Octave / Matlab Tutorial

Introduction to Octave and Matlab

- Both:
  - allows matrix manipulations
  - plotting of functions
  - implementation of algorithms
  - Share common syntax https://en.wikipedia.org/wiki/GNU_Octave#Syntax_compatibility
- Octave : free software (https://gnu.org/software/octave/)
- Matlab : proprietary

2 modes of using Octave

- Interactive mode
- Scripting mode

- Case sensitive

Octave as Calculator

- 2+5
- 7.5-16.8
- 8.5e-7*1.0e+9
- (5+2i)/(5+3i)
- 3^4

Built-in Functions and Constants

- sin(-2)
- cos(3.14)
- tan(22/7)
- sqrt(4)
- abs(-5)

- exp(9)
- log(2.7183)
- log10(10)
- round(12.4)
- pi, e

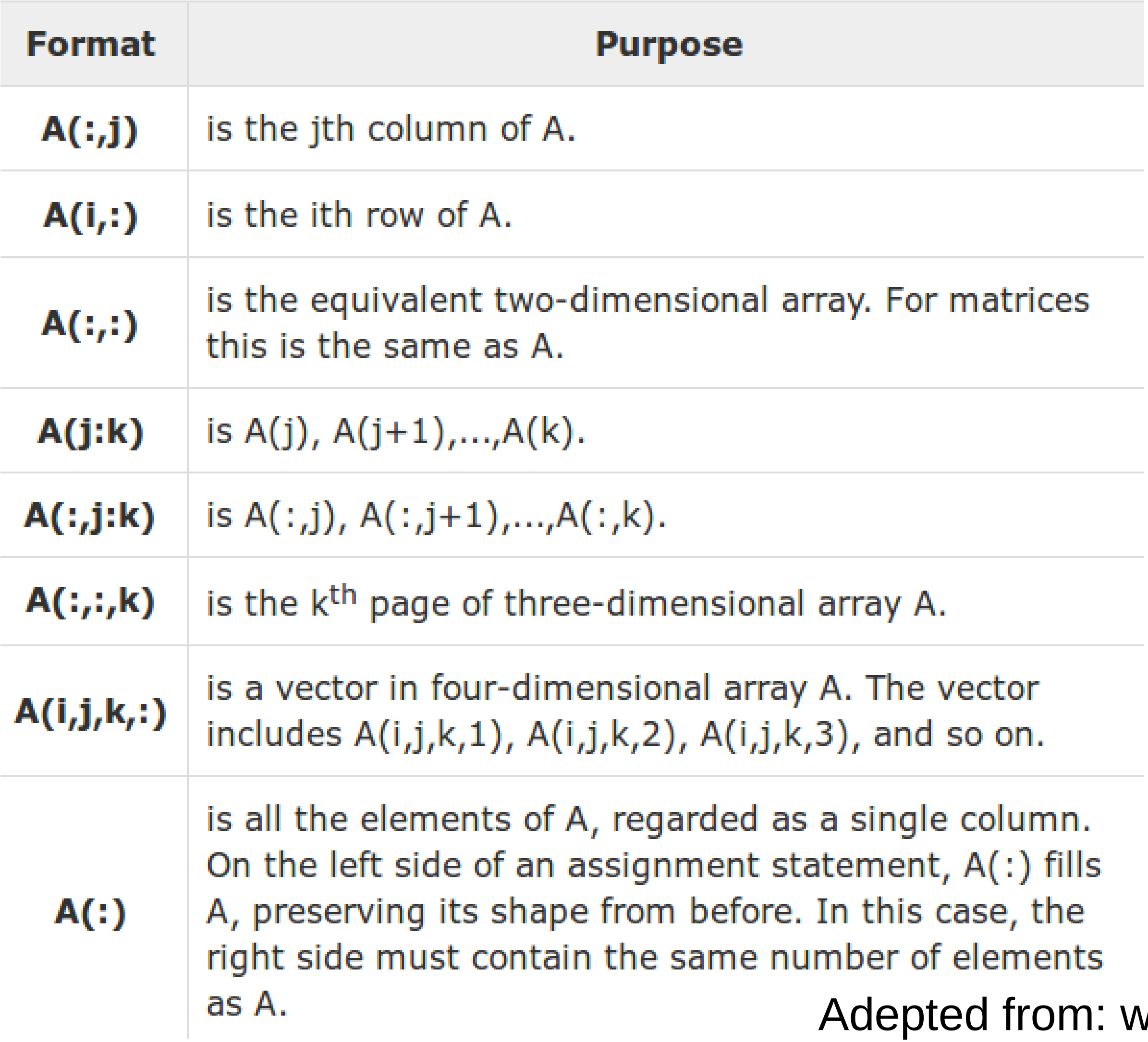
Creating vector and matrices

- X = -2.46 % create a variable
- X = [1 2]; % a vector
- X = [1, 2; 3, 4] % 2X2 matrix
- X = [2 3 , 4 5; 6 7, 8 9; -1 -2, -3 -4] % a matrix

