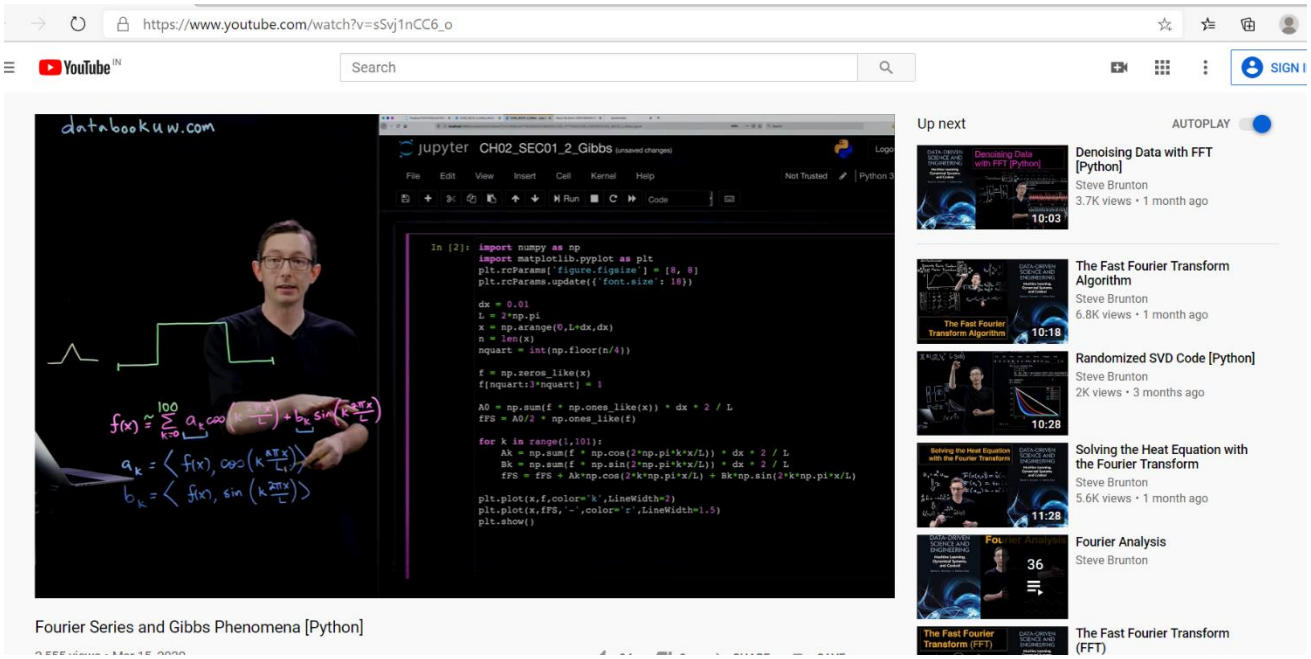
DAILY ASSESSMENT FORMAT

Date:	**26-05-2020**	**Name:**	**BINDUSHRI**
Course:	**dsp**	**USN:**	**4AL17EC011**
Topic:	**1.fourier series&gibbs phenomena using python.** **2.fourier transform derivative** **3.fourier transform and convolution.** **4.inition of fourier transform&laplace transform** **5.laplace transform of firstorder** **6.implementation of fourier transform using matlab** **7.application of z transform** **8.z transform of sequence using matlab.** **9.fourier transform**	**Semester & Section:**	**6th A**
Github Repository:	**Bindushri**		

