MA3990 Matlab refresher, week 05, 2018/9

Contents

This handout, some files and further information.
`http://people.brunel.ac.uk/~icstmkw/ma3990/index.html`

Getting the software

- Matlab is not free but there is a **student version**.
- There are free "clones" such as **octave** which has a similar functionality. The following is a URL for octave.

  `https://www.gnu.org/software/octave/`

  At the time of this week 5 lecture `octave-4.4.1` is the latest version.

Access to the software at Brunel

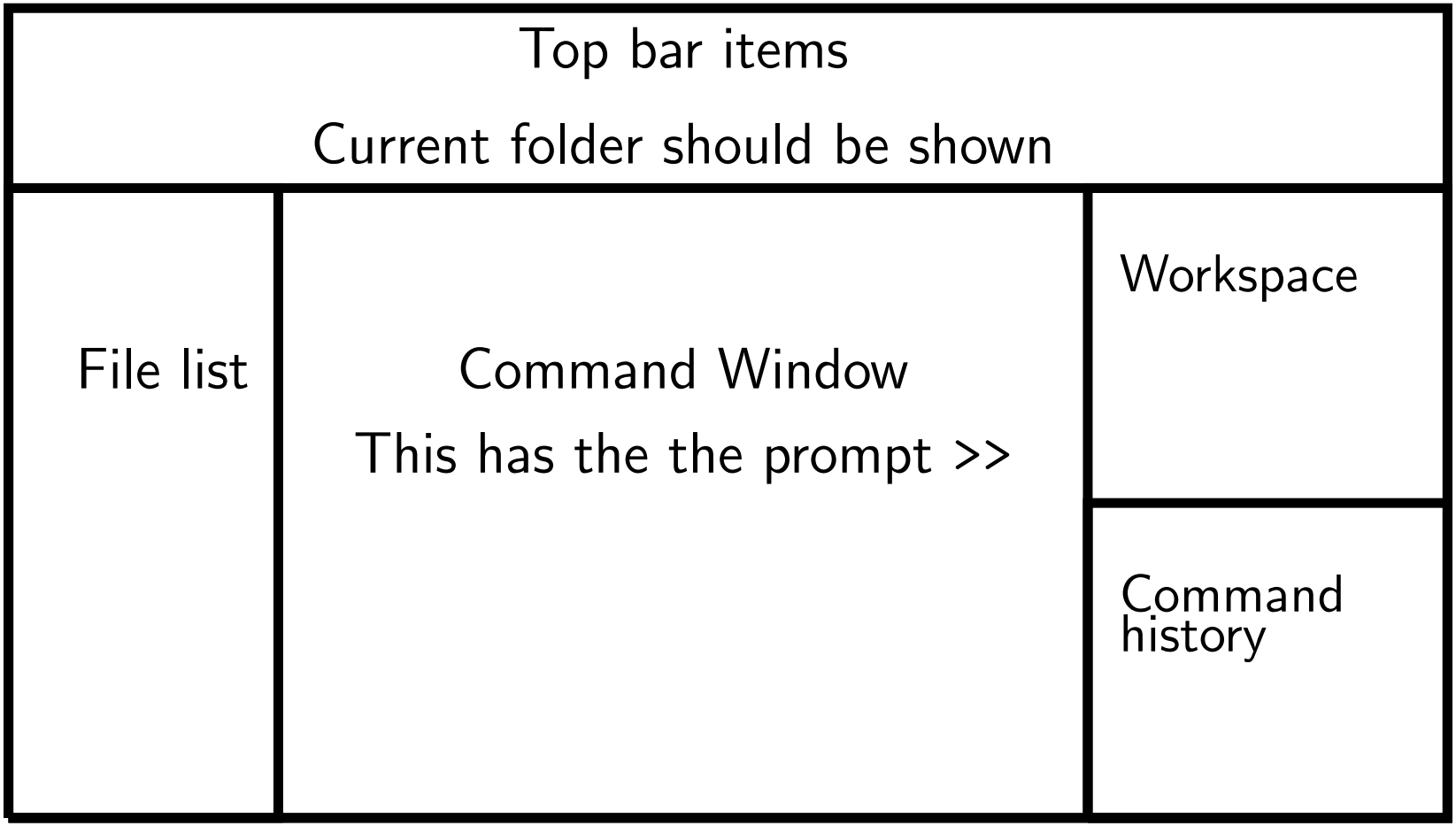
All the labs that I know about now run windows 10.