More about matrices

- X % retrieve the whole matrix
- X(2,3) % row 2, col 3
- X(:,2) % all values in col 2
- X(3,:) % all values in row 3

Colon notation



Format	Purpose
A(:,j)	is the jth column of A.
A(i,:)	is the ith row of A.
A(:,:)	is the equivalent two-dimensional array. For matrices this is the same as A.
A(j:k)	is A(j), A(j+1),...,A(k).
A(:,j:k)	is A(:,j), A(:,j+1),...,A(:,k).
A(:,:,k)	is the k^{th} page of three-dimensional array A.
A(i,j,k,:)	is a vector in four-dimensional array A. The vector includes A(i,j,k,1), A(i,j,k,2), A(i,j,k,3), and so on.
A(:)	is all the elements of A, regarded as a single column. On the left side of an assignment statement, A(:) fills A, preserving its shape from before. In this case, the right side must contain the same number of elements as A.

Adepted from: www.tutorialspoint.com

Colon notation

- X = 1:5 % X=[1 2 3 4 5]
- X = 1 : 0.5 : 5 % X = [1.0 1.5 2 … 5]

- Multi dimension matrices
  - X = [1 2 3; 4 5 6]
  - X(:,:,2) = [-1 -2 -3; -4 -5 -6] % can you run this alone?

Operators and functions

- inv(X) % inverse of X
- det(X) % determinant of X
- X’ % transpose of X
- X + Y
- Z = X – Y;
- Z = X * Y; % matrix multiplication
- Z = X.*Y; % array multiplication
- X/Y % X * inv(Y)
- X./Y % array division

Matrix related functions

- zeros(3,4) % all entries are 0
- X = ones(4,3); % all entries are 1
- X = rand(3,5) % random
- eye(4) % guess?
- size(X) % return the dimension of X

Exercise

- Construct A, a 4X4 matrix
- Find B and C, the transpose and inverse of the matrix.
- What is the output of size(B)?
- Reset the value of a_{24} from A to -99.
- Construct matrix D (2X2) using a_{23},a_{24},a_{33},a_{34} from A.
- Construct E, a row vector such that

  E = [100, 100.1, 100.2, 100.3, …, 120]
- If there is a 4X4 matrix F such that $f_{ij} = a_{ij}^2$ for 1<= i <=4 and 1<= j <= 4.

  What probably is the operation that created F?

Script program

- Sometimes, multiple lines of code is required for us to solve a problem. For example:

Step 1: Create matrix A	A = [1 2 ; 3 4];
Step 2 : Find A×A	B = A*A;
Step 3 : Find the det of A	p = det(A);
Step 4 : Find the det of A×A	q = det(B);
Step 5 : Find the difference	difference = p-q

Script program

- If we want to run the program multiple times, we can create a script program to perform the similar task.
- You can use the built-in editor or any text editor (notepad, geany, notepad++, ...), type all the commands to perform the task in sequence.
- A line starts with a % is comment.

```
% a simple program written by little Snoppy
A = [1 2 ; 3 4];
B = A*A;
p = det(A);       % find determinant of A
q = det(B);       % find determinant of AXA
difference = q-p
```

Script program

- Save the file with your preferred name and extension name .m (for example, findDiff.m) in the current working directory.
- Now, every time we run findDiff (without .m) in the Octave command prompt, the program will execute and the difference will be computed.
- To find out our working directory, run command `pwd` in the Octave command prompt prints the path to the directory.