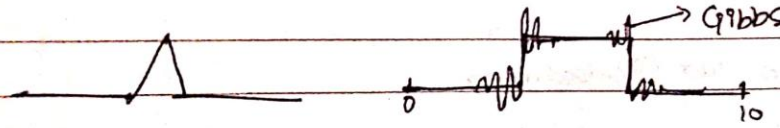
FORENOON SESSION DETAILS


Day 2 :- 26-05-2020 / FOURIER SERIES & GIBBS Phenomena using [Python]

$$f(x) \cong \sum_{k=0}^{100} a_k \cos\left(k \frac{2\pi x}{L}\right) + b_k \sin\left(k \frac{2\pi x}{L}\right)$$

$$a_k = \langle f(x), \cos\left(k \frac{2\pi x}{L}\right) \rangle$$

$$b_k = \langle f(x), \sin\left(k \frac{2\pi x}{L}\right) \rangle$$



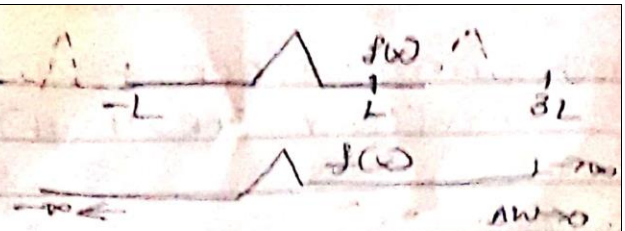
* Fourier transform :-

$$f(x) = \sum_{k=-\infty}^{\infty} c_k e^{ik\pi x/L}$$

$$c_k = \frac{1}{2\pi} \langle f(x), \psi_k \rangle = \frac{1}{2L} \int_{-L}^{L} f(x) \underbrace{e^{-ik\pi x/L}}_{\psi_k} dx$$

$w_k = \frac{k\pi}{L}$

$\Delta w = \pi/L$



$$f(x) = \lim_{\Delta w \to 0} \sum_{k=-\infty}^{\infty} \frac{\Delta w}{2\pi} \int_{-\pi/\Delta w}^{\pi/\Delta w} f(\xi) e^{-ik\Delta w \xi} d\xi \; e^{ik\Delta w x}$$

$$= \int_{-\infty}^{\infty} \frac{1}{2\pi} \int_{-\infty}^{\infty} f(\xi) e^{-i\omega\xi} d\xi \; e^{i\omega x} d\omega$$

$$\hat{f}(\omega) = \mathcal{F}(f(x)) = \int_{-\infty}^{\infty} f(x) e^{-i\omega x} dx$$

$$f(x) = \mathcal{F}^{-1}(\hat{f}(\omega)) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{f}(\omega) e^{i\omega x} d\omega$$

* Fourier transform derivative.

$$\mathcal{F}\left(\frac{d}{dx} f(x)\right) = \int_{-\infty}^{\infty} \underbrace{\frac{df}{dx}}_{dv} \underbrace{e^{-i\omega x}}_{u} dx$$

$$= \underbrace{\left[f(x) e^{-i\omega x}\right]_{-\infty}^{\infty}}_{uv=0} - \int_{-\infty}^{\infty} \underbrace{f(x)}_{v} \underbrace{(-i\omega e^{-i\omega x})}_{du} dx$$

$$= i\omega \int_{-\infty}^{\infty} f(x) e^{-i\omega x} dx = i\omega \underbrace{\mathcal{F}(f(x))}_{\mathcal{F}(f(x))}$$

$$\mathcal{F}\left(\frac{df}{dx}\right)$$

$$u_t = c u_{xx} \xrightarrow{\mathcal{F}} \hat{u}_{tt} = -\omega^2 \hat{u}$$

(PDE) (ODE)

* Fourier transform and convolution

$$(f * g) = \int_{-\infty}^{\infty} f(x-\xi) g(\xi) d\xi$$

$$\mathcal{F}(f*g) = \mathcal{F}(f)\mathcal{F}(g) = \hat{f}\hat{g}$$

$$\mathcal{F}^{-1}(\hat{f}\hat{g})(x) = \frac{1}{2\pi} \int_{-\infty}^{\infty} \hat{f}(\omega) \hat{g}(\omega) e^{i\omega x} d\omega$$

$$= \frac{1}{2\pi} \int_{-\infty}^{\infty} f(\omega) \left(\int_{-\infty}^{\infty} g(y) e^{-i\omega y} dy \right) e^{i\omega x} dx d\omega$$

