Christopher Lum
lum@uw.edu

Lecture06e
Discrete Fourier Transform

The YouTube video entitled 'Discrete Fourier Transform' that covers this lecture is located at https://youtu.be/W30U_rcThLg

References

Roots of Unity - https://ccrma.stanford.edu/~jos/mdft/Roots_Unity.html
Roots of Unity - https://en.wikipedia.org/wiki/Root_of_unity
Discrete Fourier Transform Derivation - https://www.dsprelated.com/freebooks/mdft/DFT_Derived.html
Discrete Fourier Transform Derivation - https://www.robots.ox.ac.uk/~sjrob/Teaching/SP/l7.pdf
Discrete Fourier Transform Derivation - https://youtu.be/FCTVIoBp7uw
Interpreting amplitudes from DFT - https://pythonnumericalmethods.berkeley.edu/notebooks/chapter24.02-Discrete-Fourier-Transform.html)

Introduction

We now look at a discrete representation of the fourier series. Somewhat confusingly, this is referred to as the **Discrete Fourier Transform** (DFT) but perhaps a more consistent name would be the Discrete Fourier Series as this is really just a discrete version of the continuous Fourier Series. In other words, we are generating functions that are periodic with period P as opposed to generating

Fourier Series $\Longleftrightarrow$ continuous representation of a function with period P
Fourier Transform $\Longleftrightarrow$ continuous representation of a function from $-\infty$ to ∞
Discrete Fourier Transform (DFT) $\Longleftrightarrow$ discrete representation of a function with period P

In many applications, instead of a continuous function, we are given a sequence of numbers which might represent a discrete sampling of a signal (for example, data logged by a computer at finite time steps). In this situation, we can look for an analogous analysis which might describe the frequency content of this discrete sampling of values. This DFT converts a finite list of equally spaced samples of

a function into a list of coefficients of a finite combination of complex sinusoids, order by their frequencies, that has those same sample values. This is effectively the discrete analogy of the formula for the coefficients of a (continuous) Fourier series.

Discrete Fourier Transform (DFT)

Roots of Unit

It seems trivial to mention that 1 raised to any power (including fractional powers) is still equal to 1

$$1^2 = 1^3 = 1^{\frac{1}{2}} = 1^{\frac{1}{3}} \ldots = 1^k = 1^{\frac{1}{N}} = 1^{\frac{k}{N}} = 1$$

In particular, we can refer to

$1^{\frac{1}{2}}$ = 2nd root of unity
$1^{\frac{1}{4}}$ = 4th root of unity
...
$1^{\frac{1}{N}}$ = N th root of unity

Recall from our lecture on complex numbers (see YouTube video entitled 'Complex Numbers, Complex Variables, and Complex Functions' at https://youtu.be/WEYX-wa9csU) that we can also write $e^{2\pi i} = 1 + 0\,i = 1$ so we can write

$$1^{\frac{1}{N}} = \left(e^{2\pi i}\right)^{\frac{1}{N}} = 1$$

As discussed above, this is true as well for any power

$$1^{\frac{k}{N}} = \left(e^{2\pi i}\right)^{\frac{k}{N}} = 1$$

This occurs often enough in mathematics that this has a special name and is referred to as the ***N*th root of of unity** https://en.wikipedia.org/wiki/Root_of_unity

$$\left(e^{2\pi i}\right)^{\frac{k}{N}} = e^{\frac{2\pi k i}{N}} \qquad (N^{\text{th}} \text{ root of unity}) \qquad \textbf{(Eq.2.1)}$$

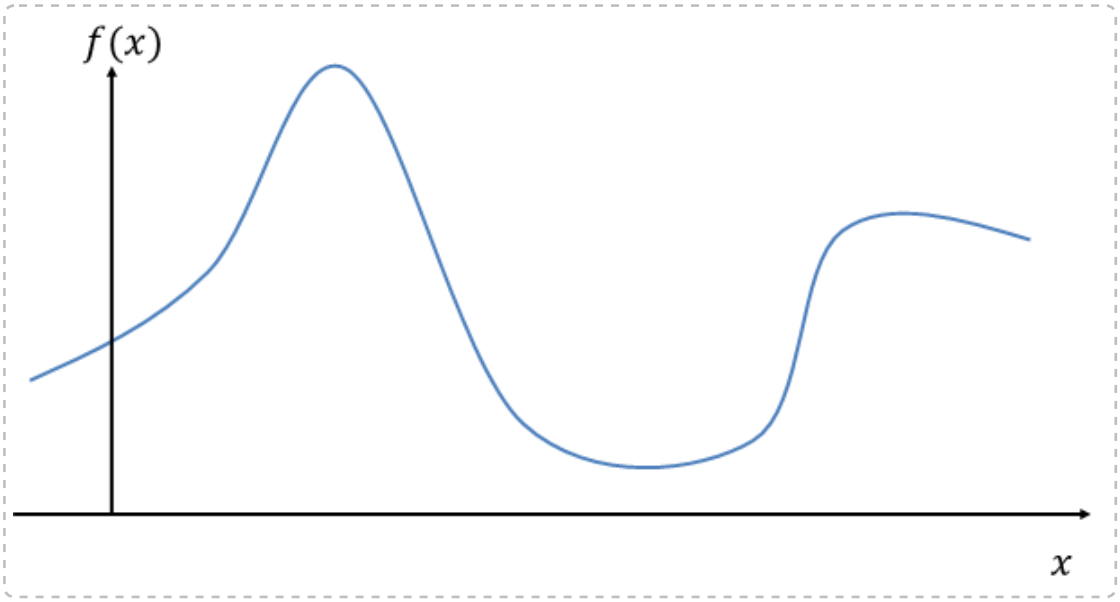
The special case where $k = 1$ is called the primitive N^{th}root of unity

$$\omega_N \triangleq \left(e^{2\pi i}\right)^{\frac{1}{N}} = e^{\frac{2\pi i}{N}} \qquad (\text{primitive } N^{\text{th}} \text{ root of unity}) \qquad \textbf{(Eq.2.2)}$$

