Numerical Analysis with Python

LECTURE 01

Introduction to Numerical Modelling with Python

Introduction to Numerical Analysis

- ‘Numerical methods are techniques by which mathematical problems are formulated so that they can be solved with arithmetic operations”.

- There are many kinds of numerical methods, they have one common characteristic: they invariably involve large numbers of tedious arithmetic calculations.

- The role of numerical methods in engineering modelling and problem solving has increased dramatically in recent years due to the development of fast and efficient **Digital Computers**

Non-computer Approach to Numerical Analysis

1. Analytical or Exact methods:

“ provided excellent insight into the behavior of some systems. However, analytical solutions can be derived for only a limited class of problems. These include those that can be approximated with linear models and those that have simple geometry and low dimensionality.”

Non-computer Approach to Numerical Analysis

2. Graphical Methods:

“These graphical solutions usually took the form of plots or nomographs. Although graphical techniques can often be used to solve complex problems, the results are not very precise. Furthermore, graphical solutions (without the aid of computers) are extremely tedious and awkward to implement.|”

Non-computer Approach to Numerical Analysis

3. Calculator and slides rules:

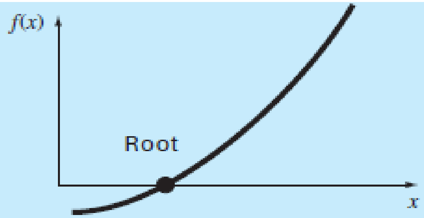
“Manual calculations are slow and tedious. Furthermore, consistent results are elusive because of simple blunders that arise when numerous manual tasks are performed.”

Merits of Numerical Methods in Engineering

1. They are capable of handling large systems of equations, nonlinearities, and complicated geometries that are often impossible to solve **analytically**.
2. They can be commercially produced as prepackaged software that can be use by even non-expert to solve engineering problems.
3. Expert with underlining knowledge of system models can produce their own numerical software to solve particular problem.
4. Numerical methods are an efficient vehicle for learning to use computers.
5. Numerical methods provide a vehicle for you to reinforce your understanding of mathematics.

Mathematical background for NA

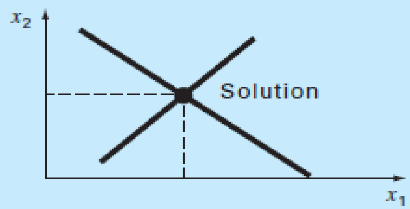
(*a*) *Part 2:* Roots of equations
Solve $f(x) = 0$ for x.



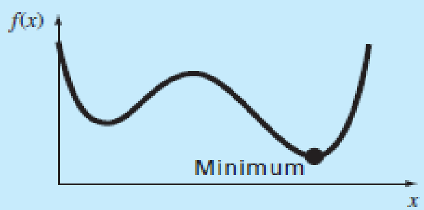
(*b*) *Part 3:* Linear algebraic equations
Given the a's and the c's, solve

$$a_{11}x_1 + a_{12}x_2 = c_1$$
$$a_{21}x_1 + a_{22}x_2 = c_2$$

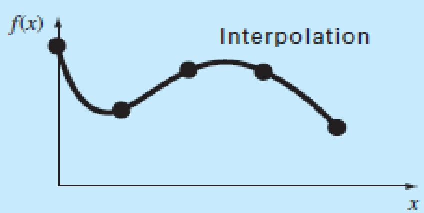
for the x's.



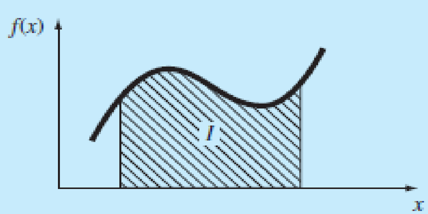
(*c*) *Part 4:* Optimization
Determine x that gives optimum $f(x)$.



(*d*) *Part 5:* Curve fitting



(*e*) *Part 6:* Integration
$I = \int_a^b f(x)\,dx$
Find the area under the curve.

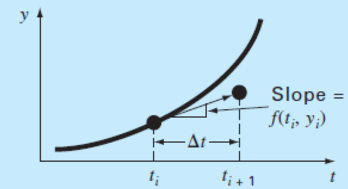


(*f*) *Part 7:* Ordinary differential equations
Given

$$\frac{dy}{dt} \simeq \frac{\Delta y}{\Delta t} = f(t, y)$$

solve for y as a function of t.