$$= \frac{1}{2\pi} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(\omega)\, g(y)\, e^{i\omega(x-y)} d\omega dy$$

$$= \frac{1}{2\pi} \int_{-\infty}^{\infty} g(y) \int_{-\infty}^{\infty} f(\omega)\, e^{i\omega(x-y)} d\omega dy.$$

$$\int_{-\infty}^{\infty} g(y) f(x-y) dy.$$

$$= f * g$$

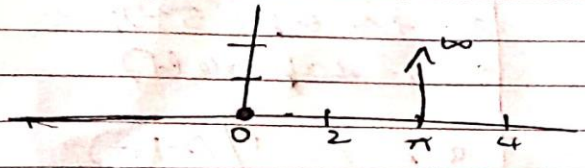
* Intition of fourier Transform & laplace transform

$$\mathcal{F}\{f(t)\} = \int_{-\infty}^{\infty} f(t) \cos(\omega t) dt - i \int_{-\infty}^{\infty} f(t) \sin(\omega t) dt.$$

$\mathcal{F}\{\cos(\pi x)\} \rightarrow \cos(\pi x)[\cos(\omega x)]$

$\rightarrow \cos(\pi x)[\sin(\omega x)]$

$\mathcal{F}\{\cos(\pi x)\} = \cos(\pi x)[\cos(\pi x)]$ area = 0



$$\int_{-\infty}^{\infty} (\cos \pi x)[\cos \pi x]\, dt = \infty \checkmark$$

Fourier: $F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-i\omega t} dt$

Laplace: $F(s) = \int_{0}^{\infty} f(t) e^{-(s+i\omega)t} dt.$

$$F(s) = \int_{0}^{\infty} f(t)\, e^{-i\omega t}\, e^{-st} dt$$

* laplace transform of first order

1st order equation. the transforms of $f(t)$ and $y(t)$ are $F(s)$ and $Y(s)$

definition: $F(s) = \int_{0}^{\infty} f(t) e^{-st} dt$

example

$f(t) = e^{at}$

$$F(s) = \int_{0}^{\infty} e^{at} e^{-st} dt \quad \left[\frac{e^{(a-s)t}}{a-s} \right]_0^{t=\infty}$$

$= \frac{1}{s-a} = Y(s)$

$dy/dt - ay = 0$ $\int_0^\infty dy/dt\, e^{-st}\, dt = \int_0^\infty y(t)\, e^{-st}\, dt$

$+[y e^{-st}]_0^\infty$

TRANSFORM $dy/dt = sY(s) - y(0)$

$Y(s) = \frac{y(0)}{s-a} \xrightarrow{\text{INVERSE LT}} y(t) = y(0)e^{at}$

* Implementation of laplace transform using matlab

Basic

```
1. clc;
2. clear all;
4. syms t f L
5. f = 5*t
6. L = laplace(f)
```

o/p L = 5/s^2.

Basic

```
1. clc;
2. clear all;
3. syms t f;
4. f = exp(-5*t);
5. L = laplace(f)
```

o/p L = 1/(s+5).

complex

```
1. clc;
2. clear all:
4. syms t f a;
5. f = (exp(-3*t)*sin(2*t))/t
6. L = laplace(f).
```

o/p = atan(2/(s+3))

Inverse

```
1. clc;
2. clear all;
3. syms F s;
4. F = 1/(s^2+1)
5. f = ilaplace(F)
```

o/p = sin(t)

* Applications of Z-transform

$\Delta f(x) = f(x+1) - f(x)$

$\Delta y_{n+1} = y_{n+2} - y_{n+1}$

& hence

$\Delta y_{n+1} + y_n = 2$

$\Rightarrow y_{n+2} - y_{n+1} + y_n = 2$

$\Delta y_{n+1} + \Delta^2 y_{n-1} = 1$

$y_{n+2} - y_{n+1} + \Delta(\Delta y_{n-1}) = 1$

Now, $\Delta y_{n-1} = y_n - y_{n-1}$

& so,

$\Delta(\Delta y_{n-1}) = \Delta(y_n - y_{n-1})$

$\Delta y_n - \Delta y_{n-1}$

$y_{n+1} - y_n - (y_n - y_{n-1})$

$= y_{n+1} - 2y_n + y_{n-1}$

$\rightarrow y_{n+2} - y_{n-1} + y_n = 2$

$y_{n+2} - 2y_n + y_{n+1} = 1$

∴ Find the z-transform of sequence using matlab.

