

## ChE 400: Applied Chemical Engineering Calculations

### Tutorial 4: Numerical Solution of Integrals

Gerardine G. Botte

This handout contains information and examples on:

- How to develop an spreadsheet in excel to solve integrals numerically using:
  - Trapezoidal rule
- \* Symbolic integration in Matlab
- \* Numerical integration in Matlab

**1. Problem 1:** Develop a spreadsheet in Excel to compute numerically the following integral, using trapezoidal rule:

$$N(x) = \int_{-2}^1 \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}} dx$$

Discretize your integrant using  $h=0.2$ .

**Solution:**

1. Discretize your function showing the trapezoids
2. Calculate “n” for the given h (answer:  $n = 15$ )
3. Each trapezoids area can be calculated using Eq. (9) of class notes:

$$I_i = \frac{(y_{i-1} + y_i)}{2} h$$

4. The integral is given by using Eq. (10) of class notes: ( $I = \sum_{i=1}^n I_i$ ). Therefore I is

given by:

$$h = 0.2 \qquad n = 15$$

node, i	x	$f_i(x)$ or $y_i$	h	$(f_{i-1}(x)+f_i(x))*0.5*h$ or $I_i$
0	-2.00	0.05399		
1	-1.80	0.07895	0.2	0.01329
2	-1.60	0.11092	0.2	0.01899
3	-1.40	0.14973	0.2	0.02606
4	-1.20	0.19419	0.2	0.03439
5	-1.00	0.24197	0.2	0.04362
6	-0.80	0.28969	0.2	0.05317
7	-0.60	0.33322	0.2	0.06229
8	-0.40	0.36827	0.2	0.07015
9	-0.20	0.39104	0.2	0.07593
10	0.00	0.39894	0.2	0.07900
11	0.20	0.39104	0.2	0.07900
12	0.40	0.36827	0.2	0.07593
13	0.60	0.33322	0.2	0.07015
14	0.80	0.28969	0.2	0.06229
15	1.00	0.24197	0.2	0.05317

$$I = 0.81743$$

Obtained by adding all  $I_i$

**2. Numerical Integration using Matlab:** The area beneath a section of a function  $F(x)$  can be determined by numerically integrating  $F(x)$ , a process referred to as quadrature. The MATLAB quadrature functions are:

*quad* (use adaptive Simpson quadrature)  
*quadl* (use adaptive Lobatto quadrature)  
*dblquad* (numerically evaluate double integral)

Syntax for *quad*

`q = quad(fun,a,b)`  
`q = quad(fun,a,b,tol)`  
`q = quad(fun,a,b,tol,trace)`  
`q = quad(fun,a,b,tol,trace,p1,p2,...)`  
`[q,fcnt] = quadl(fun,a,b,...)`

*Description*

Quadrature is a numerical method used to find the area under the graph of a function, that is, to compute a definite integral.

`q = quad(fun,a,b)` approximates the integral of function `fun` from `a` to `b` to within an error of  $10^{-6}$  using recursive adaptive Simpson quadrature. *fun* **accepts a vector  $x$**  and returns a vector `y`, the function `fun` evaluated at each element of `x`.

`q = quad(fun,a,b,tol)` uses an absolute error tolerance `tol` instead of the default which is  $1.0e-6$ . Larger values of `tol` result in fewer function evaluations and faster computation, but less accurate results.

`q = quad(fun,a,b,tol,trace)` with non-zero `trace` shows the values of `[fcnt a b-a Q]` during the recursion.

`q = quad(fun,a,b,tol,trace,p1,p2,...)` provides for additional arguments `p1,p2,...` to be passed directly to function `fun`, `fun(x,p1,p2,...)`. Pass empty matrices for `tol` or `trace` to use the default values.

`[q,fcnt] = quad(...)` returns the number of function evaluations.

The function `quadl` may be more efficient with high accuracies and smooth integrands.