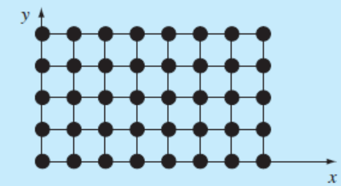
$$y_{i+1} = y_i + f(t_i, y_i)\,\Delta t$$



(*g*) *Part 8:* Partial differential equations
Given

$$\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f(x, y)$$

solve for u as a function of x and y



Mathematical Modeling & Problem solving

- **What is a mathematical Model ?**

‘’is broadly defined as a formulation or equation that expresses the essential features of a physical system or process in mathematical terms’’

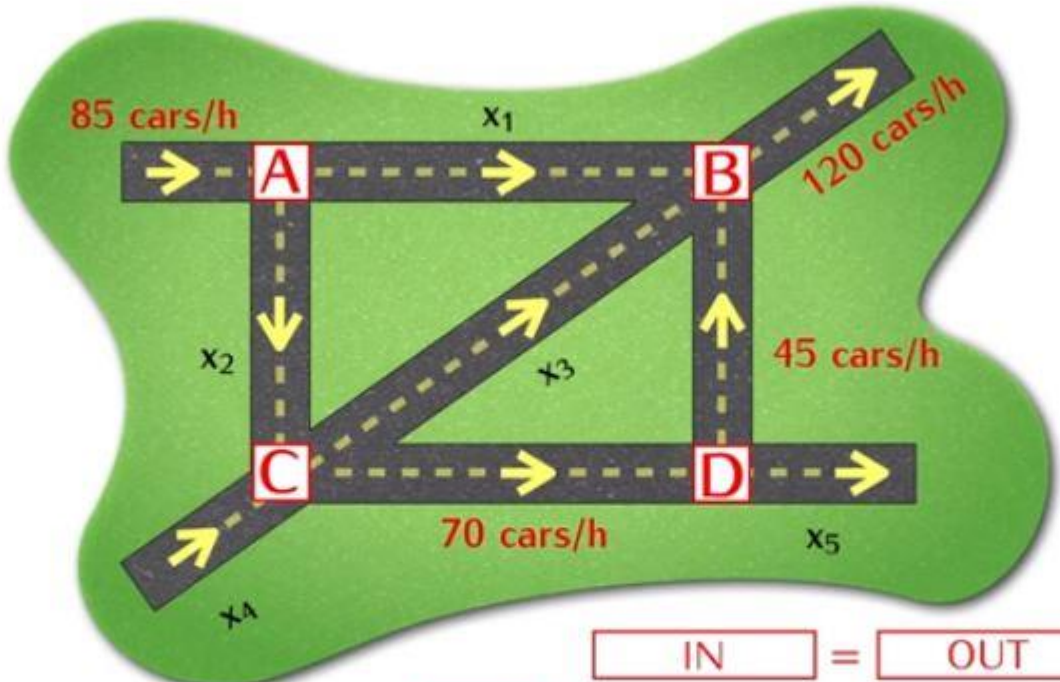
It can take the form;

$$\textit{Dependent variable} = f\left(\begin{matrix}\textit{independent}\ , \\ \textit{variable}\end{matrix} \quad \textit{parameters}, \quad \begin{matrix}\textit{forcing} \\ \textit{functions}\end{matrix}\right)$$

- ***Dependent variable***: characteristic that usually reflects the behavior or state of the system
- ***Independent variable***: are usually dimensions, such as time and space, along which the system’s behavior is being determined
- ***Forcing Functions***: *are external influences acting upon the system*
- ***Parameters***: *are reflective of the system’s properties or composition*