* Command window

→ syms n;

>>>

>>> a = n+1;

>>> b = ztrans(a);

>>> disp(b)

z/(z-1) + z/(z-1)^2

>> pretty(b)

o/p
$$\frac{z}{z-1} + \frac{z}{(z-1)^2}$$

2)

>>> a = 2^n;

>> b = ztrans(a);

>> b

o/p → b = z/(z-2)

3) >> syms n w;

>> a = sin(w*n);

>> b = ztrans(a);

>> disp(b)

(sin(w))/(z^2) - z*(cos(w)*2+1)

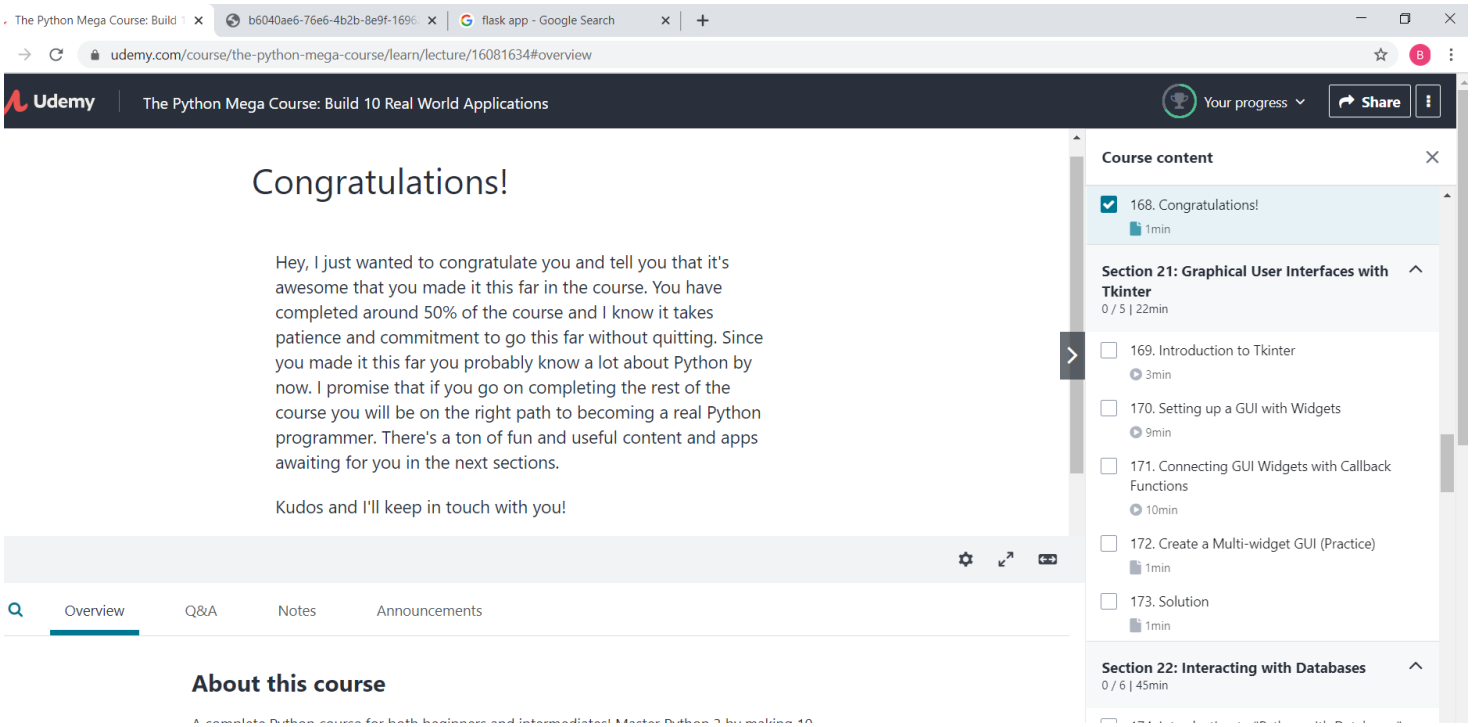
o/p
$$\frac{z\sin(\omega)}{z^2 - \cos(\omega)z^2 + 1}$$

