Fourier Transformation

March 24, 2021

[1]:
```
import numpy as np
import matplotlib.pyplot as plt
import sympy as sy
from numpy import pi
from scipy.fft import fft, ifft

from IPython.display import set_matplotlib_formats
set_matplotlib_formats('svg', 'pdf')
plt.style.use('seaborn')
plt.rc('figure', figsize=(15, 3))
```

1 To find the Fourier transforamtion for the following function

$$f(x) = \begin{cases} 1, & -1 < x < 0 \\ -1, & 0 < x < 1 \\ 0, & |x| > 1 \end{cases}$$

We have $g(x)$ as

$$g(x) = \frac{i}{\pi} \frac{\cos \alpha\pi - 1}{\alpha}$$

Thus the Fourier transforamtion of $g(x)$ is

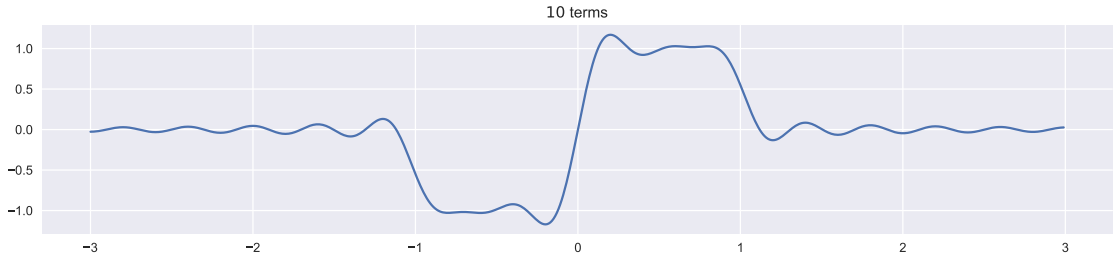
$$f(x) = \frac{i}{\pi} \int_{-\infty}^{\infty} g(x) e^{i\pi x} \, d\alpha$$

or for approximation

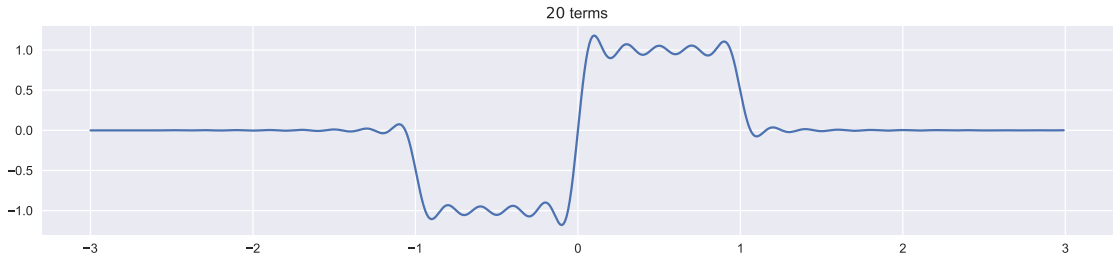
$$f(x) = \frac{i}{\pi} \int_{-\text{LIMIT}}^{\text{LIMIT}} g(x) e^{i\pi x} \, d\alpha$$

[2]:
```
def f_x(x, limit, da):
    result = 0
    for i in np.arange(-limit, limit, da):
        if i == 0 :
            result += 0
        else:
            result += (np.cos(pi * i) - 1) / i * np.exp(1j * i * x * pi) * da
```

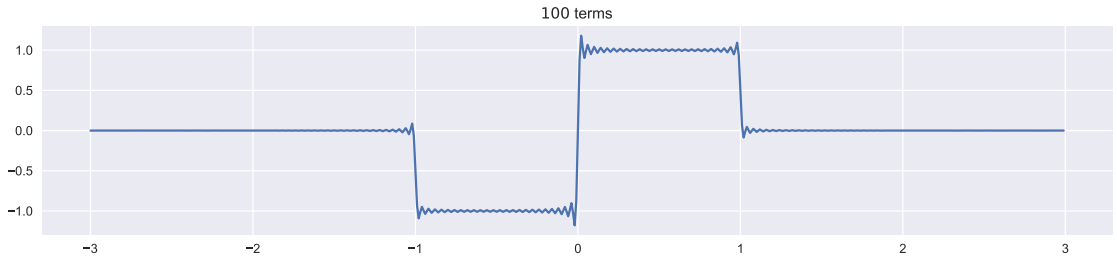
```
    return 1j / np.pi * result

def f2(x, limit, da):
    result = 0
    for i in np.arange(0, limit, da):
        if i == 0 :
            result += 0
        else:
            result += (np.sin(i) * np.cos(i * x)) / i * da

    return 2 / np.pi * result

LIMIT = 5
LIMIT_x = 3
da = 0.1
x = np.arange(-LIMIT_x, LIMIT_x, 0.01)
plt.plot(x, f_x(x, LIMIT, da).real)
plt.title('$10$ terms')
plt.show()
```



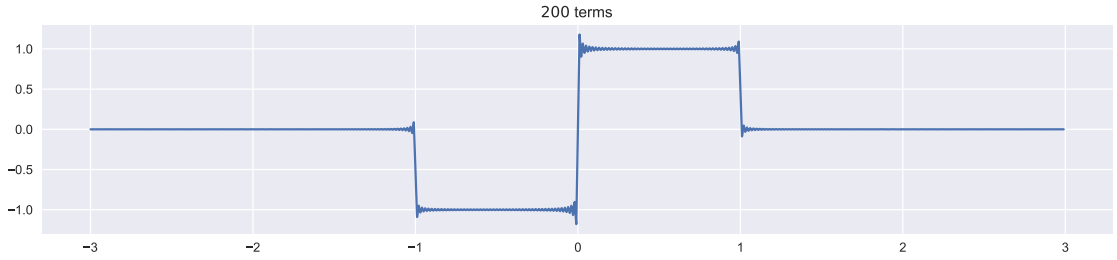
[3]:
```
LIMIT = 10
plt.plot(x, f_x(x, LIMIT, da).real)
plt.title('$20$ terms')
plt.show()
```



[4]:
```
LIMIT = 50
plt.plot(x, f_x(x, LIMIT, da).real)
plt.title('$100$ terms')
plt.show()
```



[5]:
```
LIMIT = 100
plt.plot(x, f_x(x, LIMIT, da).real)
plt.title('$200$ terms')
plt.show()
```



[6]:
```
LIMIT = 500
plt.plot(x, f_x(x, LIMIT, da).real)
plt.title('$1000$ terms')
plt.show()
```