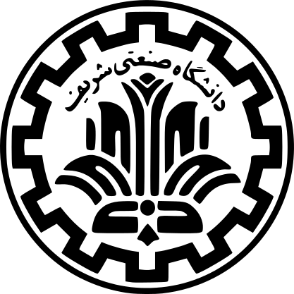
**باسمه تعالی**

****

**دانشگاه صنعتی شریف**

**دانشکده مهندسی برق**

**گزارش پروژۀ درس محاسبات عددی**

**نگارنده**

**بهراد منیری**

**95109564**

**استاد**

**دکتر ایمان‌ غلام‌پور**

**بهمن 1396**

**محاسبه انتگرال**

تابع integ عمل محاسبۀ عددی انتگرال را انجام می‌دهد. این تابع می‌تواند انتگرال را به چهار روش مستطیلی، ذوزنقه‌ای، نقطه میانی و سیمسون محاسبه کند.

integ( f, N, lower, upper, method)

در این تابع:

f فانکشن هندل تابعی است که می‌خواهیم انتگرال آن را محاسبه کنیم. (برای توان باید از .^ استفاده کرد و الخ.)

N تعداد نقاطی است که در محاسبه انتگرال از آن استفاده می‌کنیم.

Lower کران پایین انتگرال معین است.

Upper کران بالای انتگرال معین است.

Method روش انتگرال گیری است.

Method مقادیر زیر را قبول می‌کند:

|  |  |
| --- | --- |
| **عملکرد** | **Method** |
| محاسبۀ انتگرال به روش مستطیلی | ‘rect’ |
| محاسبۀ انتگرال به روش نقطه میانی | ‘midpoint’ |
| محاسبۀ انتگرال به روش سیمسون | ‘simpson' |
| محاسبۀ انتگرال به روش ذوزنقه | ‘trapz’ |

**مثال یک**: محاسبۀ با روش سیمسون و استفاده از 100 نقطه

%% Example One: integ

f = @(x) sin(x);

I = integ(f, 100, 0, 1, 'simpson');

disp(I)

خروجی:

مقدار واقعی:

در این محاسبه خطا از کوچکتر است.

**محاسبه مشتق**

تابع derv عمل محاسبۀ عددی مشتق را انجام می‌دهد. این تابع می‌تواند از روش‌های دو، سه و چهار نقطه‌ای پیشرو و پسرو و روش دو و چهار نقطه‌ای متقارن مشتق را به صورت عددی به دست بیاورد.

derv( f, x0, h, method)

در این تابع:

f فانکشن هندل تابعی است که می‌خواهیم مشتق آن را محاسبه کنیم. (برای توان باید از .^ استفاده کرد و الخ.)

X0 نقطه‌ای است که مشتق در آن محاسبه می‌شود.

h طول گام های ما در محاسبه مشتق است.

Method مقادیر زیر را قبول می‌کند:

|  |  |
| --- | --- |
| **عملکرد** | **Method** |
| محاسبه مشتق به روش چهار نقطه‌ای متقارن | ‘4sym’ |
| محاسبه مشتق به روش دو نقطه‌ای متقارن | ‘2sym’ |
| محاسبه مشتق به روش چهار نقطه‌ای پیشرو | ‘4forward’ |
| محاسبه مشتق به روش سه نقطه‌ای پیشرو | ‘3forward’ |
| محاسبه مشتق به روش دو نقطه‌ای پیشرو | ‘2forward’ |
| محاسبه مشتق به روش چهار نقطه‌ای پسرو | ‘4backward’ |
| محاسبه مشتق به روش سه نقطه‌ای پسرو | ‘3backward’ |
| محاسبه مشتق به روش دو نقطه‌ای پسرو | ‘2backward’ |

**مثال دو**: محاسبۀ در نقطه x = 1 با روش مشتق پیشرو چهار نقطه‌ای.

%% Example Two: derv

clear

disp('Example Two');

f = @(x) 1./(1+x.^2);

D = derv(f, 1, 0.001, '4forward');

disp(D)

خروجی:

مقدار واقعی:

در این محاسبه خطا از مرتبۀ است.

**محاسبه چند جمله‌ای درون‌یاب**

تابع newton\_interpolation چند جمله‌ای درون یاب را به کمک روش نیوتن محاسبه می‌کند.

newton\_interpolation(x,y,p)

در این تابع:

نقاطی هستند که به کمک آنها تابع درون‌یابی می‌شود.

P برداری از اعداد است که تابع مقدار چندجمله‌ای درونیاب را در آنها باز می‌گرداند.