Date:26may2020
Course: python
Topic:
Basics:sec20-22

Name:Bindushri
USN:4AL17EC011
Sem&Sec:6th A

AFTERNOON SESSION DETAILS

Image of session



Python. ISCAPSL.PY

26-05-2020.

Doc 20 Personal website - How the O/P will look like

* https://pythonflasktutorial.herokuapp.com.

→ pip install flask

demo → script.py

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def home():
        return "website content goes here!"

if __name__ == "__main__":
    app.run(debug=True)
```

to check O/P open chrome ⇒ localhost:5000

→ HTML templates

```
from flask import Flask, render_template
app = Flask(__name__)

@app.route('/')
def home():
        return render_template("home.html")

@app.route('/about')
def about():
        return "about content goes here!"

if __name__ == "__main__":
        app.run(debug = True)
```

[Now "home.html" should reflect real html code that has to be created.
right click on demo (folder) → New folder → (named as) templates → (Paste the templates (create a new file) (right click on templates) → templates\home.html → enter]

write a html file inside home.html

```
1. <!DOCTYPE html>
2. <html>
3. <body>
4.          <h1>My home page</h1>
5.          <p>This is a test website</p>
6. </body>
7. </html>
```

o/p # (check in localhost:5000)

My home page
this is a test website.

[Now to create another html page
rightclick on home.html → Duplicate → (rename) about.html]

start write html code at about.html

```
1. <!DOCTYPE html>
2. <html>
3. <body>
4.          <h1>my about page</h1>
5.          <p>this is a test website again</p>
6. </body>
7. </html>
```

[Eg: open script.py] now make changes in code

```
def about():
    return render_template("about.html")
```

O/P : check localhost:5000/about/

My about page.
this is a test website again.

[Now create one more html folder under templates i.e. "layout.html" & start writing code here]

```
<!DOCTYPE html>
<html>
  <body>
    <header>
      <div class = "container">
        <h1 class = "logo"> Flask's web app</h1>
        <strong><nav>
          <ul class = "menue">
            <li><a href = "{{url_for('home')}}">
              HOME</a>
            </li>
            <li><a href = "{{url_for('about')}}">
              About</a></li>
          </ul>
        </nav></strong>
      </div>
    <header>
    <div class = "container">
      {% block content %}
      {% end block %}
    </div>
  </body>
</html>
```

Script.py code.

```
1. from Flask import Flask, render_template
2.
3. app = Flask (__name__)
4.
5. @app.route ('/')
6. def home():
7.         return render_template ("home.html")
8.
9. @app.route ('/about/')
10. def about():
11.         return render_template ("about.html")
12.
13. if __name__ == "__main__":
14.     app.run (debug = True)
```

⇒ [make changes as well in home.html]

```
1. {% extends "layout.html" %}
2. {% block content %}
3.     <div class = "home">
4.         <h1> my homepage </h1>
5.         <p> this is a test website </p>
6.     </div>
7. {% endblock %}
```

⇒ [also in about.html]

```
1. {% extends "layout.html" %}
2. {% block content %}
3.     <div class = "about">
4.         <h1> my aboutpage </h1>
5.         <p> this is a test website again </p>
6.     </div>
7. {% endblock %}
```

-> CSS styling (Demo -> (create static) -> (create CSS) -> (paste main.css)
[Next change user layout.html]