String and output

- To display an output, x:
  - `disp(x);`
  - `fprintf("my output is %d", x);`
- We also can assign strings to a variable, e.g:
  - `X = "this is my string";`
  - `fprintf("I wrote: %s .", X);`
- `\n` and `\t`

Decision (if… else)

- Octave has decision making statement known as if ... else selection. The form of the statement looks like this:

```
if (condition)
   then-body
elseif (condition)
   elseif-body
else
   else-body
endif
```

Example

```
x=5;
if (x > 1)
  fprintf ("x is greater than 1 \n");
else
  fprintf ("x is not greater than 1 \n");
endif
```

Example (nested if)

```
x=y=5;
if (x > 1)
 if (y>1)
   fprintf ("both x and y greater than 1 \n");
 else
   fprintf ("only x greater than 1 \n");
 endif
else
  fprintf ("x is not greater than 1 \n");
  fprintf ("y is not tested\n");
endif
```

Condition

- Also called loop
- Repetition control structure
- Allow code to repeat until criteria is fulfilled.
- 2 types:
  - `for loop`
  - `While loop`
- A loop may nested in another loop

FOR

- Syntax:

```
for index = values
    % put statements to repeat here
end
```

- Conditions of repetition:
  - **initval:endval** - increments the index variable from initval to endval by 1, and repeats execution of program statements until index is greater than endval.
  - **initval:step:endval** - increments index by the value step on each iteration, or decrements when step is negative.
  - **valArray** - creates a column vector index from subsequent columns of array valArray on each iteration

Adepted from: www.tutorialspoint.com

Examples (FOR)

```
a = [2 4 6 8];
for i=a
   disp(i);
end
```

```
for i=-2:2
   disp(i);
end
```

```
for i=-2:0.5:2
   disp(i);
end
```

while loop

- Repeat the code until a condition is not fulfill
- Check whether the condition is fulfilled at the beginning of the code.
- Syntax:

```
while condition
    % put statements to repeat here
end
```

Example (while)

```
i=1;
while (i<=10)
   disp(i);
   i=i+1;
end
```

AND and OR

and - &

```
i=1;
j=10;
while (i<=10 & j>=5)
   disp(i+j);
   i=i+1;
   j=j-2;
end
```

or - |

```
i=1;
j=10;
while (i<=10 | j>=5)
   disp(i+j);
   i=i+1;
   j=j-2;
end
```

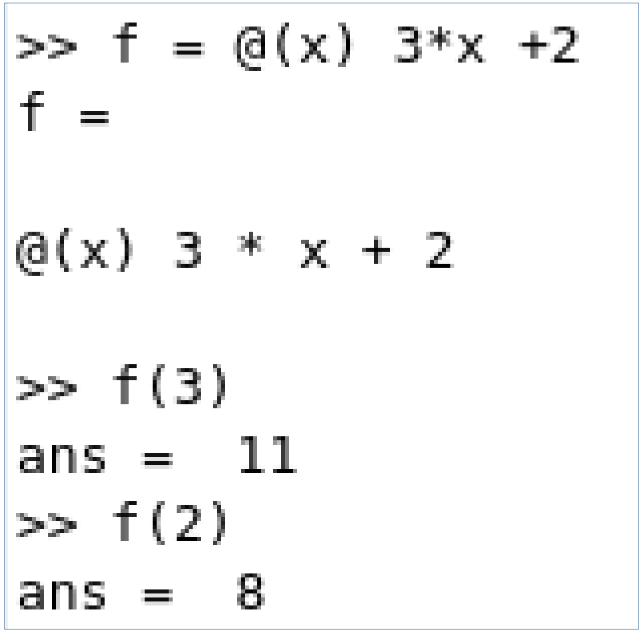
Exercise

- Create a script that find the sum of 2 matrices if $a_{11}>b_{11}$. Otherwise find their product.
- Create a script that repeat finding the square of a matrix until:
  - The first entry is greater than 100, or
  - The process repeated 10 times.
- Create a program that require nested loop.

Math function

- In math, f(x) = 3x+2
- In Octave :

  f = @(x) 3*x +2

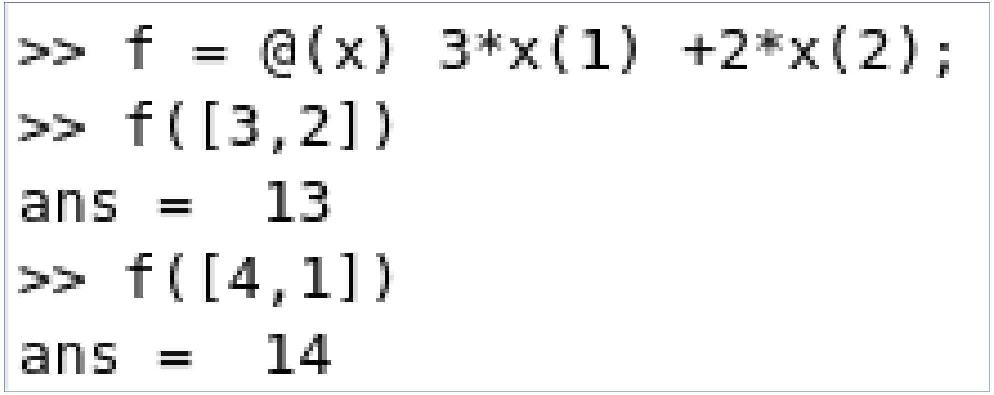


```
>> f = @(x) 3*x +2
f =

@(x) 3 * x + 2


>> f(3)
ans =  11
>> f(2)
ans =  8
```

