

SGF Tutorial 2

Numerical Differentiation

Differentiation is carried out by the method of finite differences. By definition, the 1st derivative of a quantity V is:

$$\frac{dV}{dx} = \frac{\Delta V}{\Delta x} \quad \text{when } \Delta x \text{ goes to zero}$$

Now let's say we set up V as an array, then the approximations of the first derivative of V at the point i :

Central difference:

$$\frac{dV}{dx} \approx \frac{V[i+1] - V[i-1]}{2\Delta x}$$

One-sided differences

$$\frac{dV}{dx} \approx \frac{V[i+1] - V[i]}{\Delta x}$$

$$\frac{dV}{dx} \approx \frac{V[i] - V[i-1]}{\Delta x}$$

Sample Program: To find the electric field in 1-D, by finding the gradient of the potential

$$E = -\frac{dV}{dx}$$

```
// Filename: gradient.sg
const PI = 3.14159265358979323846; // pi
const DIM = 101; // number of mesh points (array length)
const DX = 2.0*PI/(DIM-1); // mesh spacing (cm) (dx in the differential)

var x[DIM], V[DIM], E[DIM];
equ E[i=0] -> E[i] = -(V[i+1]-V[i]) / (1.0*DX); //one-sided difference
equ E[i=1..DIM-2] -> E[i] = -(V[i+1]-V[i-1]) / (2.0*DX); //central difference
equ E[i=DIM-1] -> E[i] = -(V[i] - V[i-1]) / (1.0*DX); //one-sided difference

begin main
  assign x[i=0..DIM-1] = i*DX;
  assign V[i=0..DIM-1] = cos(x[i]);
  solve;
  write;
end
```

- The code shown above is available on the course webpage (in the SGF folder – Gradient Code)
- It's a good idea to create a new folder for each of your SGF code files, since on running SG, a lot more extra files are generated.
- Download the code file(gradient.sg) and compile the code

SG gradient

- In this program, we have not explicitly written the answers into an answer file as we did in our previous cases (refer SGF Tutorial 1)
- By default, the results from your program (the variables used) are stored in a .res file.

Example: For the gradient program that you ran, the results are stored in **gradient.res**

- Since this file is in an unreadable format, we need to extract the results into a standard format. To do this use the command: **extract filename.res**
- Extracting the file results in the each of the variables being stored in a separate variablename.000 file.

- ✓ In our gradient program – first extract the results, by the command

extract gradient.res

- ✓ You should be having three new .000 files in your directory (you can check using 'ls')

V.000 – values of the potential array

E.000 – values of the Electric field array

x.000 – values of the distance array ($x[i] - x[i-1] = dx$)

- You can check the contents of the files using any text editor (Unix: pico, emacs ; Windows: notepad, WordPad) or just a display command (Unix: more). In this case, each of the files contain 101 values (size of the array)

Plotting results:

- We will use matlab to plot our results. To run Matlab from a unix terminal, use the command: **matlab &** (the & prevents your terminal window from getting locked, so that you can still use the terminal window for other commands)
- Matlab will open up eventually (it does take time to load) and by default it has a command window with a prompt '>>'. All commands can be typed at this prompt.
- If Matlab opens and you don't get a prompt '>>' then it's a problem with the matlab settings in unix. You need to delete the .matlab folder in your home directory - exit matlab, then do either one of the following
 - ✓ In Windows, go to U:/ and delete the **.matlab** folder (it's a hidden folder), or
 - ✓ In Unix, in the home directory delete the **.matlab** folder

- Load the variables that you need to plot
 >> **load V.000**
 >> **load E.000**
 >> **load x.000**
- The three arrays are now stored in Matlab with the same variable names as the filenames (E, V and x). To check the variables in Matlab at any point use the command: **who**
- Now let's plot the potential (V):
 >> **plot(x, V)** (this command uses values of x for the x-axis and V for the y-axis)

It should be plotting a cosine wave from $0 - 2\pi$ since we defined x as an array of values from $0 - 2\pi$ and V was $\cos(x[i])$ – Refer program above

- Similarly you can plot the Electric field (E):
 >> **plot(x,E)**

We'd defined E at the negative differential of V, so E is a sine wave from $0-2\pi$
 Since $E = -dV/dx$
 $V = \cos(x)$
 $dV/dx = -\sin(x)$
 $E = -(-\sin(x)) = \sin(x)$

- For help using a Matlab command, use **help command-name**
 Example: For help using the plot command
 >> **help plot**
- For most commonly used Matlab commands, check the CAE matlab handout. You can get a copy at the CAE main windows lab: CAE 170. Additional information on Matlab (CAE tutorials, CAE FAQ etc) is available at http://www.cae.wisc.edu/software/display_tpl.php?aid=315
- If you'd rather work with the windows version of Matlab, you can do so. It should be under CAE Applications > Engineering. Also all your files (.000 result files) should be available in U:\ . Once you open matlab, you need to change the working directory to the directory that contains the .000 files.

Additional Work:

- Go and modify the Voltage distribution in the file gradient.sg

Let

1) $V = \sin(x)$

2) $V = x^2$ (in SGF: the code denoting x^2 is **sq(x)**)

what will the Electric field be in both cases?

Plot the results and check if it coincides with the theoretical calculations.