Mathematical Modeling & Problem solving

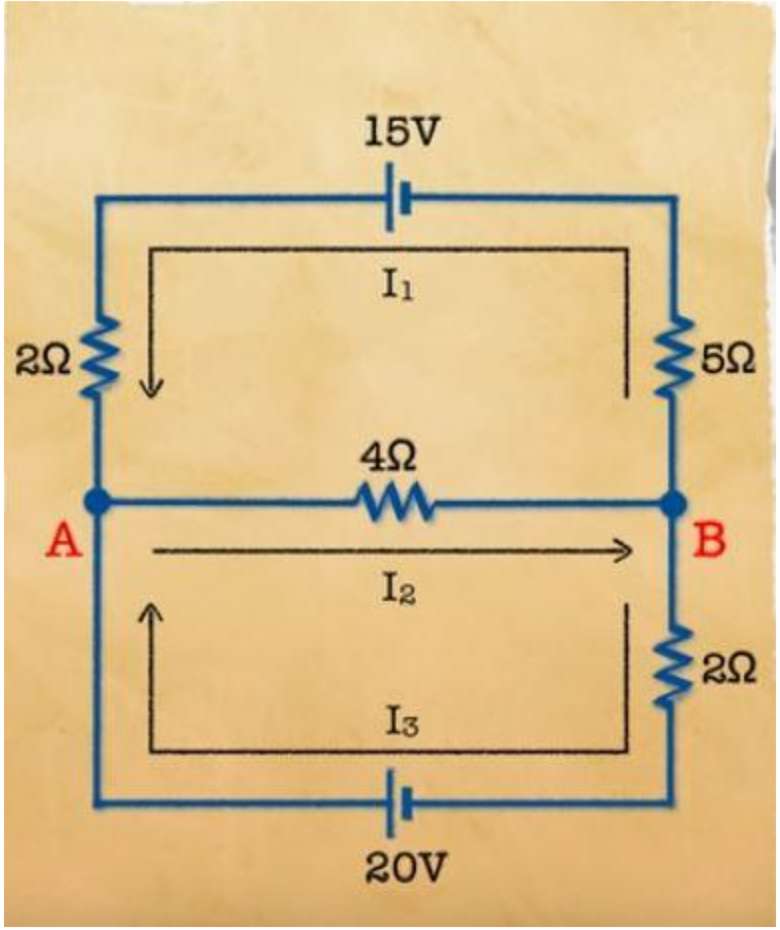
- **Traffic mathematical Model as system of linear equations**



	IN = OUT
total:	$85 + x_4 = 120 + x_5$
@ A:	$85 = x_1 + x_2$
@ B:	$x_1 + x_3 + 45 = 120$
@ C:	$x_2 + x_4 = 70 + x_3$
@ D:	$70 = 45 + x_5$

Mathematical Modeling & Problem solving

- **Electrical circuit mathematical Model as system of linear equations**



$$I_1 - I_2 + I_3 = 0$$

$$7I_1 + 4I_2 - 15 = 0$$

$$4I_2 + 2I_3 - 20 = 0$$

Model of a falling Object

- *Assume a free falling object near the earth surface*

$$Newtons\ Second\ law, net\ force, on\ the\ object\ F,$$
$$F = ma$$

Or

$$a = \frac{F}{m} \qquad (1.1)$$

Where F, net force in *N*
M, is the mass of the body in Kg
A, is acceleration in m/s^2

Substituting $a = \frac{dv}{dt}$ and net force $F = F_D + F_V$ in (1.1)

$$\frac{dv}{dt} = \frac{F_D + F_V}{m} \qquad (1.2)$$

Model of a falling Object

- The Net force F acting on the body consist of two components

1. The downward force $\mathbf{F_D}$ due to the gravitational drag
2. The upward force $\mathbf{F_V}$ due to air resistance

$$F_D = mg \quad where\ g\ is\ accelaration\ due\ to\ gravity$$

$$F_V = -cv$$

$$where\ c\ is\ a\ constant\ called\ \boldsymbol{drag\ coefficient}\ and\ v\ is\ the\ air\ velocity$$

Or

$$F = F_D + F_V = \text{mg} - cv \qquad (1.3)$$

Hence substituting (1.3) in (1.2) we have (1.4)

$$\frac{dv}{dt} = g - \frac{c}{m}v \qquad (1.4)$$

Analytical solution to falling Object model

- The model that relates the acceleration of a falling object to the forces acting on it.

- It is a differential equation because it is written in terms of the differential rate of change ($\frac{dv}{dt}$) of the variable that we are interested in predicting.

- Simple algebraic manipulation can not solve the equation (1.4), hence advanced techniques in calculus is required.

Analytical solution to falling Object model

- For example, if the parachutist is initially at rest (v=0 at t=0), calculus can be used to solve Eq. (1.4) for $\boldsymbol{V(t)}$.

$$\boldsymbol{V(t) = \frac{gm}{c}\left(1 - e^{-\left(\frac{c}{m}\right)t}\right)} \qquad \boldsymbol{(1.5)}$$

Exercise: prove (1.5)

Analytical solution to falling Object model

Problem statement 1:

- *A parachutist of mass 68.1 kg jumps out of a stationary hot air balloon. Using analytical solution of Eq. (1.5) to compute velocity prior to opening the chute. The drag coefficient is equal to 12.5 kg/s.*

Solution: Substituting the parameters in the equation below

$$V(t) = \frac{gm}{c}\left(1 - e^{-\left(\frac{c}{m}\right)t}\right)$$

$$V(t) = \frac{9.81(68.1)}{12.5}\left(1 - e^{-\left(\frac{12.5}{68.1}\right)t}\right)$$

