



David S. Gilliam
Department of Mathematics
Texas Tech University
Lubbock, TX 79409

806 742-2566
gilliam@texas.math.ttu.edu
http://texas.math.ttu.edu/~gilliam

Mathematics 4330/5344 – # 4

Examples and Programming

1 Programming in Matlab

1. In Matlab there are 2 types of files: script-files and function files: typically a script-file is just a sequence of commands (a program) while a function file builds a function. script-files can call function files and execute them.
2. Lets do an example. You will need to open a text editor and type in the file

```
% this is a test m-file called test1.m
h=input('mesh size h = ');
x=(0:h:1);
lx=length(x);
y=x.^2;
int=(h/2)*(y(1)+2*sum(y(2:(lx-1)))+y(lx) )
% trap rule for integral with exact value 1/3
```

After typing in the file into the editor, use the *save as* command to save the file as `test1.m` and go back to the Matlab window. Then type the name of the file, in this case `test1` and hit return. You will be prompted for a mesh size `h` . Enter a value, try several values like `.25` , `.1` , `.05` , `.01` and `.001` . After entering a value for `h` hit return or enter .

3. Next we give an example of a program that uses Fourier series to approximate a function. We have built the function in a function file with two choices for the function. This program demonstrates how a Fourier sine series can be used to approximate a function on the interval $(0,1)$. First we will build a file containing the desired function to approximate. Go to the editor and write a file named `fn.m` containing the lines:

```
function y=fn(x)
global prob
if prob==1
    y=(1-(1-2*x).^2);
elseif prob==2
    y=(1-2*x).^2;
end
```

Save the file (as `fn.m`) and now build an m-file to carry out the approximation. The following should be typed into an m-file and saved as `four_sin.m`.

```
% four_sin.m
% approximate a function as a sum of sines
%--- clear the workspace
clear
clear global
%--- Declare prob as a global variable
global prob
%--- input a problem number
prob=input(' pick a problem (1 or 2) prob = ')
%---- Input the number of terms in the Fourier series
N=input('number of terms in Fourier series, N = ')
%--- Obtain a partition for trap quadrature
t=linspace(0,1,200);
%---- determine the mesh size
h=t(2)-t(1);
%----- trap quadrature
%----- note that since sin(0)=sin(k*pi)=0
%----- there are no terms for the end points
for n=1:N
    a(n)= 2*h*fn(t)*sin(n*pi*t)';
end
%----- Build a partition for plotting
delta=.05;
x=0:delta:1;
```

```

%---- compute matrix of sin values
p=sin((1:N)'*x*pi);
%--- compute Fourier series approximation
fn_app=a*p;
%----- Compute exact function values
y=fn(x);
%----- compute the L2 norm of difference
err=norm(y-fn_app)*sqrt(delta)
%plot the results
plot(x,y,x,fn_app)

```

- (a) Save the file as `four_sin.m`, return to Matlab and type `four_sin`.
 - (b) Try a few different cases for *prob equal 1 and 2* .
 - (c) Note that the sine series does not approximate very well a function that is not zero at the ends. Try $N=10, 50, 100, 200$.
 - (d) Since Matlab keeps track of all variables in the workspace, it is usually a good idea to begin a Matlab program with statements that delete all the current variables so there won't be any confusion. `clear` and `clear global` delete the variables and global variables in the workspace.
 - (e) Note that Matlab allows the use of global variables. Their use should be kept to a minimum and always give special names to such variables since they effect all parts of Matlab and will cause considerable trouble if not used properly.
4. This example which is taken from the book "Atlast" by S. Leon, E. Herman, and R. Faulkenberry, demonstrates how Matlab can be used to find formulas for the sum of the k th powers of the first n integers. It is known, but not proved here, that

$$\sum_{j=1}^n j^k = a_{k+2}n^{k+1} + a_{k+1}n^k + \cdots + a_2n + a_1,$$

for some numbers $\{a_j\}_{j=1}^{k+2}$. The main purpose of this discussion is to reinforce you Matlab skills by way of an example which allows us to find these numbers using Matlab. We will also get our first glance at the symbolic toolbox.