**مثال سه**: به دست آوردن و رسم چندجمله‌ای درونیاب sin(x) در بازه به کمک شش نقطۀ

[0, pi/4, pi/2, pi, 3\*pi/2, 2\*pi]

%% Example Three : newton\_interpolation

t = 0:0.1:2.5\*pi;

x1 = [0, pi/4, pi/2, pi, 3\*pi/2, 2\*pi];

f1 = newton\_interpolation(x1, sin(x1), t);

figure

hold on

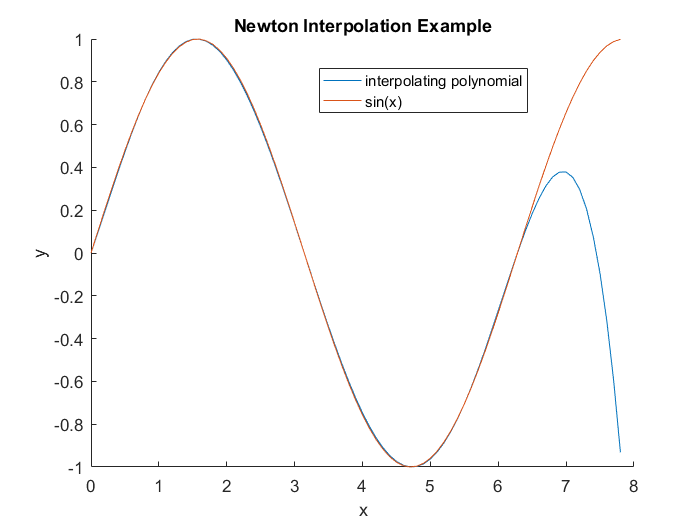
title('Newton Interpolation Example'); ylabel('y'); xlabel('x');

plot(t,f1);

plot(t, sin(t));

legend('interpolating polynomial','sin(x)');

خروجی تابع:



این چندجمله در بازه مورد نظر با دقت بسیار بالایی تابع را تقریب می‌زند ولی همانطور که دیده می‌شود برای برونیابی مناسب نیست.

**محاسبه اسپلاین مکعبی طبیعی**

تابع newton\_interpolation چند جمله‌ای درون یاب را به کمک روش نیوتن محاسبه می‌کند.

cubic\_spline(x, y, t)

در این تابع:

نقاطی هستند که به کمک آنها تابع درون‌یابی می‌شود.

P برداری از اعداد است که تابع مقدار اسپلاین در آنها باز می‌گرداند.

**مثال چهار**: به دست آوردن و اسپلاین در بازه به کمک شش نقطۀ

[0, 2, 4, 6, 8, 10]

%% Example Four: cubic\_spline

clear

t = 0:0.01:10;

x1 = 0:2:10;

y1 = exp(-x1);

f1 = cubic\_spline(x1, y1, t);

figure

hold on

title('Spline Example'); ylabel('y'); xlabel('x');

plot(t,f1);

plot(t, exp(-t));

legend('spline','exp(-x)');

خروجی تابع:



این چندجمله در بازه مورد نظر با دقت بسیار بالایی تابع را تقریب می‌زند.

**حل دستگاه معادلات خطی به روش حذفی گاوس**

تابع newton\_interpolation چند جمله‌ای درون یاب را به کمک روش نیوتن محاسبه می‌کند.

gauss\_elimination(A,b)

دستور فوق دستگاه ماتریسی را حل کرده و x را برمیگرداند. در صورت عدم وجود جواب تابع عبارت 'det(A) = 0'

را چاپ می‌کند.

**مثال پنج**: حل معادلۀ

%% Example Four: gauss\_elimination

clear

A = [1 2 3; 4 7 8; 12 0 2];

b = [1; 2; 3];

sol = gauss\_elimination(A, b);

خروجی تابع:

این جواب دقیقا جواب تابع درونی متلب نیز هست.

**حل دستگاه معادلات خطی با روش های تکراری ژاکوبی و گاوس-سایدل**

تابع newton\_interpolation چند جمله‌ای درون یاب را به کمک روش نیوتن محاسبه می‌کند.

matiter( A, b, x, N, method )

دستور فوق دستگاه ماتریسی را حل کرده و y را برمیگرداند. x حدس اولیه ماست.

Method مقادیر زیر را قبول می‌کند:

|  |  |
| --- | --- |
| **عملکرد** | **Method** |
| حل دستگاه به روش ژاکوبی | ‘jacobi’ |
| حل دستگاه به روش سایدل | ‘seidel’ |

**مثال شش**: حل معادلۀ

با حدس اولیۀ صفر، به هر دو روش با صد تکرار.

clear

disp('Example Six');

A = [10 1 1; 4 55 2; 2 10 2];

b = [1; 2; 3];

sol1 = matiter(A, b, [0;0;0], 100, 'jacobi');

disp(sol1)

sol2 = matiter(A, b, [0;0;0], 100, 'seidel');

disp(sol2)

خروجی تابع برای هر دو روش

این جواب دقیقا جواب تابع درونی متلب نیز هست.