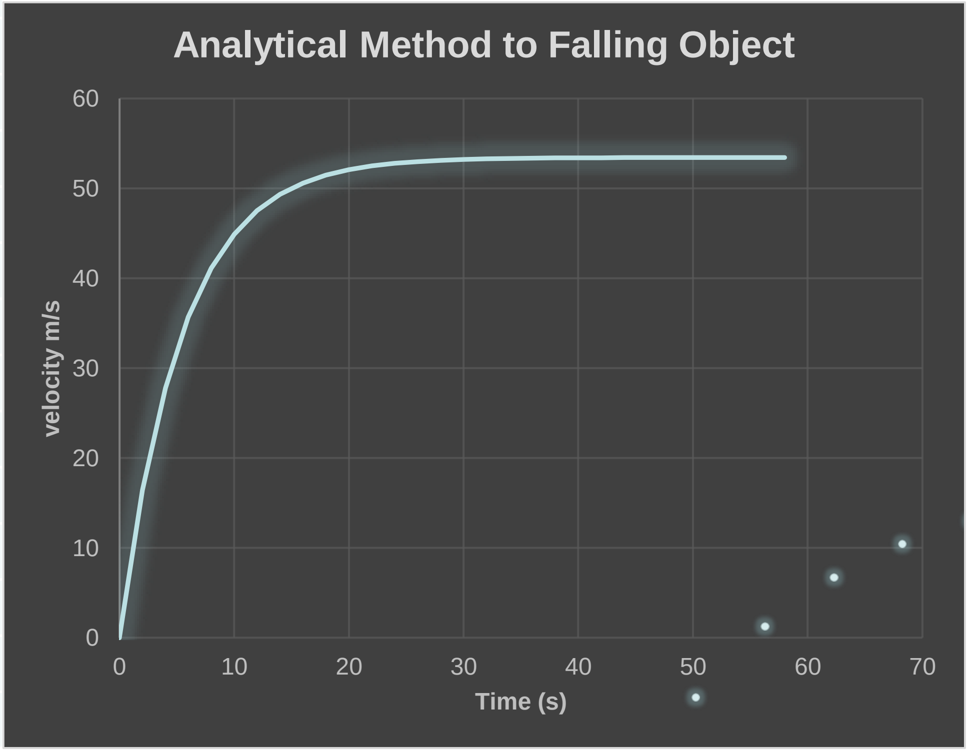
$$V(t) = 53.44\left(1 - e^{-0.18355t}\right)$$

Analytical solution to falling Object model

- *Computing velocities at different times yields the following results*

$$V(t) = 53.44\left(1 - e^{-0.18355t}\right)$$

Time(s)	Velocity (m/s)
0	0
2	16.41995476
4	27.79472025
6	35.67447948
8	41.13310679
10	44.91451827
12	47.53405465
14	49.34871325
16	50.60580051
18	51.47663561
20	52.07989823
22	52.4978026
24	52.78730183
26	52.98784963
28	53.12677718
30	53.22301791
∞	53.44



Numerical solution to falling Object model

- To solve the problem modelled by Eq. (1.4) numerically, *we can use finite difference method (Euler's Method) to approximate* $\frac{dv}{dt}$ *in Eq. (1.4)*

$$\frac{dv}{dt} \cong \frac{\Delta v}{\Delta t} = \frac{v(t_{i+1}) - v(t_i)}{t_{i+1} - t_i} \qquad (1.6)$$

- Now using (1.6) in Eq. 1.4 we have,

$$\frac{v(t_{i+1}) - v(t_i)}{t_{i+1} - t_i} = g - \frac{c}{m} v$$

- On rearranging the equation above

$$\boldsymbol{v(t_{i+1}) = v(t_i) + \left[g - \frac{c}{m} v(t_i)\right](t_{i+1} - t_i)} \qquad \mathbf{(1.7)}$$

Numerical solution to falling Object model

- $v(t_{i+1})$ *is new velocity*
- $v(t_i)$ *is old velocity,*
- $(t_{i+1} - t_i) = step\ size\ or\ time\ difference$

Problem statement 2

- *A parachutist of mass 68.1 kg jumps out of a stationary hot air balloon. Use numerical solution of Eq. (1.7) to compute velocity prior to opening the chute. The drag coefficient is equal to 12.5 kg/s. Use* $t_i = 0$ *and* $t_{i+1} = 2s$.