You can specify *fun* three different ways:

*A string expression involving a single variable*

`Q = quad('1./(x.^3-2*x-5)',0,2);`

*An inline object*

`F = inline('1./(x.^3-2*x-5)');`  
`Q = quad(F,0,2);`

*A function handle*

`Q = quad(@myfun,0,2);`  
 where `myfun.m` is an M-file.  
 function `y = myfun(x)`  
`y = 1./(x.^3-2*x-5);`

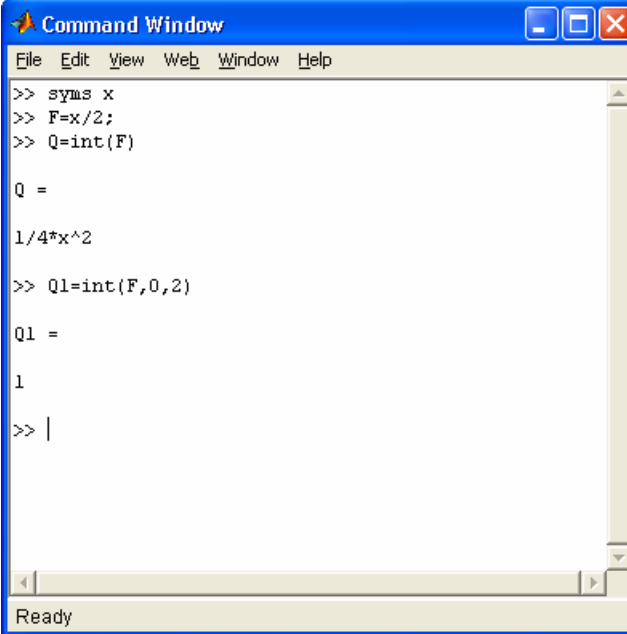
Notice that the special division operation “./” and product operation “.^3” are used because  $x$  is a vector.

**IMPORTANT:** The numerical integration “*qua*” can’t handle improper limits, therefore, very large positive and very large negative numbers must be used to approximate for  $-\infty$  and  $\infty$ , respectively.

### 3. Symbolic Integration using Matlab:

1. Use *syms* to specify that the variable is symbolic
2. Define the integrand function
3. Use *int* to integrate the function

Example:



```

Command Window
File Edit View Web Window Help
>> syms x
>> F=x/2;
>> Q=int(F)

Q =

1/4*x^2

>> Q1=int(F,0,2)

Q1 =

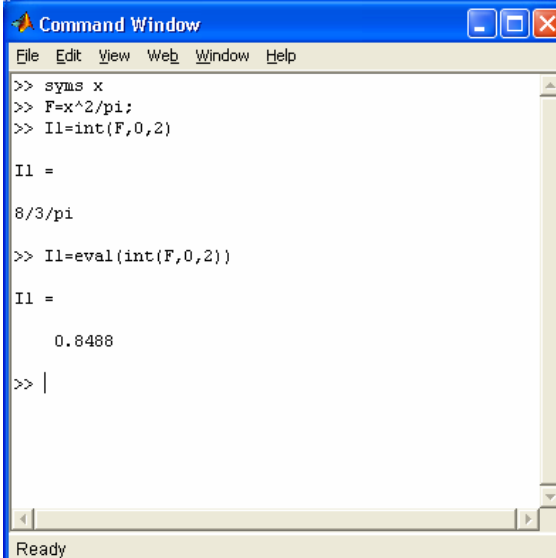
1

>> |
Ready

```

**IMPORTANT:**

1. *int* can use *-inf* and *inf* as integration limits (for  $-\infty$  and  $\infty$ )
2. You should use the function *eval* to express the function as a numeric value



```

Command Window
File Edit View Web Window Help
>> syms x
>> F=x^2/pi;
>> I1=int(F,0,2)

I1 =

8/3/pi

>> I1=eval(int(F,0,2))

I1 =

    0.8488

>> |
Ready