There are many toolboxes for Matlab. One toolbox is the Symbolic toolbox which allows us to carry out symbolic mathematics using a special version of the Maple kernel. We will talk about various Symbolic toolbox features later. The following to statements use Maple to find a formula for the sum of the squares of the first n intergers.

```

symsum('j^2','j',1,'n')

```

```
maple('sum', 'j^2', 'j=1..n')
```

produces the output

```
ans =  
1/3*(n+1)^3-1/2*(n+1)^2+1/6*n+1/6
```

We can bring this into a nicer form with

```
expand ans
```

which yields

```
ans =  
1/3*n^3+1/2*n^2+1/6*n
```

Practice using the symbolic toolbox commands above to find formulas for $k = 2, 3, 4, 5, 6$. You can sum to a definite integer n by simply replacing the n by an integer, for example,

```
symsum('j', 'j', 1, '5')
```

should give the sum of the first five integers. Probably everyone remembers that

$$\sum_{j=1}^n j = \frac{n(n+1)}{2}.$$

To formulate the problem in Matlab let us turn the problem of finding the unknown polynomial coefficients a_j into a problem in linear algebra. In this development, we only consider the case $k = 2$. We seek numbers $\{a_j\}_{j=1}^4$ such that

$$\sum_{j=1}^n j^2 = a_4 n^3 + a_3 n^2 + a_2 n + a_1.$$

Substituting $n = 0, 1, 2, 3$ we get the system of equations

$$\begin{aligned} 0a_4 + 0a_3 + 0a_2 + 1a_1 &= 0 \\ 1a_4 + a_3 + a_2 + a_1 &= 1 \\ 8a_4 + 4a_3 + a_2 + 2a_1 &= 5 \\ 27a_4 + 9a_3 + 3a_2 + a_1 &= 14 \end{aligned}$$

which can be solved using matlab. To do this we build the matrices

$$A = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \\ 8 & 4 & 2 & 1 \\ 27 & 9 & 3 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 0 \\ \sum_{j=1}^1 j^2 \\ 1 \\ \sum_{j=1}^2 j^2 \\ 2 \\ \sum_{j=1}^3 j^2 \\ 3 \\ \sum_{j=1}^4 j^2 \end{bmatrix}.$$

Then we have $a = A \setminus b$ gives the coefficients of the polynomial. Unfortunately, the answer will be given in floating point notation even though we know that the answers should be nice rational numbers. Matlab will do its best to convert floating point to rational format using the *rats* command.

The following Matlab code solves this problem for any k that you input. This is your first example of a Matlab script or m-file. You should copy the code and then paste it into the editor. Then save the file as `sum_form.m` – note that all matlab m-files must end with “.m”.

```
% Matlab script  sum_form.m
% Find a formula for the sum of the kth powers
% of the first n integers
% n      k      (k+1)      k
% sum i  = a(k+2) n  + a(k+1) n  + ... + a(2) n  + a(1)
% i=1
clear
disp('input a value of k ')
disp('      ')
k=input(' k = ');
I= (0:(k+1))';
% build the right hand side b
b=cumsum(I.^k);
% build the coefficient matrix A
for j=1:(k+1)
A1(:,j)=I.^j;
end
A2=fliplr(A1);
A=[A2 ones((k+2),1)];
% solve the system
```

```
a2=A\b;
% change to a row vector and convert to rational
a=rats(a2');
```

To check the work let us evaluate the sum. First use the Maple toolbox

```
symsum('j^2','j',1,'5')
```

Now lets use our formula. Remember a is a vector which, in particular means that its elements are the coefficients of a polynomial.