Solution: Substituting the parameters in the equation below

Step 1: $t_i = 0, v(t_i) = 0$

$$stepsize\ (t_{i+1} - t_i) = 2 - 0 = 2s$$

$$\boldsymbol{v(t_{i+1}) = v(t_i) + \left[g - \frac{c}{m} v(t_i)\right](t_{i+1} - t_i)}$$

Numerical solution to falling Object model

$$v(t_{i+1}) = 0 + \left[9.81 - \frac{12.5}{68.1} * 0\right] * 2$$

$$v(t_{i+1}) = 19.62\ m/s$$

Step 2: now at , $t_i = 2\ s, v(t_i) = 19.62 m/s$

$$v(t_{i+1}) = 19.62 + \left[9.81 - \frac{12.5}{68.1} * 19.62\right] * 2$$

$$v(t_{i+1}) = 32.04\ m/s$$

Step 3: : now at , $t_i = 4\ s, v(t_i) = 32.04 m/s$

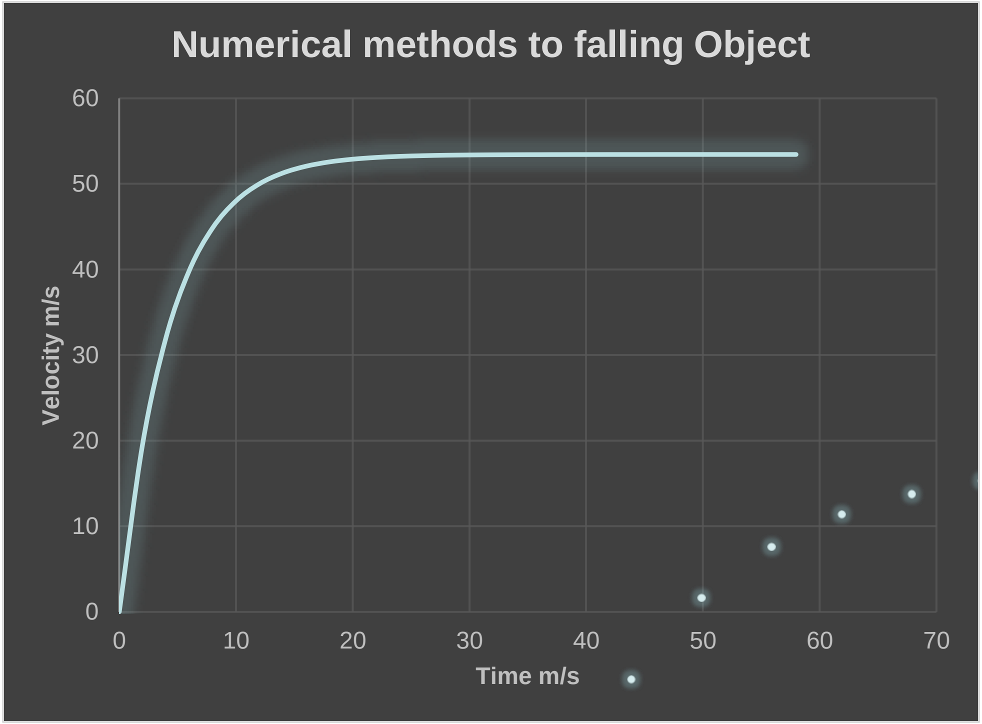
$$v(t_{i+1}) = 32.04 + \left[9.81 - \frac{12.5}{68.1} * 32.04\right] * 2$$

$$v(t_{i+1}) = 39.90\ m/s$$

- Repeating at subsequent intervals will yield the results below

Numerical solution to falling Object model

Time(s)	Velocity (m/s)
0	**0**
2	**19.62**
4	**32.03735683**
6	**39.89621262**
8	**44.8700259**
10	**48.01791654**
12	**50.01019387**
14	**51.27109186**
16	**52.06910513**
18	**52.57416198**
20	**52.89380883**
22	**53.09611102**
24	**53.22414662**
26	**53.30517943**
28	**53.35646452**
30	**53.38892248**



Numerical Methods with Python

- Given the two approaches (analytical and numerical), its obvious that the numerical approach can find solution similar to the exact solution.
- Tedious computation is needed to get more accurate results, but fortunately this can easily be done using computer.
- The numerical solution to falling object problem can be implemented using a software package like **PYTHON** and **MATLAB** to easily fine solution.
- Throughout the course PYTHON will be used for numerical analysis

Numerical Methods with Python

- Python IDE and Installation: There many Integrated development environment (IDE) use to create python program.
- In this course Python SPYDER IDE distributed by Anaconda is recommended
- For instructions on how to download and install python SPYDER on different operating systems visit the link below:

1. https://www.anaconda.com/distribution/

- **Anaconda** is one of several Python distributors. Python distributions provide the Python interpreter, together with a list of Python packages and sometimes other related tools, such as editors.