Math function

- In math, $f(x) = 3x_1+2x_2$
- In Octave :

    f = @(x) 3*x(1) +2*x(2)



```
>> f = @(x) 3*x(1) +2*x(2);
>> f([3,2])
ans =  13
>> f([4,1])
ans =  14
```

Plotting

- To plot a contour:

```
% create linear space for x (-5 to 5) and y
xSpace = linspace(-5,5);
ySpace = linspace(-3,3);
% create a grid for the plot to take place
[X, Y] = meshgrid(xSpace, ySpace);
% the function Z, which the values are
% determined by X and Y
Z = X.*Y;
% plot the contour
contour(X,Y,Z);
```

Plotting

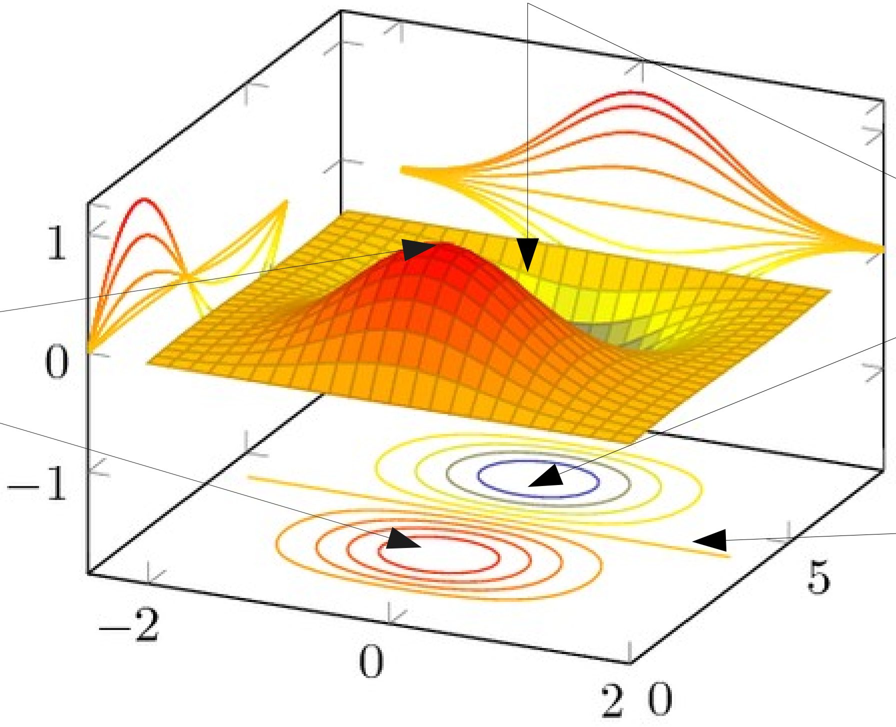
- To add lines to the contour:

```
% to continue plotting on the same fig
hold on;
% set the points
lineX = [-2 -0.5 1 -1.5 2 4];
lineY = [2  1.5  0 -1.5 1 -2];
% plot the line with colour ‘b’ – blue
p = plot(lineX, lineY, 'b');
% change the line width
set(p(1),'LineWidth',3);
```

Other plotting options

- `contour(X,Y,Z,n)` – Similar to `contour(X,Y,Z)`, but with additional parameter n to specify number of levels.
- `contourf(X,Y,Z)`- plot filled 2-D contour.
- `plot3(X,Y,Z)` – plot 3D graphs.
- `mesh(X,Y,Z)` – another 3D plotting function

