Numerical Computations

Computer Assignment 3

Professor : Jamal Kazazi

Shahaboddin Sheybani
810101454

2023

Q1a)

Main Orders :

خط 1 تا 2 : حذف دستورات و دیتای قبلی

خط 3 و 4 : تعریف تابعی که برابر y’ می باشد و تعریف پاسخ تحلیلی تابع برای بررسی مقدار واقعی آن

خط 5: تعریف ماتریس h و ذخیره تمامی طول گام ها در آن

خط 6 : تعریف ماتریس مقادیر به دست آمده f(1) و ذخیره در f1 values

خط 7: تعریف ماتریس خطای مقدار f(1) به ازای هر h

خط 8 : مقداردهی مقدار واقعی f(1) و ذخیره در f1_real

خط 9 تا 12 : حلقه for که به ازای هر h مقدار f1_value(i) از خروجی تابع euler به ازای ورودی 1 و 0 و مقدار تابع در شروع 3 و ماتریس h و تابع f به دست می آید و f1_error(i) که برابر قدر مطلق اختلاف مقدار واقعی و به دست آمده است

خط 13 : دستور plot برای رسم نمودار خطاها برحسب h

Euler Function :

خط 14 : تعریف تابع حل معادله به روش اویلر با دریافت نقطه شروع و پایان و مقدار تابع در نقطه شروع و طول گام و تابع مورد نظر که y’ برابر آن است

خط 15 : تعریف مقدار n که برابر تعداد بازه ها است

خط 16 و 17 : تعریف ماتریس x و y ها به طول n است

خط 18 تا 21 : حلقه for برای ریختن مقادیر x که با طول گام مورد نظر حرکت کنند و در ماتریس x ذخیره شوند

خط 22 : ذخیره مقدار اولیه y برابر با مقدار شروعی داده شده

خط 23 تا 25 : مقداردهی y در هر نقطه که با طول گام تغییر کند همانند فرمول اویلر

$Y(i+1) = y(i) + hf(xi , yi)$

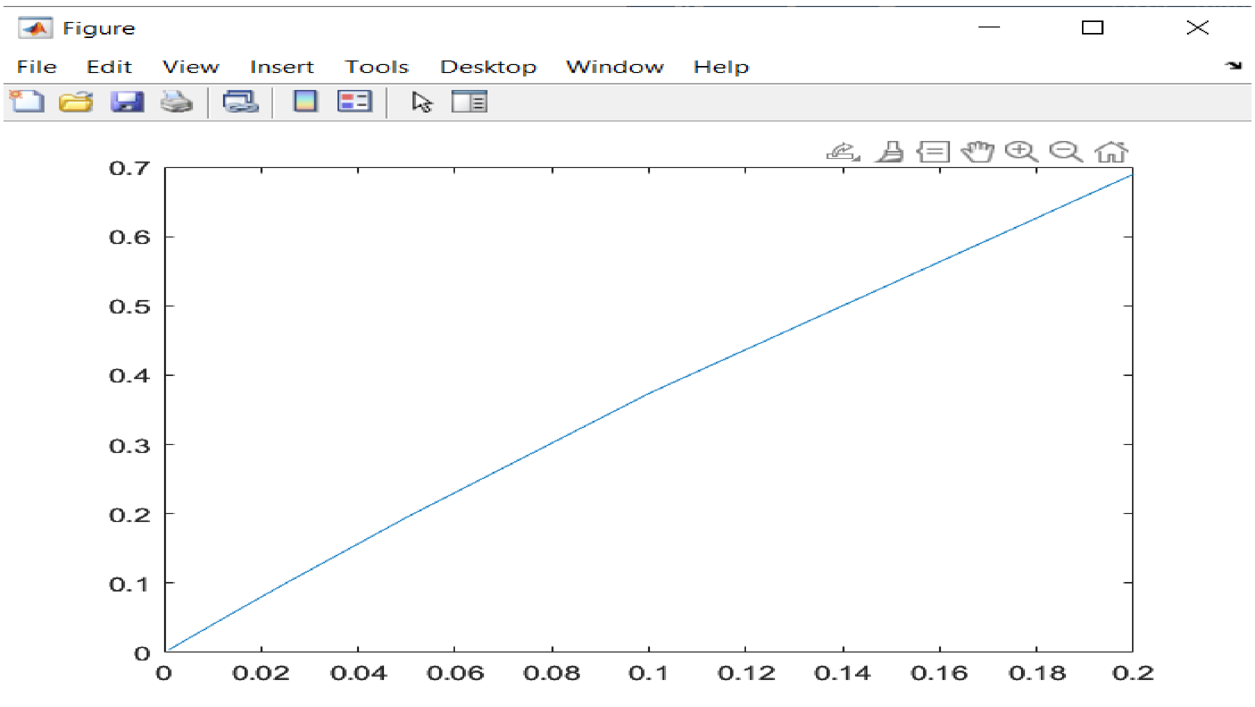
خط 26 : خروجی را برابر آخرین نقطه ماتریس y قرار می دهیم
خط 27 : پایان تابع