- The latest version of Matlab is R2018b.
- In labs such as WLFB 013 and WLFB 106 we now have Matlab R2018a installed.
- Older versions may be in other labs.
- In locations without Matlab being available from the Windows search the following may still work for an older version of Matlab if you use a command window.

  ```
  net use L: \\academic.windsor\depapps
  L:\cc\matlab2010b32\bin\matlab.exe
  ```

The Matlab window layout

With the version in the labs the layout may be as follows.



Most of the time you are creating `m-files` with an `editor` and running things in the command window and observing the results.

Creating vectors and matrices with [and]

In mathematical notation consider the following.

$$A = \begin{pmatrix} 2 & -3 & 1 \\ 3 & 4 & -5 \\ 5 & -1 & -3 \end{pmatrix}, \quad \underline{b} = \begin{pmatrix} 0 \\ 2 \\ 1 \end{pmatrix}, \quad C = \begin{pmatrix} 2 & -3 & 1 & 0 \\ 3 & 4 & -5 & 2 \\ 5 & -1 & -3 & 1 \end{pmatrix},$$

In Matlab these can be created with the following statements.

```
A = [2 -3 1; 3 4 -5; 5 -1 -3]
b = [0; 2; 1]
C = [A, b]
```

We can also create `b` by putting

```
b = [0  2  1]'
```

Creating vectors of equally spaced values

$n+1$ equally spaced points in $[a, b]$ are given by

$$x_i = a + ih, \quad i = 0, 1, \ldots, n, \quad h = \frac{b-a}{n}.$$

In Matlab a row vector can be created by putting the following.

```
h=(b-a)/n;
x=a:h:b;
```

Alternatively you can put the following.

```
x=linspace(a, b, n+1);
```

To create a column we can always reshape as in the following.

```
x=x(:);
```

In Matlab the entries are `x(1)`, `x(2)`, ..., `x(n+1)`.

Basic 2D plots

Let

$$f(x) = \sin(x) + 0.3\exp(-0.1x^2)\sin(10x).$$

To plot this for $x \in [0, 2\pi]$ we can do the following.

```
x=linspace(0, 2*pi, 500);
y=sin(x)+0.3*exp(-0.1*x.^2).*sin(10*x);
figure(2)
plot(x, y, 'LineWidth', 3)
```

The `figure` command ensures that the window with the plot is not hidden.

Remember that the standard functions operate entry-wise on vectors and matrices and also note the use of `.*`

Fine tuning a plot and saving

```
% to adjust the numbers on the axis
set(gca, 'FontSize', 12)

% labels on the axis and a title
xlabel('x-axis')
ylabel('y-axis')
title('Plot created for revision talk', 'FontSize', 16,...
      'FontName', 'Times New Roman')

% saving and getting a colour eps version
print('my_plot.eps', '-depsc')
```

Note that Word may not deal with this format as well as other formats. See next to create a pdf or appropriately copy and paste from the Matlab figure window.

Do not use formats such as `jpg` or `png` for this type of graphics.

Getting a pdf version of the right size

Matlab solutions

I would not use the `-dpdf` option without modification. I have a function `print2pdf.m` which modifies by trimming some of the surrounding white space. An example of using the function is as follows.

```
print2pdf('my_plot_trimmed.pdf')
```

