fourier

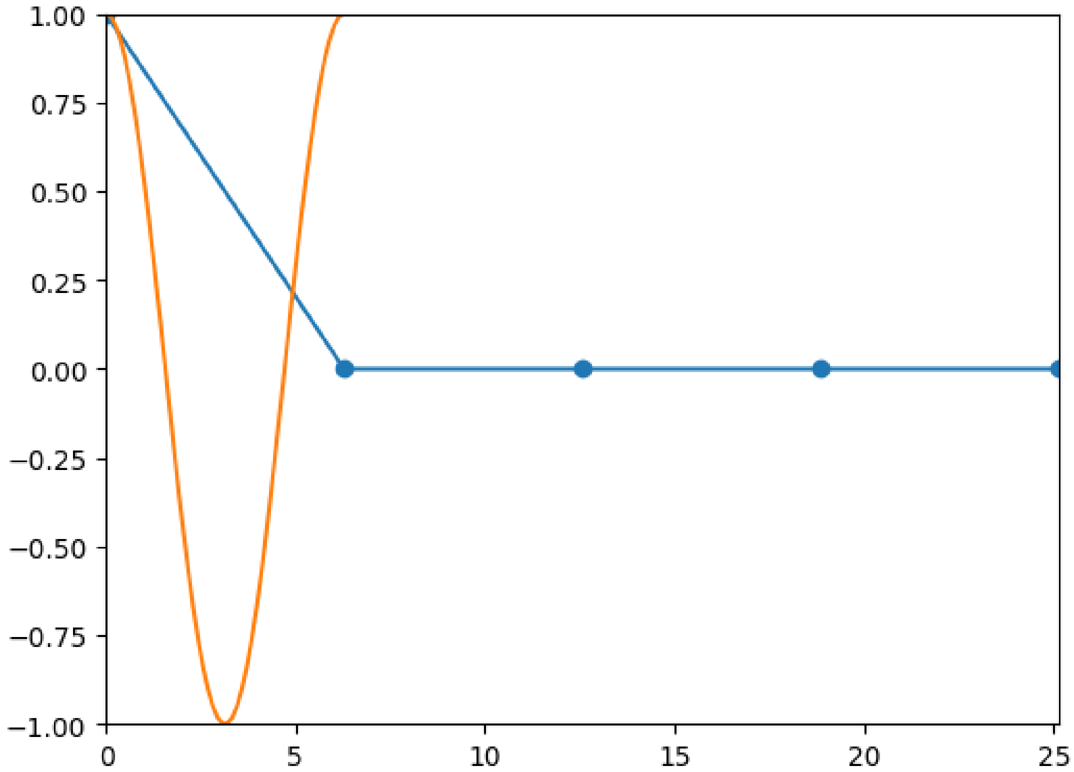
April 24, 2023

1 Fourier Transform Go Over With Diego

1.0.1 How many frequencies does a periodic function sampled at 1,2,3,4 points have?

1. With one sample we have exactly 1 frequency.

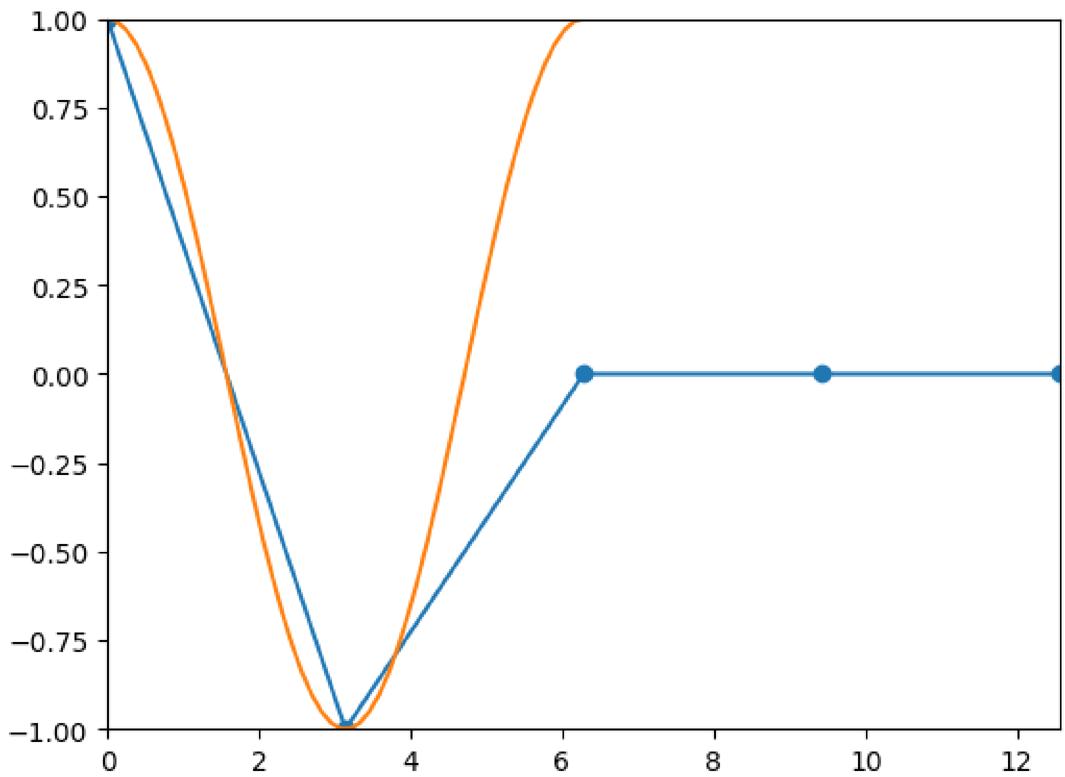
[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
# Period
T = np.pi * 2

# Domain
x = np.linspace(0,T * 4,5)
y = np.zeros_like(x)

# Sample only one point at x->0.
y[0] = np.cos(x[0])
plt.plot(x,y, marker = 'o')
plt.plot(np.linspace(0,T,),np.cos(np.linspace(0,T)))
plt.axis([0, float(8 * np.pi),-1,1])
plt.show()
```