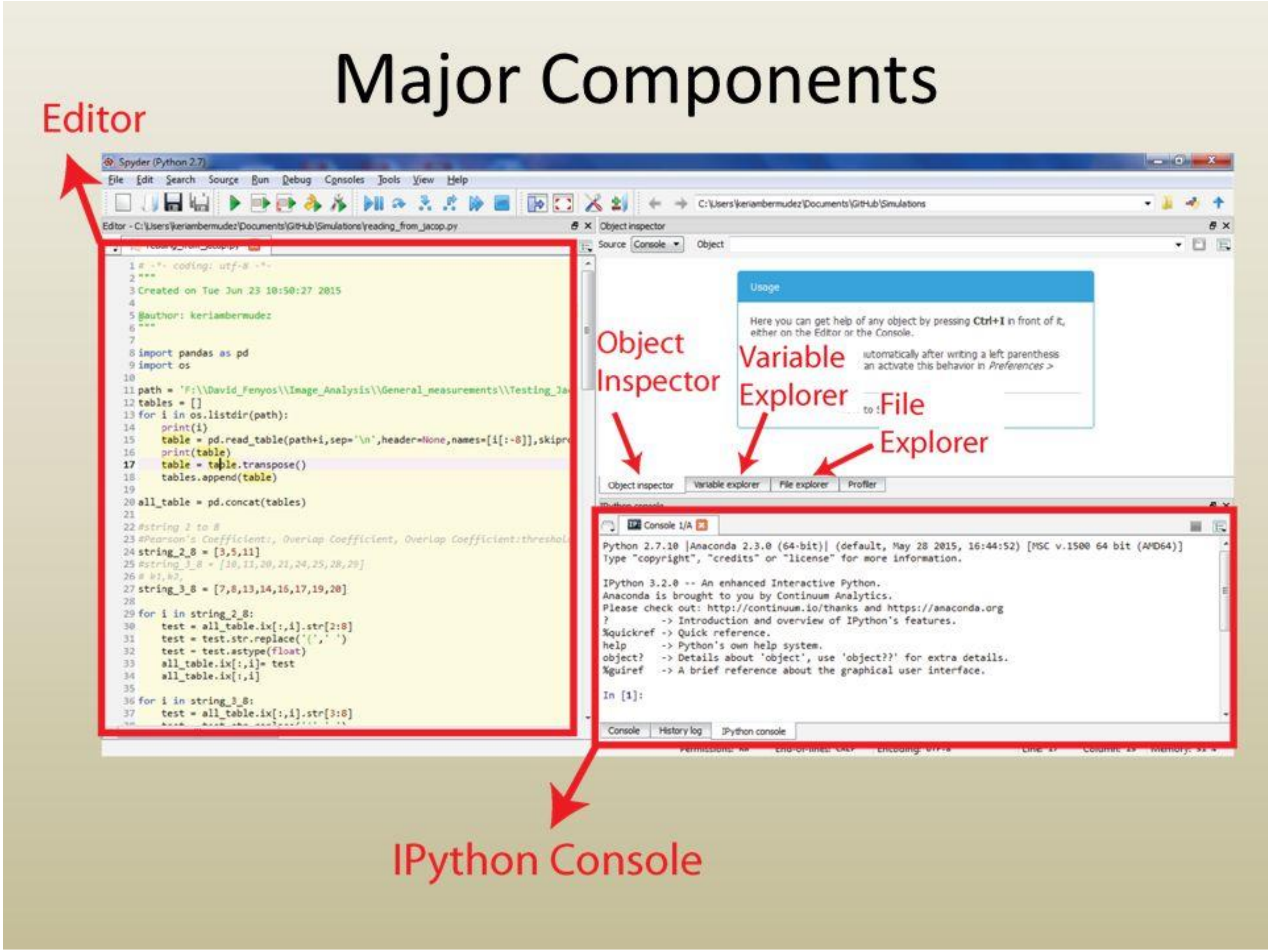
Numerical Methods with Python

- **Python packages:** For scientific computing and computational modelling, we need additional libraries (so called packages) that are not part of the Python standard library.

- The packages we generally need are:
- **NumPy:** (Numeric Python): For matrices and linear algebra
- **Pandas**: Python data science tools (Series and Dataframes)
- **SciPy:** (Scientific Python): many numerical routines
- **matplotlib**: (Plotting Library) creating plots of data
- **Sympy:** (Symbolic Python): symbolic computation
- **Pytest**: (Python Testing): a code testing framework

Numerical Methods with Python

- Python Spyder IDE



Numerical Methods with Python

- **Packages Installation:** all python packages can be installed via the python installation manager found in python version 3.4 and above.

