b,1)-correct);`
`thang10 = abs(tichphanhinhthang(f,a,b,10)-correct);`
`thang50 = abs(tichphanhinhthang(f,a,b,50)-correct);`
	Biến thang1 nhận giá trị chênh lệch của giá trị hàm tích phân hình thang với n = 1 so với giá trị chính xác integral; biến thang10 , thang50 tương ứng với n = 10, 50

`simpson1 = abs(tichphanSimpson(f,a,b,1)-correct);`
`simpson10 = abs(tichphanSimpson(f,a,b,10)-correct);`
`simpson50 = abs(tichphanSimpson(f,a,b,50)-correct);`
	Biến simpson1 nhận giá trị chênh lệch của giá trị hàm tích phân hình thang với n = 1 so với giá trị chính xác integral; biến simpson 10 , simpson 50 tương ứng với n = 10, 50

`fprintf('Do chenh lech truong hop n = 1:\n');`
`fprintf('tich phan hinh thang: %f\n', thang1);`
`fprintf('tich phan hinh thang: %f\n', simpson1);`
	In ra màn hình độ chênh lệch của tpht vs simpson so với integral tại n=1

`fprintf('Do chenh lech truong hop n = 10:\n');`
`fprintf('tich phan hinh thang: %f\n', thang10);`
`fprintf('tich phan simpson: %f\n', simpson10);`
	In ra màn hình độ chênh lệch của tpht vs simpson so với integral tại n=10

`fprintf('Do chenh lech truong hop n = 50:\n');`
`fprintf('tich phan hinh thang: %f\n', thang50);`
`fprintf('tich phan simpson: %f\n', simpson50);`
	In ra màn hình độ chênh lệch của tpht vs simpson so với integral tại n=50

Kết quả:

```
Do chenh lech truong hop n = 1:
tich phan hinh thang: 0.243637
tich phan hinh thang: 0.103392
Do chenh lech truong hop n = 10:
tich phan hinh thang: 0.002553
tich phan simpson: 0.000004
Do chenh lech truong hop n = 50:
tich phan hinh thang: 0.000102
tich phan simpson: 0.000000
```

Nhận xét:

+ Có thể thấy N càng lớn thì độ chênh lệch với kết quả chính xác thông qua hàm integral là càng nhỏ

+ Ở bất kì điểm N nào (1 hay 10 hay 50) thì độ chênh lệch với kết quả chính xác của phương pháp Simpson là nhỏ hơn của phương pháp hình thang, cụ thể tại N = 50 thì độ chênh lệch là 0 nghĩa là N càng lớn thì độ chênh lệch càng chính xác ($n \to +\infty \gg chenhlech \to 0$

⇨ Từ đó rút ra kết luận là phương pháp Simpson có độ chính xác cao hơn phương pháp hình thang