In this case we have 1 frequency because there is just one single point to sample. Since we have only one frequency $cos(0) = 1$ and we are sampling only once then $s = [1]$.

2. With two samples we have two frequencies.

[ ]:
```
x = np.linspace(0,2 * T,5)
y = np.zeros_like(x)

y[0] = np.cos(x[0])
y[1] = np.cos(x[1])

plt.plot(x,y, marker = 'o')
plt.plot(np.linspace(0,T),np.cos(np.linspace(0,T)))
plt.axis([0,4 * np.pi,-1,1])
plt.show()
```

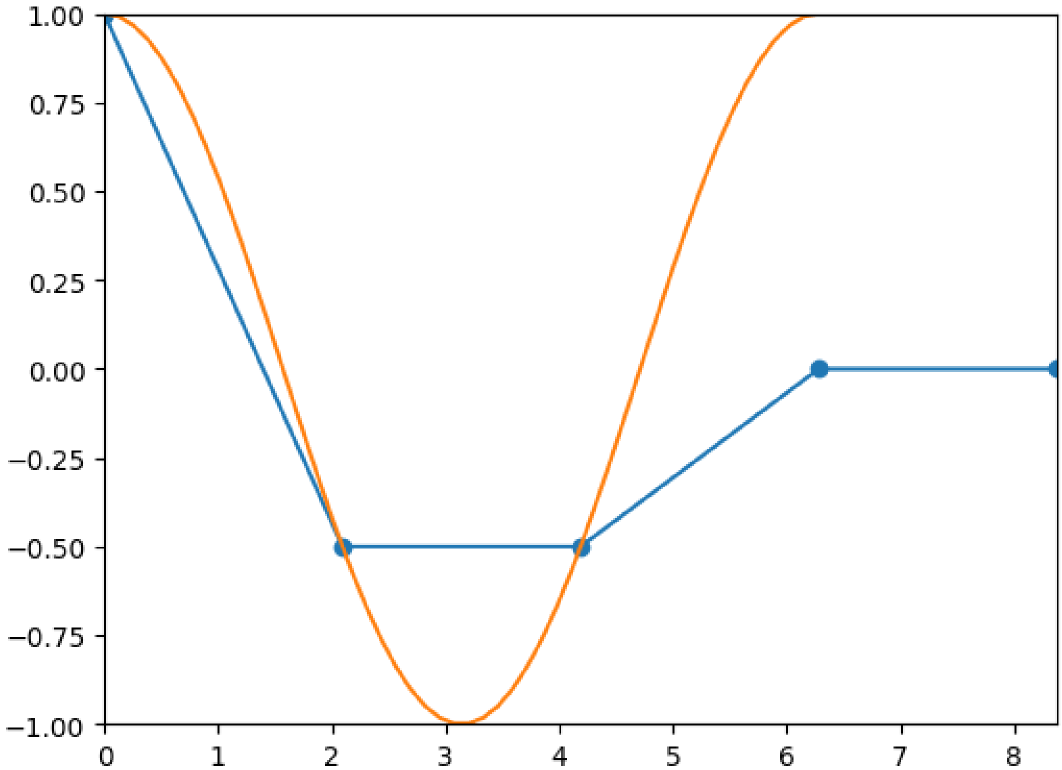


With $N = 2$ samples we have $cos(0) = 1$ and \$ cos( ) = -1\$, which are two frequencies since they have different value. $s = [1, -1]$.

3. With three sample points

[ ]:
```
x = np.linspace(0,T + (2 * np.pi)/ 3,5)
y = np.zeros_like(x)