- In python command window or Ipython (Spyder):

```
>>  pip install numpy
```

- Will download and install NumPy packages

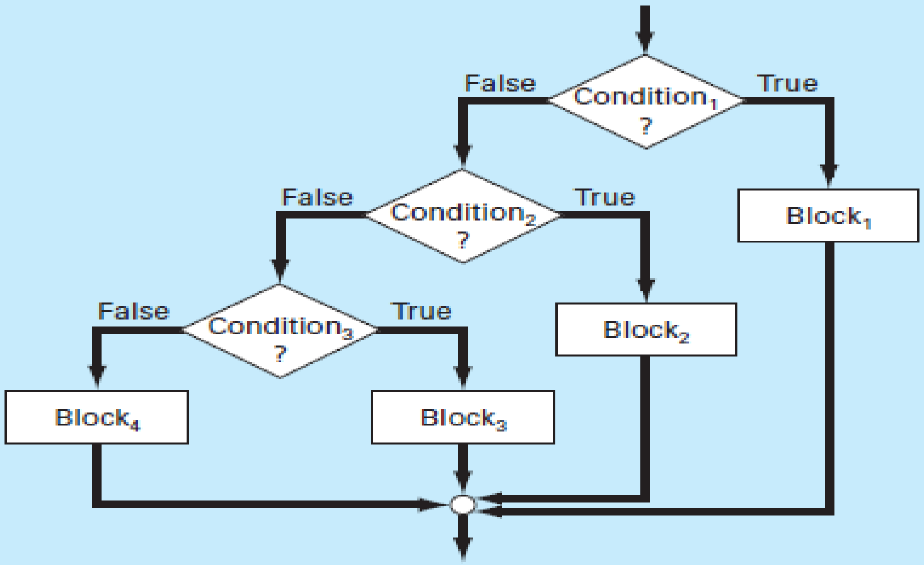
- In the program script:

```
import numpy as np
```

- Will import all libraries from NumPy packages and create an object np of those libraries.

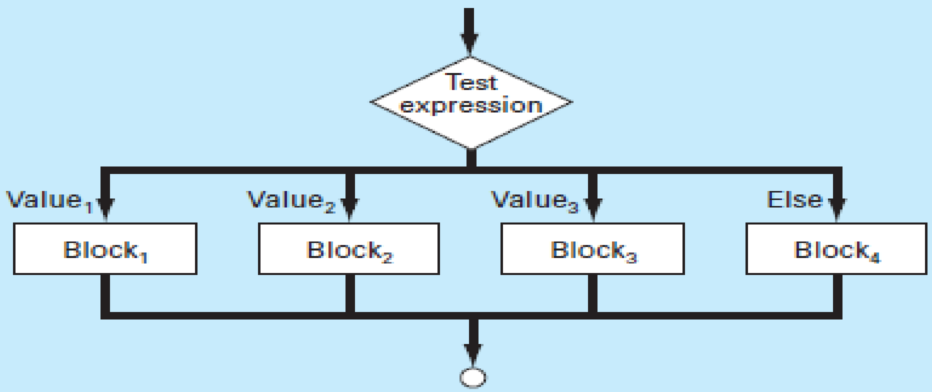
- Flowchart and Pseudocodes

Flowchart | Pseudocode




```
IF condition₁ THEN
  Block₁
ELSEIF condition₂
  Block₂
ELSEIF condition₃
  Block₃
ELSE
  Block₄
ENDIF
```

(a) Multialternative structure (IF/THEN/ELSEIF)




```
SELECT CASE Test Expression
  CASE Value₁
    Block₁
  CASE Value₂
    Block₂
  CASE Value₃
  Block₃
  CASE ELSE
  Block₄
END SELECT
```

(b) CASE structure (SELECT or SWITCH)