LaTeX software solutions

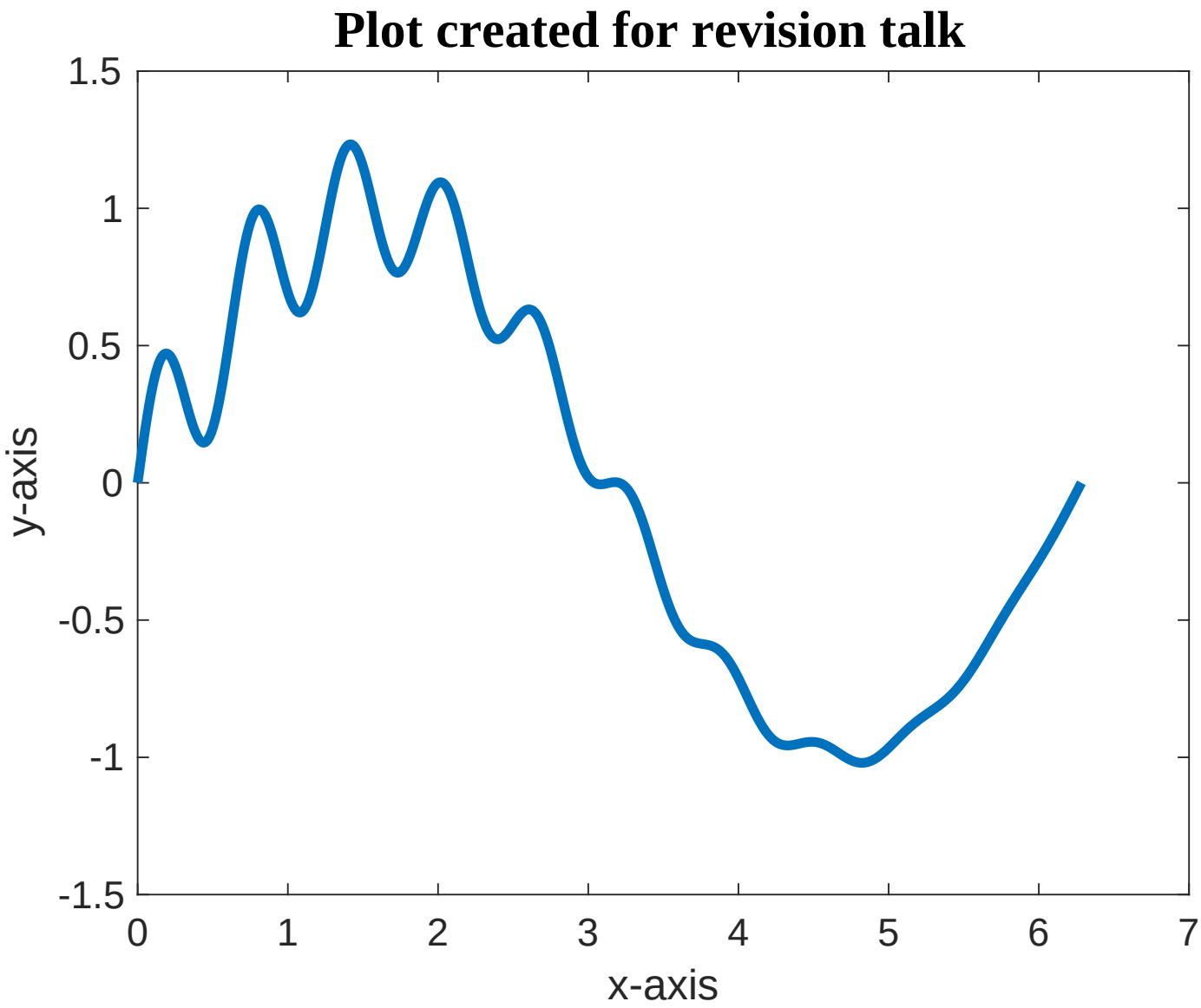
If you have LaTeX and related software then in a command window which has LaTeX in the path you can create `my_plot.pdf` by typing

```
epstopdf my_plot.eps
```

If you already have a pdf but want to remove all the surrounding white space then in a command window you can type

```
pdfcrop my_plot.pdf
```