y[0] = np.cos(x[0])
y[1] = np.cos(x[1])
y[2] = np.cos(x[2])
print(y[0])
print(y[1])
print(y[2])

plt.plot(x,y, marker = 'o')
plt.plot(np.linspace(0,T),np.cos(np.linspace(0,T)))
plt.axis([0,2 * np.pi + (2* np.pi) / 3,-1,1])
plt.show()
```

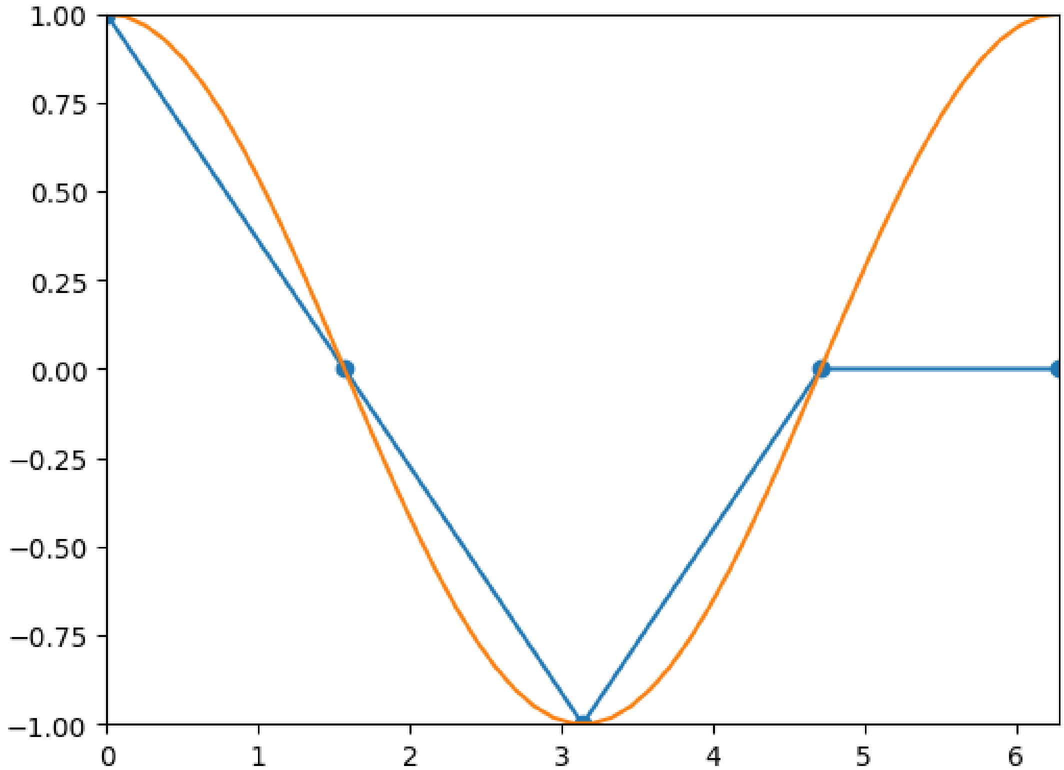
```
1.0
-0.4999999999999998
-0.5000000000000004
```



With $N = 3$ we have: $cos(0) = 1, cos(\frac{2\pi}{3}) = -\frac{1}{2}, cos(\frac{4\pi}{3}) = -\frac{1}{2}$.
By the cosine property: $cos(\pi + \alpha) = -cos(\alpha)$

4. With four points.

[ ]:
```
x = np.linspace(0,T,5)
y = np.zeros_like(x)