Numerical Methods with Python

- Pseudocodes for solving falling Object Using Euler's Numerical Approach

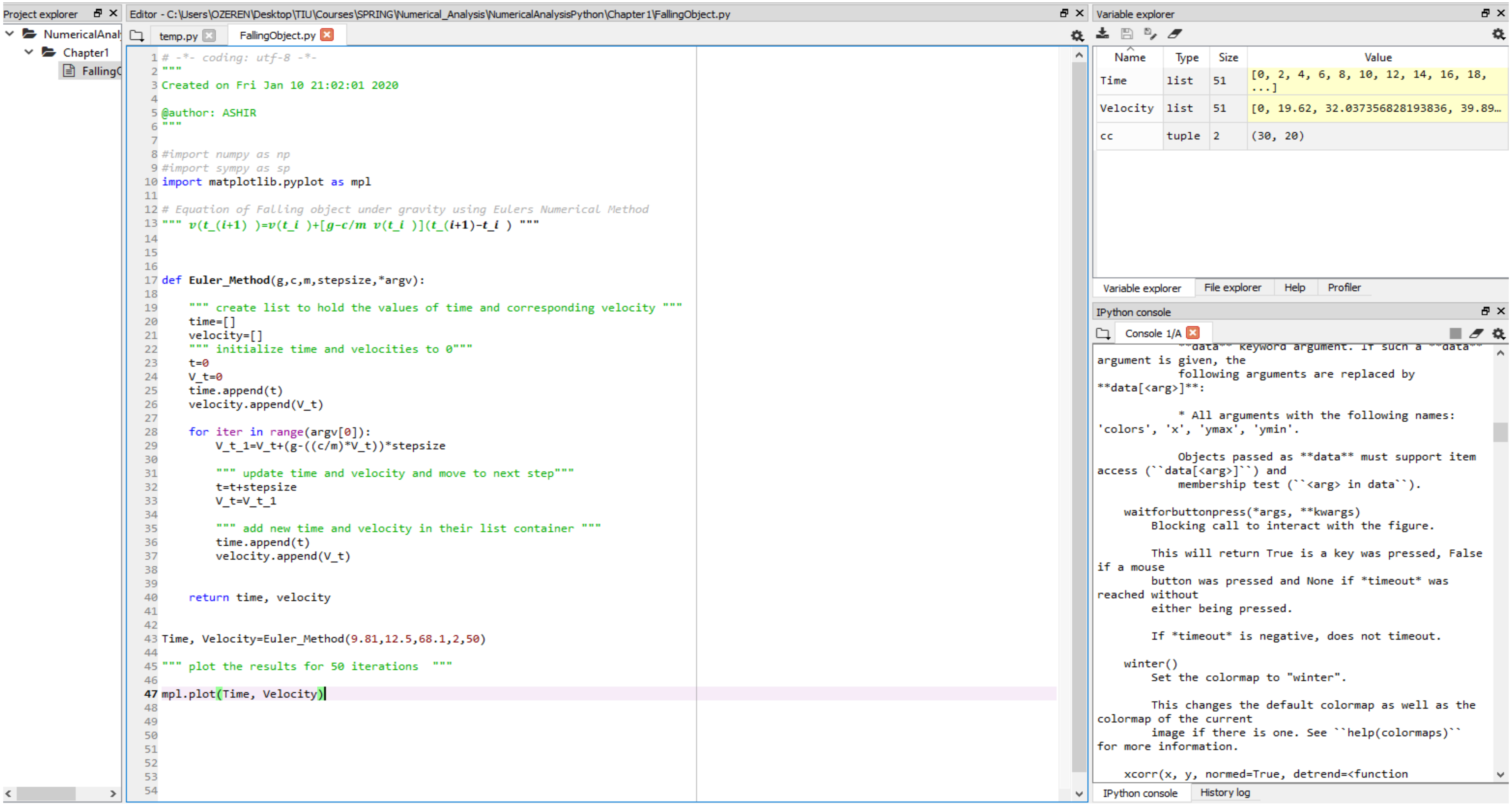
$$\boldsymbol{v(t_{i+1}) = v(t_i) + \left[g - \frac{c}{m}v(t_i)\right](t_{i+1} - t_i)}$$

1. *Input g , c, m, stepsize*
2. *Initialize* $v(t_i) = 0, and t_i = 0$
3. *Compute* $v(t_{i+1}) = v(t_i) + \left[g - \frac{c}{m}v(t_i)\right] * stepsize$
4. *Replace* $v(t_i)$ *with new velocity computed in step 3*
5. *If stoppage criteria is not met GOTO step 3 Else GOTO 6*
6. *Return* $v(t_{i+1})$
7. *Stop*

Numerical Methods with Python

- Python function for solving falling Object Using Euler's Numerical Approach

$$v(t_{i+1}) = v(t_i) + \left[g - \frac{c}{m} v(t_i)\right](t_{i+1} - t_i)$$



Numerical Methods with Python

- Python function for solving falling Object Using Euler's Numerical Approach

$$\boldsymbol{v(t_{i+1}) = v(t_i) + \left[g - \frac{c}{m} v(t_i)\right](t_{i+1} - t_i)}$$

$$v(t_{i+1}) = v(t_i) + \left[g - \frac{c}{m} v(t_i)\right](t_{i+1} - t_i)$$