The plot created

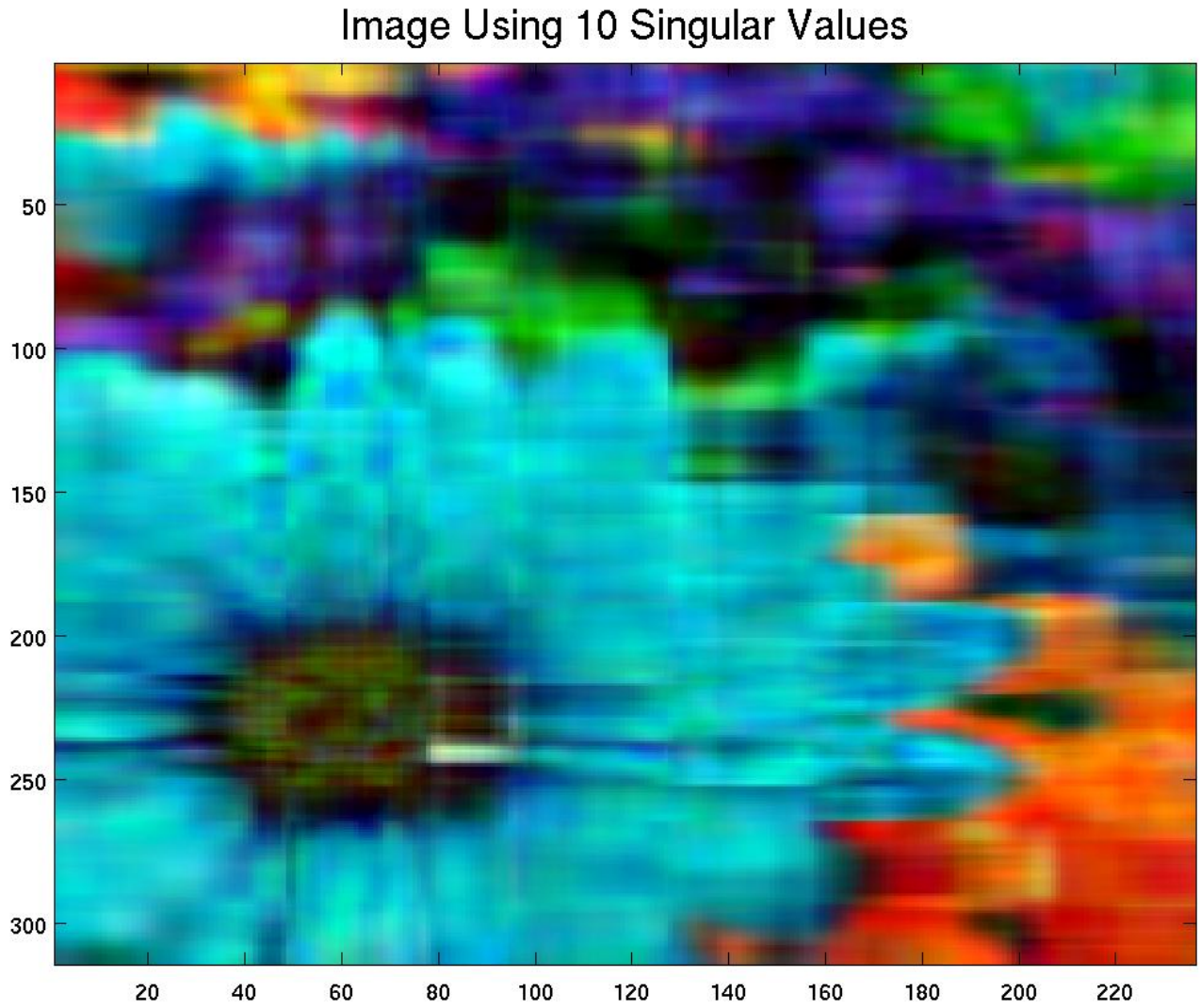


When are `jpg`, `png` etc. appropriate

For images, e.g. photos, these are good formats to use.

One application of a project in 2015/6 involved approximating matrices and the matrices corresponded to a colour image.