y[0] = np.cos(x[0])
y[1] = np.cos(x[1])
y[2] = np.cos(x[2])
y[3] = np.cos(x[3])

plt.plot(x,y, marker='o')
plt.plot(np.linspace(0,T),np.cos(np.linspace(0,T)))
plt.axis([0, 2 * np.pi,-1,1])
plt.show()
```



Wit $N = 4$ we have $cos(0) = 1, cos(\frac{\pi}{2}) = 0, cos(\frac{\pi}{2}) = -1, cos(\frac{\pi}{2}) = 0$
So in total 3 frequencies $s = [1, 0, -1, 0]$, 0 is repeated.

We can say then that:

$$n_{freq} = \lfloor \frac{N}{2} \rfloor + 1 \tag{1}$$

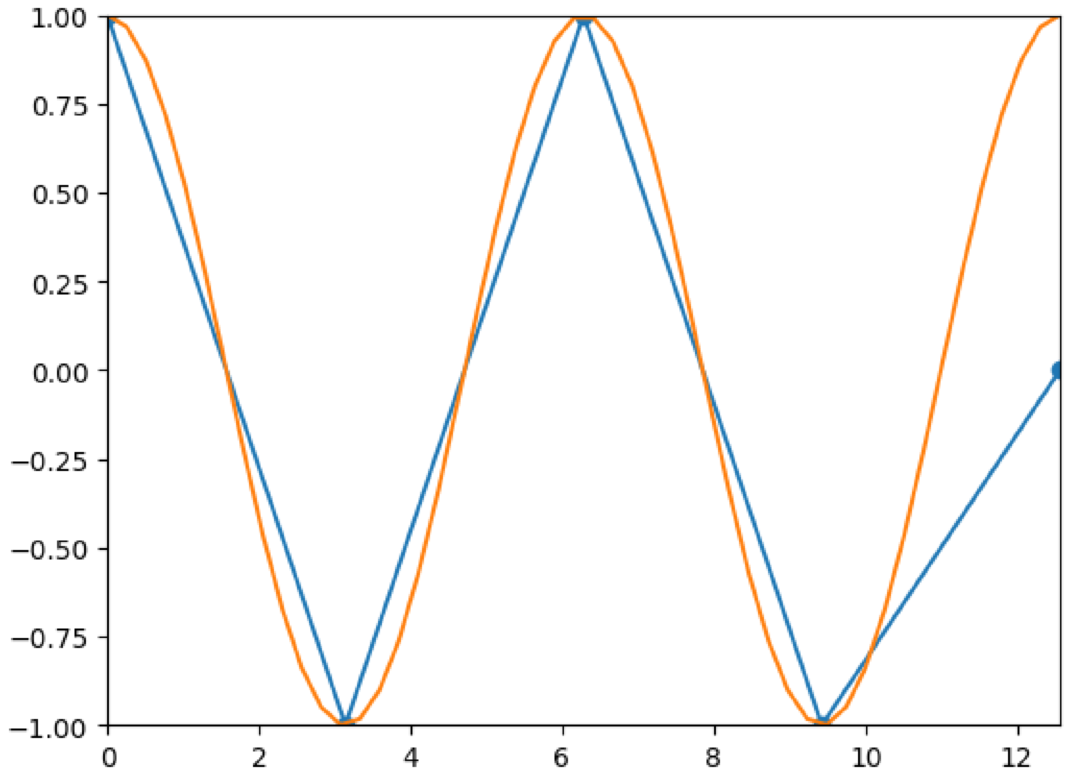
1.0.2 Example with $N = 4$ cos and period $T = 2\pi$

If our period is 2π and our $N = 4$ then we evaluate cos at intervals of $\frac{2\pi}{4}$, $\frac{2\pi}{4}k$
Given:

$$cos(\frac{2\pi}{N}mk) \tag{2}$$

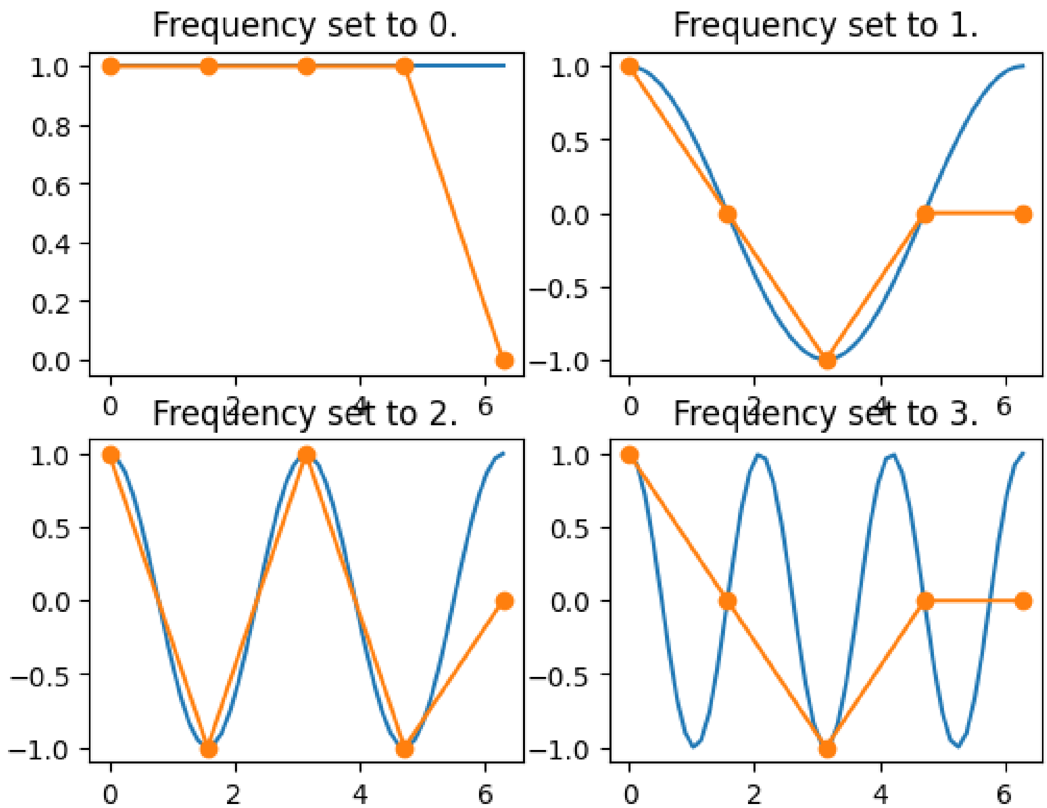
[ ]:
```
x = np.linspace(0,T * 2, 5)
y = np.zeros_like(x)