To see this consider the following example

```
p=[1 2 1];
x=-4:.05:2;
y=polyval(p,x);
plot(x,y)
grid
```

The vector p contains the coefficients $p=[1 \ 2 \ 1]$, in decreasing order, of the polynomial $x^2 + 2x + 1$. The command *polyval* is used to evaluate a polynomial. To evaluate the ploynomial a at $n = 5$, we type

```
symsum('j^2','j',1,'5')
polyval(a,5)
```

ASSIGNMENT 4 – Math 4330 and 5344

1. Trapezoid rule question: Use the trapezoid rule to approximate the integral of $f(x) = (x + 1) * \cos(x) * \exp(x)$ on $(0, 1)$. You must first build a function file, say `fn41.m` for assignment 4, problem 1 function. Use several values of N until you seem to get about 3 places of accuracy. Then compare you result with the symbolic toolbox command:

```
eval(int('(x+1)*cos(x)*exp(x)',0,1))
```

2. Alter the code for Fourier sine series to do a Fourier cosine series. Apply it to the functions in *fn.m*. What do you conclude about the approximations of these functions using a cosine expansion? Note for this exercise you will need to include endpoint terms in the trapezoid rule.
3. Exercises on the summation formulas:

- (a) Look for help on the command *vander* and rewrite the script `sum_form.m` using *vander* to build A .
 - (b) Check the code for $k = 3, 4, 5$ and $n = 5, 10, 20$.
4. Write a program to carry out Euclids algorithm for computing the greatest common divisor of two numbers a and b . The algorithm is:
- (a) input two numbers a and b
 - (b) compute the remainder of a/b
 - (c) replace a by b
 - (d) replace b by the remainder computed in the second step
 - (e) repeat the second through fourth steps until b is zero
 - (f) the gcd is the final value of a

For this problem you should see help on the *rem* command.

Math 5344 only

1. Write a program to compute the binomial coefficients $C(n, r)$. Do this by building a matrix P which is of size $(n + 1)$ by $(n + 1)$ for a given n so that p_{ij} satisfies

```
p_{i,1}=p_{1,j}=1,
for i+j <= (n+2), p_{i,j}=p_{i,(j-1)}+p_{(i-1),j},
for i+j > (n+1), p_{i,j}=0
```

You can use loops but it is much easier to use the built-in commands *diag*, *pascal* and *rot90*.

```
% pascal(k) builds the pascal matrix of size k by k
% rot90(A) rotates the matrix A by 90 degrees
% diag(A) builds a column vector from the diagonal of A
```

Extra Credit Problems

1. An exercise intended to convince you not to play the lottery. Write a program that lets you input six integers from 1 to 50 and also an integer n corresponding to playing the lottery n times. Next the program should generate six distinct integers from 1 to 50 and compare with your six numbers to see if you have matched 3, 4, 5 or 6 numbers. The program should do this process n times keeping track of how

many times you had a winner and what type of winner it was, i.e., a 3, 4, 5, or 6 number winner.

You might want to write it as a "function" m-file with variable n (the number of times to run the lottery) and a vector v which contains your lottery numbers (e.g. [1 23 32 27 19 48]).

(Hint: You might find the following useful:

```
test=zeros(1,5);
while all(test~=0)~=1
% this while statement helps to obtain a unique vector
lottst=floor(50*rand(1,6))+1
% After you run the program a few times put a
% semicolon at the end to suppress printing lottst
lot1=sort(lottst);
test=lot1(2:6)-lot1(1:5);
end

disp('the lottery numbers are')
disp(lottst);
```

References

- [1] *The Matlab Primer*, Kermit Sigmon
- [2] *Introduction to scientific computing: a matrix vector approach using Matlab*, Printice Hall, 1997, Charles Van Loan
- [3] *Mastering Matlab*, Printice Hall, 1996, Duane Hanselman and Bruce Littlefield
- [4] *Advanced Mathematics and Mechanics Applications Using Matlab*, CRC Press, 1994, Howard B. Wilson and Louis H. Turcotte
- [5] *Engineering Problem Solving with Matlab*, Printice Hall, 1993, D.M Etter
- [6] *Solving Problems in Scientific Computing Using Maple and Matlab*, Walter Gander and Jiri Hrebicek
- [7] *Computer Exercises for Linear Algebra*, Printice Hall, 1996, Steven Leon, Eugene Herman, Richard Faulkenberry.
- [8] *Contemporary Linear Systems using Matlab*, PWS Publishing Co., 1994, Robert D. Strum, Donald E. Kirk