Understanding contour plot



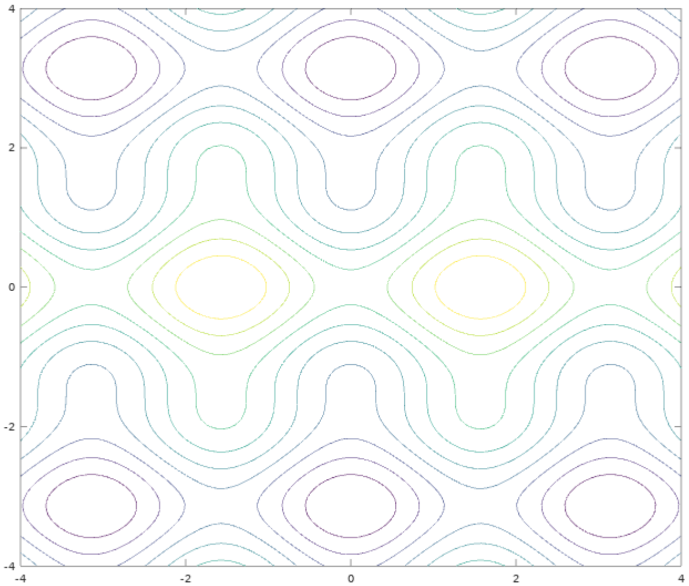
maximum

minimum

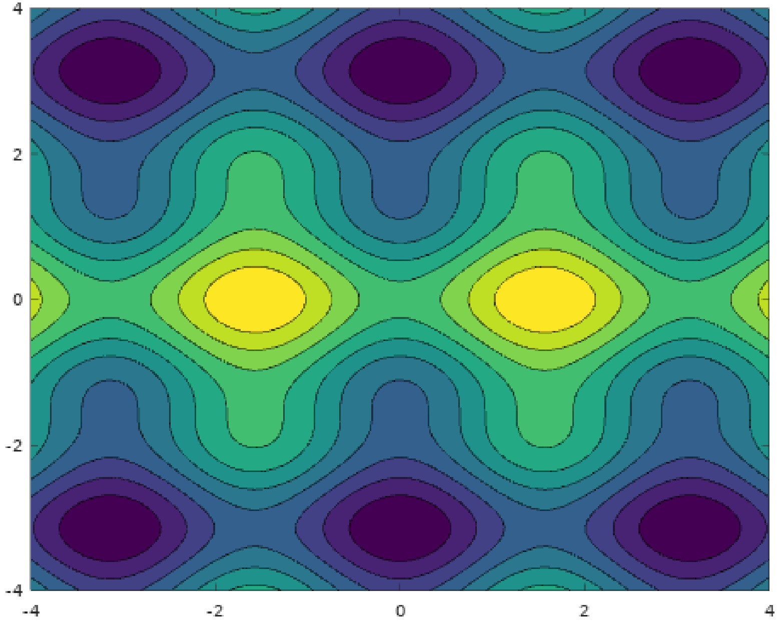
Contour plot

Courtesy of http://pgfplots.net/tikz/examples/contour-and-surface/

Understanding contour plot



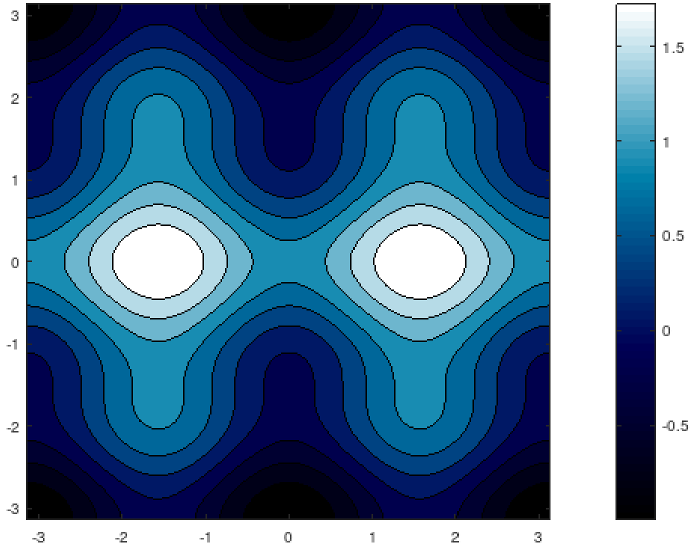
contour

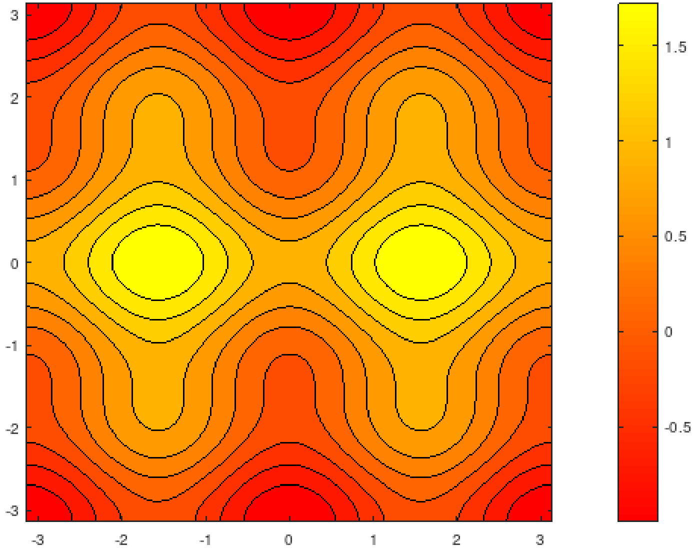


contourf

Colormap

- Execute `colorbar()` to show colour bar
- Various colour maps. To use : `colormap(name)`





Map	Description
viridis	default
jet	colormap traversing blue, cyan, green, yellow, red.
cubehelix	colormap traversing black, blue, green, red, white with increasing intensity.
hsv	cyclic colormap traversing Hue, Saturation, Value space.
rainbow	colormap traversing red, yellow, blue, green, violet.
—————-	————————————————————————————————————
hot	colormap traversing black, red, orange, yellow, white.
cool	colormap traversing cyan, purple, magenta.
spring	colormap traversing magenta to yellow.
summer	colormap traversing green to yellow.
autumn	colormap traversing red, orange, yellow.
winter	colormap traversing blue to green.
—————-	————————————————————————————————————
gray	colormap traversing black to white in shades of gray.
bone	colormap traversing black, gray-blue, white.
copper	colormap traversing black to light copper.
pink	colormap traversing black, gray-pink, white.
ocean	colormap traversing black, dark-blue, white.
—————-	————————————————————————————————————
colorcube	equally spaced colors in RGB color space.
flag	cyclic 4-color map of red, white, blue, black.
lines	cyclic colormap with colors from axes `"ColorOrder"` property.
prism	cyclic 6-color map of red, orange, yellow, green, blue, violet.
—————-	————————————————————————————————————
white	all white colormap (no colors).

User Define Functions

- User define functions are Octave scripts that require inputs upon the execution of the scripts.
- We can run user define functions either from the Octave command prompt, or in other functions or scripts.