y[0] = np.cos(x[0])
y[1] = np.cos(x[1])
y[2] = np.cos(x[2])
y[3] = np.cos(x[3])

plt.plot(x,y, marker='o')
plt.axis([0, 4 * np.pi,-1,1])
```

```
plt.plot(np.linspace(0,T*2),np.cos(np.linspace(0,T*2)))
plt.show()
```



[ ]:
```
import matplotlib.animation as anim
x = np.linspace(0,T, 5)
y = np.zeros_like(x)
m = 0

x_cos = np.linspace(0,T)
y_cos = np.cos(x_cos * m)


y[0] = np.cos(x[0] * m)
y[1] = np.cos(x[1]* m)
y[2] = np.cos(x[2]* m)
y[3] = np.cos(x[3]* m)

figure, ax = plt.subplots(2,2)

ax[0, 0].plot(x_cos, y_cos)
ax[0, 0].plot(x, y, marker = 'o')
```

```
ax[0, 0].set_title("Frequency set to 0.")

m = 1
y_cos = np.cos(x_cos * m)
y[0] = np.cos(x[0] * m)
y[1] = np.cos(x[1]* m)
y[2] = np.cos(x[2]* m)
y[3] = np.cos(x[3]* m)

ax[0, 1].plot(x_cos, y_cos)
ax[0, 1].plot(x, y, marker = 'o')
ax[0, 1].set_title("Frequency set to 1.")
m = 2

y_cos = np.cos(x_cos * m)
y[0] = np.cos(x[0] * m)
y[1] = np.cos(x[1]* m)
y[2] = np.cos(x[2]* m)
y[3] = np.cos(x[3]* m)

ax[1, 0].plot(x_cos, y_cos)
ax[1, 0].plot(x, y, marker = 'o')
ax[1, 0].set_title("Frequency set to 2.")

m = 3
y_cos = np.cos(x_cos * m)
y[0] = np.cos(x[0] * m)
y[1] = np.cos(x[1]* m)
y[2] = np.cos(x[2]* m)
y[3] = np.cos(x[3]* m)

ax[1, 1].plot(x_cos, y_cos)
ax[1, 1].plot(x, y, marker = 'o')
ax[1, 1].set_title("Frequency set to 3.")

plt.show()
```



From these plots we can see that our y values repeat themselves at $f = 1$ and $f = 3$. Exaclty how we saw above when finding the number of frequencies.

We have Euler's Forumla which states:

$$e^{ix} = cos(x) + isin(x) \tag{3}$$

Which in our case gives us:

$$e^{i\frac{2\pi}{N}mk} = cos(\frac{2\pi}{N}mk) + isin(\frac{2\pi}{N}mk) \tag{4}$$

Again here we can evaluate the frequencies with $N = 4$ samples as: * $m = 0$ then: $e^{i\frac{2\pi}{4}0k} = cos(0) + isin(0) = 1 + i0$ for every k. * $m = 1$ then: $e^{i\frac{\pi}{2}k} = cos(\frac{\pi}{2}k) + isin(\frac{\pi}{2}k) = [1, i, -1, -i]$. * $m = 2$ then: $e^{i\pi k} = cos(\pi k) + isin(\pi k) = [1, -1, 1, -1]$. * $m = 3$ then: $e^{i\frac{3\pi}{2}k} = cos(\frac{3\pi}{2}k) + isin(\frac{3\pi}{2}k) = [1, i, -1, -i]$
Since for $ k = 0 →cos(0) +isin(0),$ for $k = 1 \rightarrow cos(\frac{3\pi}{2}) = isin(\frac{3\pi}{2}) = i, \ldots.$

So as we saw before 1 and 3 repeat themselves.

Now let's take this and apply it to a function $g(x)$

[ ]:
```
import cmath
# Trying something with cos function.
```

```python
def ft(N,k,m):
    return cmath.exp(-1j * (2 * np.pi)/N * k * m)


def ift(N,k,m):
    return cmath.exp(1j * (2 * np.pi)/N * k * m)

g_k = [0, np.pi /2, np.pi , 3/2 * np.pi, 2 * np.pi]
g_k_i = np.cos(g_k)

N = 4
# At
m = 0
output = np.zeros(int(np.floor(N/2)+1), dtype=complex)
for m in range(0, int(np.floor(N/2))):
    X_m = 0
    for i in range(0,N-1):
        X_m += g_k_i[i] * ft(N,i,m)
    output[m] = X_m


new_output = np.zeros(N + 1, dtype=complex)
for k in range(0, N + 1):
    X_k = 0
    for m in range(0,int(np.floor(N/2))):
        X_k += output[m] * ift(N,m,k)
    X_k = X_k / N
    new_output[k] = X_k
print(new_output)
# print(output)
# fig, ax = plt.subplots()
# ax.scatter(output.real, output.imag)
# ax.scatter([0,1,2], output.imag)
```

```
[ 5.00000000e-01+1.53080850e-17j  1.53080850e-17+5.00000000e-01j
 -5.00000000e-01+4.59242550e-17j -7.65404249e-17-5.00000000e-01j
  5.00000000e-01-1.07156595e-16j]
```