```

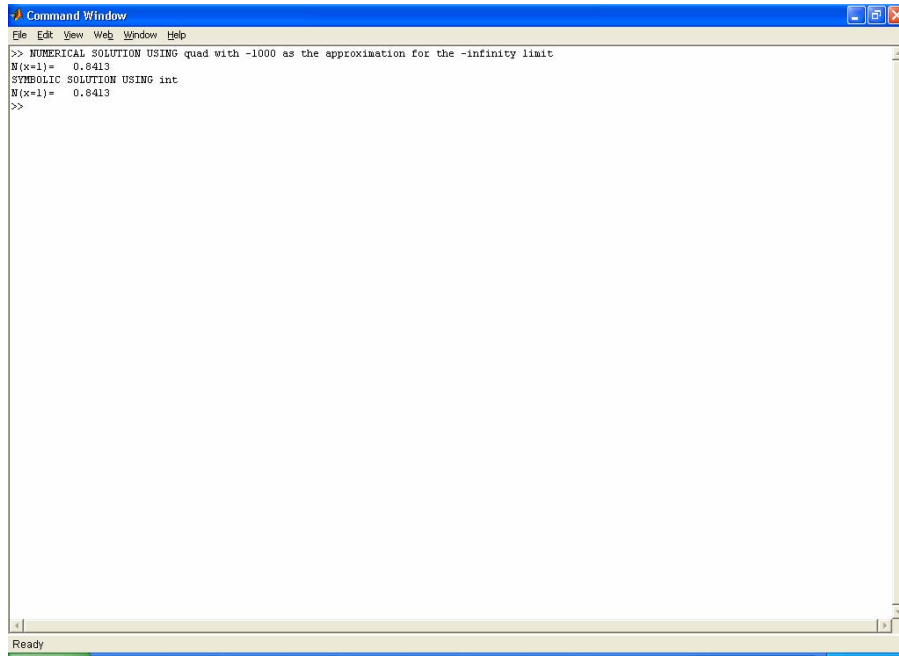
4. Develop a program in Matlab to solve for the integral given below (this integral is known as the normal distribution)

$$N(x) = \int_{-\infty}^1 \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}} dx$$

Use the numerical quadrature and the symbolic procedure.

**Program for the solution of the normal distribution function using Matlab (problem 4):**

```
% This program solves for the integral given in H-4 using quad numerical method and symbolic solution
% The integral is the normal distribution function N(x), evaluated at x=1: N(1)
% The output of this program is N(x=1)
% The integrand is called: F=exp(-x^2/2)/(2*pi)^0.5
% The integral limits are -infinity and 1. The infinity limit is abbreviated in Matlab as
% -inf or inf depending on the sign.
% The infinity limit only works in symbolic solutions (int)
% To represent the infinity limit in numerical solutions we need to choose a very large number
% Developed by GGB on 10/25/02. Last modified by GGB on 10/25/02.
%-----
clc
%-----
% NUMERICAL SOLUTION using quad
%-----
% The first step is to define the integrand: F
% Reminder: x is a vector (because it contains all the small fragments in which the integrand
% has been divided for its numerical integration (as explained in class)
% The variable "pi" represents the constant pi
% sqrt is used to calculate the square root
%-----
F=inline('exp((-x.^2)/2)/sqrt(2*pi)'); % Notice that the product is defined by .^2
%-----
% Solution of the integral
% the limits are: -1000 and 1. -1000 has been chosen as a very large negative number
%-----
Q=quad(F,-1000,1);
%-----
% Printing the numerical solution
%-----
fprintf(1,'NUMERICAL SOLUTION USING quad with -1000 as the approximation for the -infinity limit\n');
fprintf(1,'N(x=1)= %8.4f \n',Q);
%-----
% SYMBOLIC SOLUTION using int
%-----
syms x % definition of the x variable as a symbol
F=exp((-x^2)/2)/sqrt(2*pi); % definition of the integrand
Q1=int(F,-inf,1); % integration using int, notice that -inf can be used in this case
%-----
% Printing the symbolic solution
%-----
fprintf(1,'SYMBOLIC SOLUTION USING int \n');
fprintf(1,'N(x=1)= %8.4f \n',eval(Q1)); % in order to print the solution you need to evaluate
% the solution using the function "eval" as shown
```



```
Command Window
File Edit View Web Window Help
>> NUMERICAL SOLUTION USING quad with -1000 as the approximation for the -infinity limit
N(x=1)= 0.8413
SYMBOLIC SOLUTION USING int
N(x=1)= 0.8413
>>
```

Ready