The image using only 10 singular values



The functions `zeros`, `ones` and `eye`

```
Z = zeros(3, 5)
O = ones(3)
I4 = eye(4, 4)
```

This generates the following output.

```
Z =
     0     0     0     0     0
     0     0     0     0     0
     0     0     0     0     0
O =
     1     1     1
     1     1     1
     1     1     1
I4 =
     1     0     0     0
     0     1     0     0
     0     0     1     0
     0     0     0     1
```

Loops and decisions

The `for`–loop syntax

```
for variable_name=list_of_values
  Instructions to do for each value in the list.
  The instructions typically use variable_name.
end
```

The `if-else` construction

```
if  logical_condition
  Statements to do if the condition is true.
else
  Statements to do if the condition is false.
end
```

You can leave a loop early with a `break` statement and you can jump to the next case with a `continue` statement.

Creating your own functions

You may have a problem leading to an algorithm or pseudo code for part of it or all of it. An appropriate way to code it may involve creating a function.

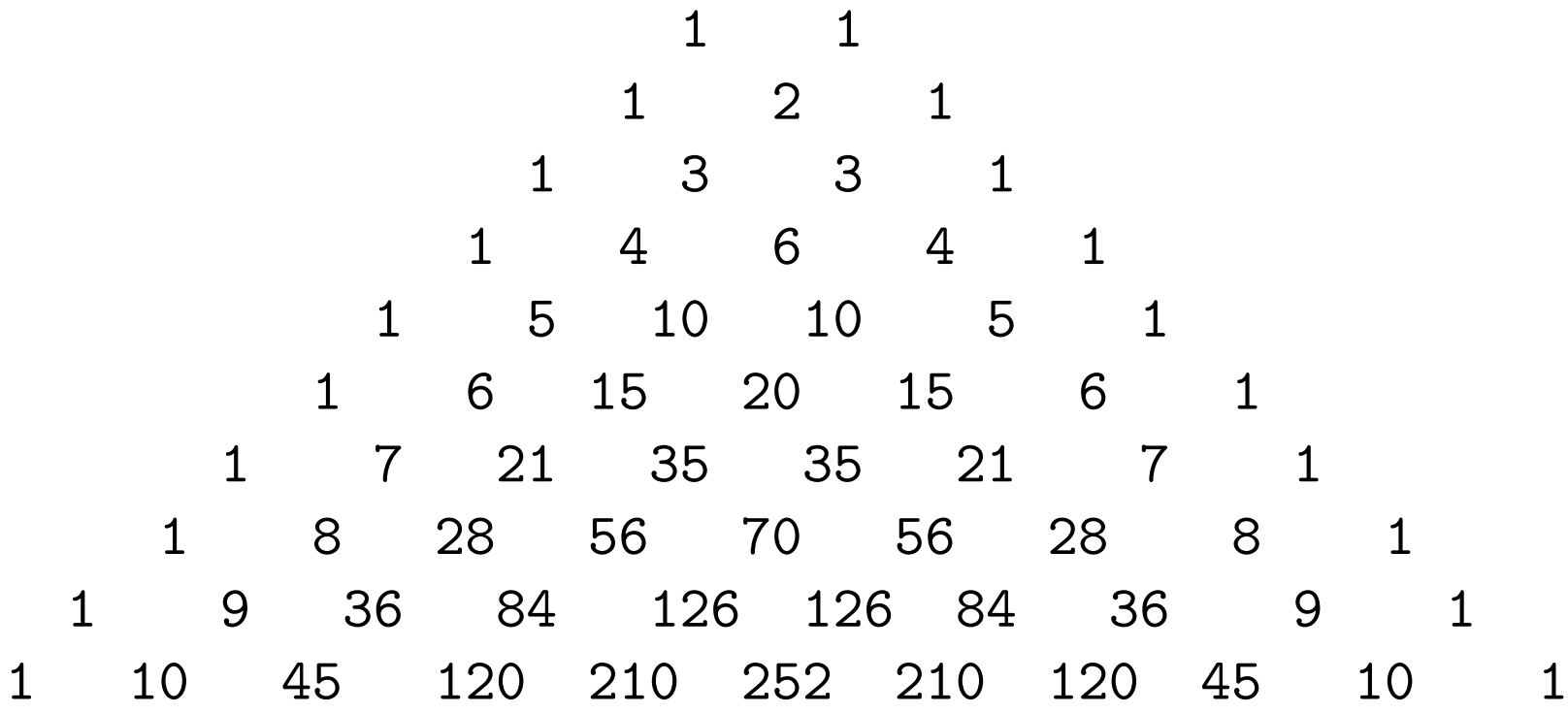
The syntax of the function statement is as follows.