User Define Functions

- A user define function with one output and multiple inputs is an Octave script starts with a line in the form:

```
    function output = name (input1, input2 , ...)
```

  and end with a line:

```
    endfunction
```

- output – a variable that hold the output value in the function. Once output variable is assigned with a value in the function, this value will be the output of this script.
- name – name of the function. The file that hold the script should also be saved with this name.
- input1, input2, ... - list of inputs that are going to be processed in the function.

Example

```
function difference = findDiff2 (A)
  B = A*A;
  p = det(A);
  q = det(B);
  difference = q-p;
endfunction
```

- To run the function in command prompt:

```
>> findDiff2([1 2; 3 4])
```

Import and export

- To save a matrix to a file:

```
save myfile.out mydata -ascii
```

- To load data from myfile.out to variable a

```
a = importdata(‘myfile.out’);
```

- To save a plotted graph to svg format:

```
print -dsvg mygraph.svg
```

Exercise (Q1)

- Write program `plotTrigo.m` that plot the contour of the following function:

$$f(x) = \sin(x_1^2) - \cos(x_2^2)$$

- Save the value of *f*(*x*) into a file “*`yourname`*`.txt`”.
- Save the plot as “`yourname_plot.svg`”.
- Submit all files to Spectrum.
- From the plot, can you identify a good location where we can start our search for a local minimum?

Exercise (Q2)

- Write an Octave program to find the minimum of the following function using Newton Method:

  $f(x) = 7x - \log(x)$ (natural logarithm)

  Start the search at x = 0.01

- Note that:

  $f'(x) = 7 - 1/x$ $f''(x) = 1/x^2$

Exercise (Q3)

$$f(\mathbf{x}) = 13.5x_1^2 + 128x_2^2 - 0.3\cos(3\pi x_1) - 0.4\cos(4\pi x_2) + 0.7$$

- Write Octave code to find the minimizer of the function using Newton method. Start your search at (5,5) with a fixed step length 0.02, and repeat the search for 1000 iterations.
- Plot a contour to verify your answer.