

7. Numerical integration and differentiation

In the functions of integration and differentiation in Matlab there are implemented ~~implements~~ various numerical algorithms.

7.1. Integration by the trapezoidal rule

The trapezoidal rule works by approximating the region under the graph of the function $f(x)$ as a [trapezoid](#) and its area is calculated as follows:

$$\int_a^b f(x)dx \approx (b - a) \left[\frac{f(a) + f(b)}{2} \right]. \quad (7.1)$$

In Matlab the integration of experimental data by trapezoidal interpolation by using command **inttrap**.

Q = trapz(X,Y)

integrates Y with spacing increment X. By default, trapz operates on the first dimension of Y whose size does not equal 1. length(X) must be equal to the size of this dimension . If X is a scalar, then trapz(X,Y) is equivalent to X*trapz(Y).

Problem 7.1

Calculate the definite integral

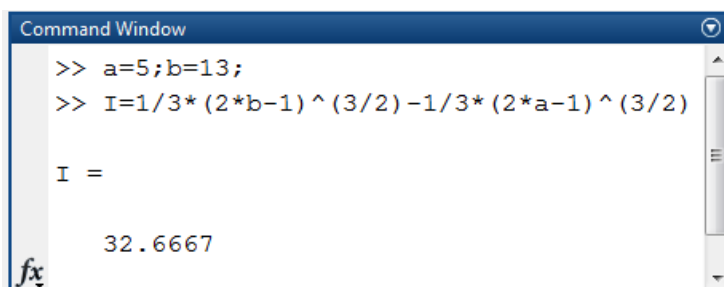
$$\int_5^{13} \sqrt{2x - 1} dx.$$

This integral is easily reduced to the list of integrals

$$\int_5^{13} \sqrt{2x - 1} dx = \frac{\sqrt{(2x - 1)^3}}{3},$$

so we calculate it using the Newton - Leibniz formula

$$\int_a^b f(x) = F(b) - F(a).$$



```
Command Window
>> a=5;b=13;
>> I=1/3*(2*b-1)^(3/2)-1/3*(2*a-1)^(3/2)

I =

    32.6667
```

Now apply the trapezoidal rule for finding given the definite integral. Consider a few options to solve this problem. In the first case, the interval of integration is divided into sections with one step in the second 0.5 and in the third 0.1. It is easy to notice the more points the partition has, the more accurate the value of the desired integral is:

```
Command Window
>> x=a:b; y=sqrt(2*x-1);
>> trapz(x,y)

ans =

fx 32.6556
```

```
Command Window
>> h=0.5; x=a:h:b; y=sqrt(2*x-1);
>> trapz(x,y)

ans =

fx 32.6639
```

```
Command Window
>> h=0.1; x=a:h:b; y=sqrt(2*x-1);
>> trapz(x,y)

ans =

fx 32.6666
```

7.2. Integration of external functions

The most universal integration in Matlab command is:

q = integral(fun,xmin,xmax)

numerically integrates function fun from xmin to xmax using global adaptive quadrature and default error tolerances.

Problem 7.2.

Calculate the integral of the problem 7.1

```
Command Window
>> fun=@(x) (2*x-1).^0.5;
>> integral(fun,5,13)

ans =

fx 32.6667
```

Problem 7.3.

Calculate the integral

$$\int_0^1 \frac{t^2}{\sqrt{(3 + \sin(t))}} dt.$$

```

Command Window
>> fun=@(t)t.^2./sqrt(3+sin(t));
>> I=integral(fun,0,1)

I =

    0.1741

```

7.3. Approximate differentiation based on the Newton interpolation formula

A Newton polynomial is the interpolation polynomial for a given set of data points in the Newton form. The Newton polynomial is sometimes called Newton's divided differences interpolation polynomial because the coefficients of the polynomial are calculated using divided differences.

The idea of numerical differentiation lies in the fact that the function $y(x)$ of approximately replaced by a polynomial interpolation Newton. The function $y(x)$ set by x_0, x_1, \dots, x_n is arranged consecutively with equal space for interval $[a, b]$ with help values of $y_i = f(x_i)$. The Newton Polynomial built for the system nodes $x_0, x_1, \dots, x_k (k \leq n)$. And calculate derivatives.

$$y'(x_0) = \frac{1}{h} \left(\Delta y_0 - \frac{\Delta^2 y_0}{2!} + \frac{\Delta^3 y_0}{3!} - \frac{\Delta^4 y_0}{4!} + \frac{\Delta^5 y_0}{5!} - \dots \right), \quad (7.2)$$

In practice, the approximate differentiation is used mainly for functions defined in a table.