`function [`*`output_arguments`*`]=function_name(`*`input_arguments`*`)`

This should be the first line of a file called `function_name.m`.

Example: Creating Pascal's triangle

Suppose you want to nicely display Pascal's triangle.



Splitting the task into parts

1. A function `pascal_tri` to create the entries.
2. A function `centre` to centre a number in a given width.
3. A function `display_tri` to deal with displaying as a triangle.
4. A script file which uses the functions. The script file is the file to run.

As well as using functions the example also shows how to write to a file.

The mathematics involved

The entries to create are the binomial coefficients with an entry on one row being the sum of two entries in the row above. The expressions are as follows.

$$\binom{k}{0} = \binom{k}{k} = 1,$$

$$\binom{k}{j} = \binom{k-1}{j} + \binom{k-1}{j-1}, \quad j = 1, \ldots, k-1.$$

In Matlab the index of an array starts at 1 and in the following we store the k, j entry in `T(k, j+1)`.

A function to create the entries

```
function T=pascals_tri(n)
%% T=pascals_tri(n)
%  the output T is a n-by-(n+1) matrix containing
%  entries in Pascal's triangle

T=zeros(n, n+1);
T(1, 1:2)=[1, 1];
for k=2:n
  T(k, [1, k+1])=1;
  T(k, 2:k)=T(k-1, 1:k-1)+T(k-1, 2:k);
end
```

A function to centre a number in a given width

In Matlab there is a function called `int2str` to create a character array with the digits corresponding to an integer. There is also a function `blanks` to create blank characters.

```
function c=centre(v, w)
%% function c=centre(v, w)
%  input:  integer v and the width w
%          of the character array c to create
%  output: character array c of length w with v centered

s=int2str(v);
m=length(s);
l=floor((w-m)/2);
c=[blanks(l), s, blanks(w-m-l)];
```

A function to display Pascal's triangle

```
function display_tri(T, fp)

% get the dimensions and set the width to use
[n, m]=size(T);
big=max(T(:));
cbig=int2str(big);
w=2+length(cbig);

% loop through the rows with and an appropriate indent
for k=1:n
  indent=floor((n-k)*w/2);
  fprintf(fp, '%s', blanks(indent));
  for j=1:k+1
    fprintf(fp, '%s', center(T(k, j), w) );
  end
  fprintf(fp, '\n');
end
```

A controlling script to run

```
% get the triangle

n=10;
T=pascals_tri(n);

% open the file to write to and display to the file

fp=fopen('output_pas_tri.txt', 'w');
display_tri(T, fp);
fclose(fp);
```

Output to the command window

To modify the previous script so that output is to the command window just requires commenting out the open and close commands and setting `fp` to 1.

```
% get the triangle

n=10;
T=pascals_tri(n);

% open the file to write to and display to the file

% fp=fopen('output_pas_tri.txt', 'w');
fp=1;
display_tri(T, fp);
% fclose(fp);
```

Final comments – planning, layout, presentation

1. Plan your task. Do you have a pseudo-code description or an algorithm? Can you do all or some of the sub-tasks? Allow enough time.

   > "A carelessly planned project takes three times longer to complete than expected; a carefully planned project will take only twice as long"

   This is Golub's Second 2nd law of Computerdom.
2. Take care with the program layout so that it is easy to read. For example, avoid long lines, indent blocks of code in loops, add helpful comments etc. The editor menu item `text` has the option `smartindent`.
3. Do not use screen images when presenting code or graphics. The quality may be low.