**به دست آوردن ویژه مقادیر یک ماتریس**

تابع eigenval با استفاده از روش توانی بزرگترین مقدار ویژه ماتریس و بردار ویژه متناظر با آن را برمیگرداند.

[a,b] = eigenval(A, N)

در دستور بالا تابع روش توانی را N بار بر ماتریس A اجرا می‌کند و در خروجی a برابر مقدار ویژه بیشینه و b برابر بردار ویژه متناظر با آن است.

**مثال هفت**: پیدا کردن مقدار ویژه بیشینۀ ماتریس زیر

%% Example Seven: eigenval

A = [7 4 1; 4 4 4; 1 4 7];

[a,b] = eigenval(A, 100);

همانطور که انتظار داریم خروجی برابر 12 و [1,1,1] است.

**حل معادلات**

**توابع زیر برای حل معادلات غیرخطی نوشته شده‌اند. به دلیل اینکه برخی از این روش ها تفاوت های اساسی با بقیه دارند، همه روش ها در یک تابع تجمیع نشدند و در عوض در این بخش چندین تابع نوشته شده است.**

* روش تنصیف:

تابع bisection معادله را به روش تنصیف با دقت خواسته شده حل می‌کند.

bisection(f, a, b, Nmax, error)

با کد بالا معادلۀ f(x) = 0 با در نظر رفتن دو نقطه a و b به عنوان نقاط شروع روش تنصیف حل می‌شود.

Nmax بیشترین تعداد تکراری است که می‌خواهیم تابع انجام دهد.

Error اگر تابع به این دقت برسد خروجی را بازگردانده و تکرار را متوقف می‌کند.

**مثال هشت**: حل معادلۀ xsin(x) – 12 = 0 با دقت 0.01 و حداکثر تکرار 100.

%% Example Eight: bisection

clear

bisection(@(x) x\*sin(x) - 2 , -227, -225, 100, 0.0001)

خروجی تابع -226.2036 است که با دقت داده شده با جواب تابع داخلی متلب یعنی

-226.20351276934028495859031358348 همخوانی دارد.

* روش نابجایی:

تابع regulafalsi معادله را به روش نابجایی با دقت خواسته شده حل می‌کند.

regulafalsi(f, a, b, Nmax, error)

با کد بالا معادلۀ f(x) = 0 با در نظر رفتن دو نقطه a و b به عنوان نقاط شروع روش تنصیف حل می‌شود.

Nmax بیشترین تعداد تکراری است که می‌خواهیم تابع انجام دهد.

Error اگر تابع به این دقت برسد خروجی را بازگردانده و تکرار را متوقف می‌کند.

**مثال نه**: حل معادلۀ xsin(x) – 12 = 0 با دقت 0.01 و حداکثر تکرار 100.

%% Example Nine: regulafalsi

clear

regulafalsi(@(x) x\*sin(x) - 2 , -227, -225, 100, 0.0001)

خروجی تابع -226.2035 است که با دقت داده شده با جواب تابع داخلی متلب یعنی

-226.20351276934028495859031358348 همخوانی دارد.

* روش نیوتن:

تابع newton معادله را به روش نیوتن با دقت خواسته شده حل می‌کند.

newton(f, x0, Nmax, error)

با کد بالا معادلۀ f(x) = 0 با در نظر رفتن x0 به عنوان نقاط شروع روش تنصیف حل می‌شود.

Nmax بیشترین تعداد تکراری است که می‌خواهیم تابع انجام دهد.

Error اگر تابع به این دقت برسد خروجی را بازگردانده و تکرار را متوقف می‌کند.

**مثال ده**: حل معادلۀ xsin(x) – 12 = 0 با دقت 0.01 و حداکثر تکرار 100.

%% Example Ten: newton

clear

newton(@(x) x\*sin(x) - 2 , -227, -225, 100, 0.0001)

خروجی تابع -226.2035 است که با دقت داده شده با جواب تابع داخلی متلب یعنی

-226.20351276934028495859031358348 همخوانی دارد.

* روش وتری:

تابع secant معادله را به روش نیوتن با دقت خواسته شده حل می‌کند.

secabt(f, a, b, Nmax, error)

با کد بالا معادلۀ f(x) = 0 با در نظر رفتن a و b به عنوان نقاط شروع روش تنصیف حل می‌شود.

Nmax بیشترین تعداد تکراری است که می‌خواهیم تابع انجام دهد.

Error اگر تابع به این دقت برسد خروجی را بازگردانده و تکرار را متوقف می‌کند.

**مثال ده**: حل معادلۀ xsin(x) – 12 = 0 با دقت 0.01 و حداکثر تکرار 100.

%% Example Eleven: Secant

clear

secant(@(x) x.\*sin(x) - 2 , -227, -225, 100, 0.0001)