In Matlab, the numerical differentiation implemented by command:

Y = diff(X)

calculates differences between adjacent elements of X along the first array dimension whose size does not equal 1:

If X is a vector of length m, then **Y = diff(X)** returns a vector of length m-1. The elements of Y are the differences between adjacent elements of X.

$$Y = [X(2)-X(1) \ X(3)-X(2) \ \dots \ X(m)-X(m-1)]$$

If X is a nonempty, nonvector p-by-m matrix, then **Y = diff(X)** returns a matrix of size (p-1)-by-m, whose elements are the differences between the rows of X.

$$Y = [X(2,:)-X(1,:); \ X(3,:)-X(2,:); \ \dots \ X(p,:)-X(p-1,:)]$$

If X is a 0-by-0 empty matrix, then **Y = diff(X)** returns a 0-by-0 empty matrix.

Y = diff(X,n)

calculates the nth difference by applying the **diff(X)** operator recursively n times.

In practice, this means **diff(X,2)** is the same as **diff(diff(X))**.

Problem 7.4.

Find $y'(50)$ of function $y = \lg(x)$ is given in table.

```
Command Window
>> h=5; x=50:h:65;
>> y=log10(x)

y =
    1.6990    1.7404    1.7782    1.8129

>> dy=diff(y)

dy =
    0.0414    0.0378    0.0348

>> ddy=diff(y,2)

ddy =
   -0.0036   -0.0030

>> dddy=diff(y,3)

dddy =
    5.7767e-04
```

```
Command Window
>> % Approximate value of y'(50) by eq.(7.2)
>> Y=(dy(1)-ddy(1)/2+dddy(1)/3)/h

Y =
    0.0087

>> % The value y'(50) for lg'(x)=1/ln(10)/x
>> 1/log(10)/x(1)

ans =
    0.0087
```

```

Command Window
>> % The approximate value of y'(x), x=50,55,60
>> Y(1)= (dy(1)-ddy(1) ./2.+ddy(1) ./3) ./h;
>> Y(2)= (dy(2)-ddy(2) ./2.+ddy(1) ./3) ./h;
>> Y(3)= (dy(3)-ddy(2) ./2.+ddy(1) ./3) ./h;
>> Y

Y =

    0.0087    0.0079    0.0073

>> %The value y'(x), x=50,55,60, for lg'(x) = 1/ln(10)/x
>> (1/log(10)) ./x(1:end-1)

ans =

fx    0.0087    0.0079    0.0072

```

7.4. Calculating the derivative of the function at the point

We can use symbolic variables and function:

syms var1 ... varN

creates symbolic variables var1 ... varN. Separate variables by spaces.

The first create symbolic variable, then symbolic function. After then we use function **diff** and get differential of the function in symbolic form. And finally we substitute the value and get the result

For symbolic substitution we can use command:

subs(s,old,new)

returns a copy of s replacing all occurrences of old with new, and then evaluating s

Problem 7.5.

Calculate $f'(1)$, if

$$f(x) = (x + 2)^3 + 5x.$$

```

Command Window
>> syms x
>> y(x)=(x+2)^3+5*x;
>> dy(x)=diff(y(x))

dy(x) =

3*(x + 2)^2 + 5

>> dy(1)

ans =

32

```

Problem 7.6.

Calculate $f'(1)$ in the points 0, 1, 2, 3 for

$$f(x) = (x + 2)^3 + 5x.$$

```

Command Window
>> syms x
>> y(x)=(x+2)^3+5*x;
>> dy(x)=diff(y(x))

dy(x) =

3*(x + 2)^2 + 5

>> s=[0 1 2 3];
>> dy(s)

ans =

[ 17, 32, 53, 80]

```

Problem 7.7.

Given a function of many variables

$$y(x_1, x_2, x_3) = x_1 x_2^{x_3} + x_1^2 x_3.$$

Calculate

$$\frac{dy}{dx_1}, \frac{dy}{dx_2}, \frac{dy}{dx_3}$$

in the points (1, 2, 3).

```
Command Window
>> syms x1 x2 x3
>> y(x1,x2,x3)=x1*x2^x3+x3*x1^2

y(x1, x2, x3) =

x1*x2^x3 + x1^2*x3

>> f(1)=diff(y,'x1');
>> f(2)=diff(y,'x2');
>> f(3)=diff(y,'x3');
>> f

f =

[ 2*x1*x3 + x2^x3, x1*x2^(x3 - 1)*x3, x1^2 + x1*x2^x3*log(x2) ]

>> x=[1 2 3];
>> res=subs(subs(subs(f,x1,x(1)), x2,x(2)), x3, x(3))

res =

[ 14, 12, 8*log(2) + 1 ]
```