Result :

f1_value

1x11 double

	1	2	3	4	5	6	7	8	9	10	11	12
1	9.4650	9.7812	9.9599	10.0552	10.0748	10.1045	10.1144	10.1295	10.1421	10.1467	10.1508	
2												
3												



Q1b)

Main Orders :

خط 1 تا 13 : دستورات همانند قسمت قبلی می باشد فقط مقدار آن برابر خروجی تابع RK2 که رانگ کوتای مرتبه 2 می باشد قرار می دهیم

Runge Kutta 2 Function :

خط 14 : تعریف تابع حل معادله به روش اویلر با دریافت نقطه شروع و پایان و مقدار تابع در نقطه شروع و طول گام و تابع مورد نظر که y′ برابر آن است

خط 15 : تعریف مقدار n که برابر تعداد بازه ها است

خط 16 و 17 : تعریف ماتریس x و y ها به طول n است

خط 18 تا 21 : حلقه for برای ریختن مقادیر x که با طول گام مورد نظر حرکت کنند و در ماتریس x ذخیره شوند

خط 22 : ذخیره مقدار اولیه y برابر با مقدار شروعی داده شده

خط 23 تا 27 : حلقه for که مقدار k1 k2 را محاسبه کند و در پایان :

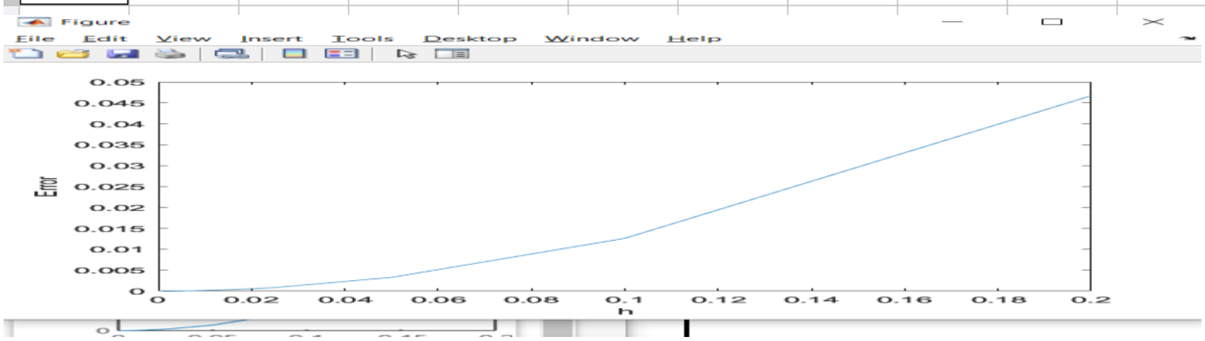
Y(i+1) = y(i) + 0.5(K1 + K2)

خط 28 : خروجی برابر آخرین عضو ماتریس y می باشد

Result:

1x11 double

	1	2	3	4	5	6	7	8	9	10	11
1	0.0467	0.0126	0.0033	8.3365e-04	5.3555e-04	2.1038e-04	1.3490e-04	5.2843e-05	1.3242e-05	5.4284e-06	1.3581e-06