```
<!DOCTYPE html>
<html>
  <head> <title> FLASK APP </title>
    <link rel = "stylesheet" href = "{{ url_for ('static',
                filename = "CSS/main.css')}}"
  <head>
  <body>
    <header>
      <div class = "container">
#rest part of code remains same as before.
```

O/P -> # can be checked in localhost:5000.

-> Creating Python Virtual Environment

* Pip install virtualenv

[Capture -> Demo [where has all the files (script.py, layout.html)
-> create another new folder near to Demo. -> Mysite. -> and add paste Demo inside mysite.
-> Mouse open (Right click) VSC in Demo because venv has to be created here]

open terminal:-
-> Python -m venv virtual
[this creates folder].

-> virtual | scripts | python.
>>> ^Z (exits)

-> virtual | scripts | pip install flask
-> virtual | scripts | python Demo | app.py

https://git-scm.com/downloads

Deploying the website to a live server

1st step:
https://www.heroku.com
Signup → create an account
* then login

2nd step: heroku toolbelt. [Download]

3rd step: Open your folder Mysite → Demo →
→ [static, templates, script.py]. Right click
Cmd window
[login here to heroku]

4th step: > heroku login
Enter your heroku credentials:
Email:
Password:

5th step: [inorder to create your app]
→ create Any name
[this will create a app in heroku]

6th step: [To upload files need a gr].
pip freeze
[this will print all the packages]

7th step: .. \ virtual \ scripts \ pip freeze

8th step: pip install gunicorn
[this is http server]

9th step: .. \ virtual \ scripts \ pip freeze > requirements.txt
[this creates first file in our directory]

10th step: Create new file [Procfile] [Must in Demo]

open VSC in procfile.

1. web: gunicorn script1:app

step11:- Create one notepad file inside Demo
i.e runtime.txt
↓
and open in VSC or notepad
and write text
python-3.5.1 or -2.7.11

step12 cmd window > git init.

step13 git add .

step14 git commit -m "first"

step15 heroku git:remote --app Anyname.

step16 git push heroku master

If it can't recognise the repository then

step17: git config --global user.email "Email...@gmail.com"

step18: git config --global user.name Bindushri

step19. git push heroku master

tkinter -> script.py

Today

Section 21 | Graphical user interfaces with Tkinter

```
1. from tkinter import *
2. window = Tk()
3.
4. def km_to_miles():
5.     print(e1_value.get())
6.     miles = float(e1_value.get())*1.6
7.     t1.insert(END, miles)
8. b1 = Button(window, text="Execute", command=km_to_miles)
9. b1.grid(row=0, column=0)
10.
11. e1_value = StringVar()
12. e1 = Entry(window, textvariable=e1_value)
13. e1.grid(row=0, column=1)
14.
15. t1 = Text(window, height=1, width=20)
16. t1.grid(row=0, column=2)
17.
18. window.mainloop()
```

Database -> sqlite.py

Section 22 : Interacting with Databases

* Here require 2 libraries > sqlite3 > psycopg2

1. connect database 2. create a cursor obj 3. write queries 4. commit changes

```
1. import sqlite3
#  conn = sqlite3.connect("lite.db")
#  cur = ...
2. def create_table():
       conn = sqlite3.connect("lite.db")
       cur = conn.cursor()
       cur.execute("CREATE TABLE IF NOT EXISTS
                   store (item TEXT, quantity INTEGER,
                   price REAL)")
```

```
con.commit()
con.close()
def insert(item, quantity, price):
    conn = sqlite3.connect('lite.db')
    cur = conn.cursor()
    cur.execute("INSERT INTO store VALUES (?,?,?)",
                (item, quantity, price))
    conn.commit()
    conn.close()

insert("coffee cup", 10, 5)
def view():
    conn = sqlite3.connect("lite.db")
    cur = conn.cursor()
    cur.execute("SELECT * from store")
    rows = cur.fetchall()
    conn.close()
    return rows
print(view())
```