Q1c)

Main Orders :

خط 1 تا 13 : دستورات همانند قسمت قبلی می باشد فقط مقدار آن برابر خروجی تابع RK2 که رانگ کوتای مرتبه 2 می باشد قرار می دهیم

Runge Kutta 4 Function :

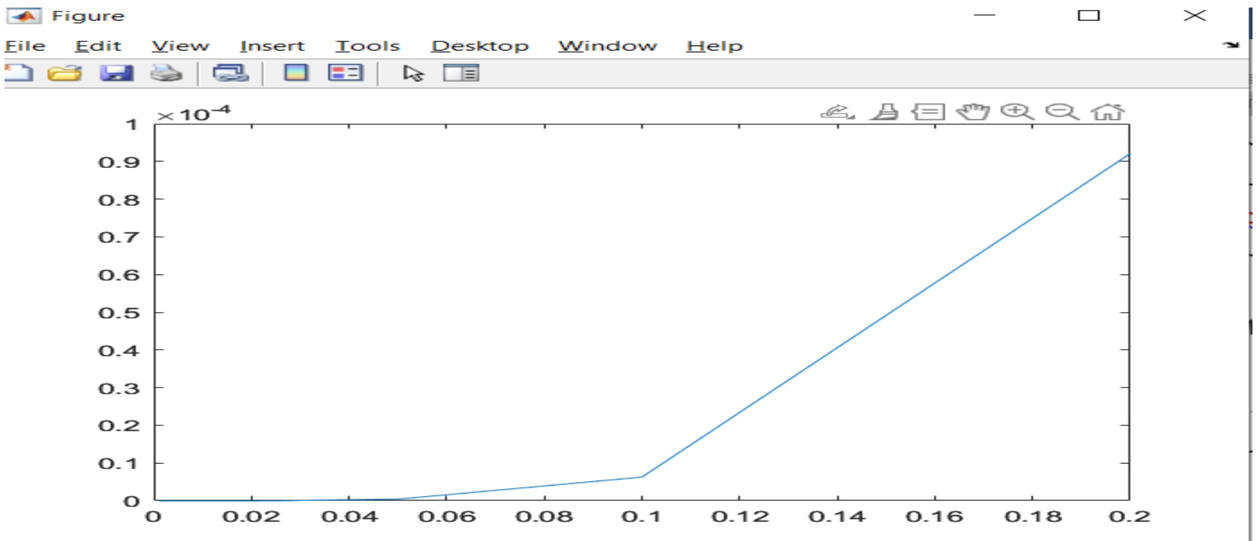
خط 14 تا 22 : توضیحات همانند قسمت قبلی است زیرا تفاوتی در کدها وجود ندارد

خط 23 تا 28 : محاسبه k1 تا k4 همانند ضرایب رانگ کوتا و در پایان خروجی مقدار طبق :

Y(i+1) = y(i) + (k1 + 2k2 + 2k3 + k4)/6

خط 30 : تساوی خروجی تابع با آخرین عضو ماتریس y

Result:



f1_error

1x11 double

	1	2	3	4	5	6	7	8	9	10	11
1	9.2076e-05	6.2530e-06	4.0741e-07	2.5999e-08	1.0693e-08	1.6419e-09	6.7393e-10	1.0316e-10	6.4766e-12	1.0978e-12	5.3291e-14

Q2a)

Main Orders:

خط 1 و 2 : دستورات اولیه حذف دستورات و دیتای قبلی

خط 3 : تعریف تابعی که قرار است انتگرال آن محاسبه شود

خط 4 : مقدار I را برابر خروجی تابع ذوزنقه ای با دریافت نقاط شروع 0 و 1 با طول گام 0.1 و تابع f

Trapezius Function:

خط 5 : تعریف تابع ذوزنقه ای با دریافت نقاط شروع و طول گام و تابع

خط 6 : تعریف مقدار n برابر با تعداد بازه ها و یکی بیشتر که برابر

تعداد داده ها است
خط 7تا11 : تعریف ماتریس x که مقدار ورودی هارا با افزایش به اندازه h در ماتریس x ذخیره کنذ
خط 12 : تعریف مقدار صورت کسر ذوزنقه ای و مقداردهی اولیه صفر برای sigmaF
خط 13 تا 19 : تعریف یک حلقه for که مقدار sigmaF را آپدیت کند به ازای نقاط اول و آخر ماتریس x خود مقدار تابع و در غیر این صورت دو برابر مقدار را بیفزاید
خط 20 : خروجی تابع را برابر مقدار زیر قرار می دهیم

I = (h/2)(f(0) + 2f(x1) + 2f(x3) + … f(1))

Result:

I = 0.6298

Q2b)

Main Orders :

خط 1 و 2 : دستورات اولیه حذف دستورات و دیتای قبلی
خط 3 : تعریف تابعی که قرار است انتگرال آن محاسبه شود
خط 4 : مقدار I را برابر خروجی تابع سیمپسون با دریافت نقاط شروع 0 و 1 با طول گام 0.1 و تابع f

Simpson Function :

خط 5 تا 12 : دستورات همانند قسمت قبلی می باشد
خط 13 تا 21 : محاسبه مقدار صورت کسر سیمپسون که ضریب مقدار نقطه اول و آخر برابر 1 و نقاط زوج 4 و ضریب نقاط فرد برابر 2 خواهد بود
خط 22 : خروجی برابر مقدار زیر خواهد بود

$(h/3)(f(0) + 4f(x1) + 2f(x2) + 4f(x3) + ...f(1))$

خط 23 : پایان تابع

Result :

```
I = 0.6321
```

Q2c)

Main Orders :

خط 1 و 2 : دستورات اولیه حذف دیتای قبلی و دستورات

خط 3 : تعریف تابع f برابر تابع صورت سوال

خط 4 و 5 : برابر قرار دادن خروجی تابع گاوس دو نقطه ای و سه نقطه ای با I و J

Double Gauss Function :

خط 6 : تعریف تابع گاوس دونقطه ای با دریافت نقطه شروع پایان

خط 7 : خروجی را برابر مقدار $f(1/sqrt(3)) + f(-1/sqrt(3))$ باید برابر دهیم . نکته این است که برای رساندن به فرم +-1 باید تغییر متغیر انجام دهیم که به شکل زیر است

$X = (b-a)u/2 + (b+a)$

که مقدار دیفرانسیل u هم به پشت خروجی اضافه می شود

Triple Gauss Function :

خط 10 : تعریف تابع 3نقطه ای گاوس

خط 11 : تعریف متغیر اول که برابر f(-sqrt(3/5)) خواهد بود و ضرایب تغییر متغیر در آن وارد شود

خط12 : تعریف متغیر اول که برابر f(0) خواهد بود و ضرایب تغییر متغیر در آن وارد شود و به فرم داده شده تبدیل شود

خط 11 : تعریف متغیر اول که برابر f(sqrt(3/5)) خواهد بود و ضرایب تغییر متغیر در آن وارد شود

خط 14 : تعریف خروجی که برابر مقدار انتگرال است و به فرم زیر است

1/9 (5f(-sqrt(3/5)) + 8f(0) + 5f(-sqrt(3/5)))

Result :

```
clc
clear
f = @(x) 2*x*exp(-(x^2));
I = double_gauss(0,1,f)
J = triple_gauss(0,1,f)
```

```
I = 0.6255
J = 0.6323
```

```
function output = double_gauss(a,b,f)
output = 0.5*(b - a)*(f(-(b - a)/(2*sqrt(3))+(b + a)/2
end

function output = triple_gauss(a,b,f)
x1 = 0.5*(b-a)*(-sqrt(3/5)) + 0.5*(b+a);
x2 = 0.5*(b+a);
x3 = 0.5*(b-a)*(sqrt(3/5)) + 0.5*(b+a);
output = (1/(2*9))*(b-a)*(5*f(x1) + 8*f(x2) + 5*f(x3))
end
```