In[•]:=
```
(*Use Nn because N is a reserved Mathematica symbol*)
ωN = Exp[2 π I / Nn];
```

Derivation of the DFT

Consider a signal as shown below



We can compute the Fourier transform of this signal using Eq.1.a (repeated here for convenience).

$$\hat{f}_1(\xi) = \int_{-\infty}^{\infty} f(x)\, e^{-2\pi i \xi x}\, dx \qquad \textbf{(Eq.1.a)}$$

However, instead of the entire continuous signal $f(x)$, suppose we only have N samples from the function.

$$f_0 = f(0 \cdot \Delta x)$$
$$f_1 = f(1 \cdot \Delta x)$$
$$f_2 = f(2 \cdot \Delta x)$$
$$\ldots$$
$$f_{N-1} = f((N-1) \cdot \Delta x)$$

In general

$$f_n = f(n\, \Delta x)$$

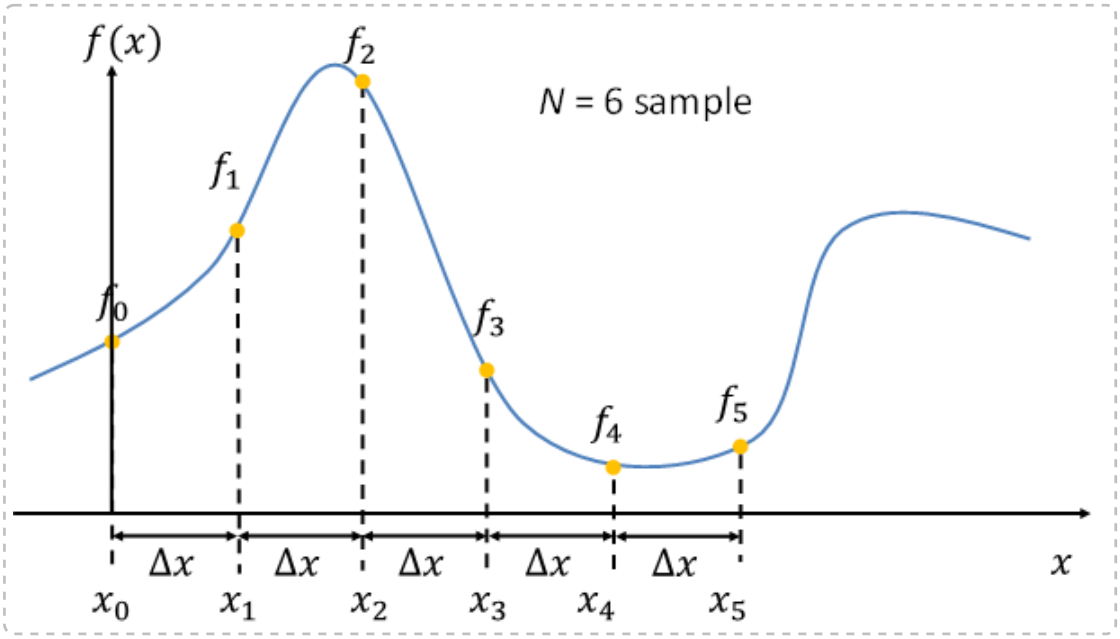
We can use the following notation to describe the sequence

$$\{f_n\} = \{f_0, f_1, \ldots, f_{N-1}\} \qquad \textbf{(Eq.3.1)}$$

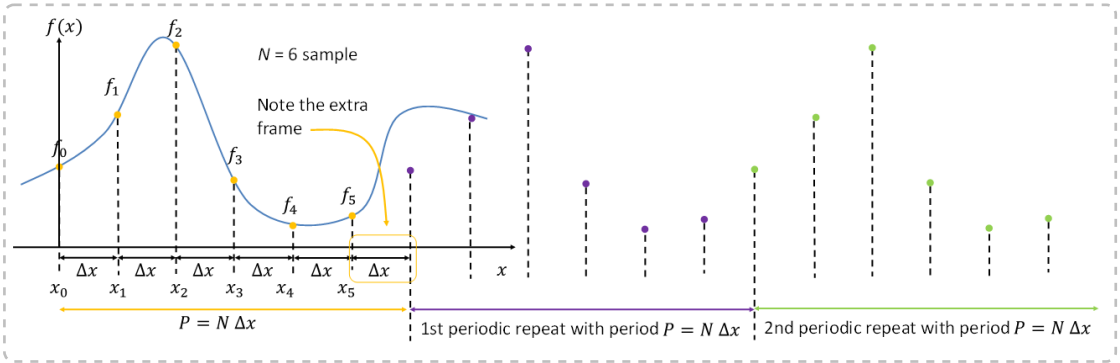
It is worthwhile to note that

n = index denoting sample in the sequence $\{f_n\}$

Without loss of generality, we assume that we start the sampling at $x = 0$. This is shown below as the orange dots.



We consider this sequence $\{f_n\}$ to be periodic as shown below



So we see that we are effectively considering the periodic discrete sequence $\{f_n\}$ with a period of

$$P = N\,\Delta x$$

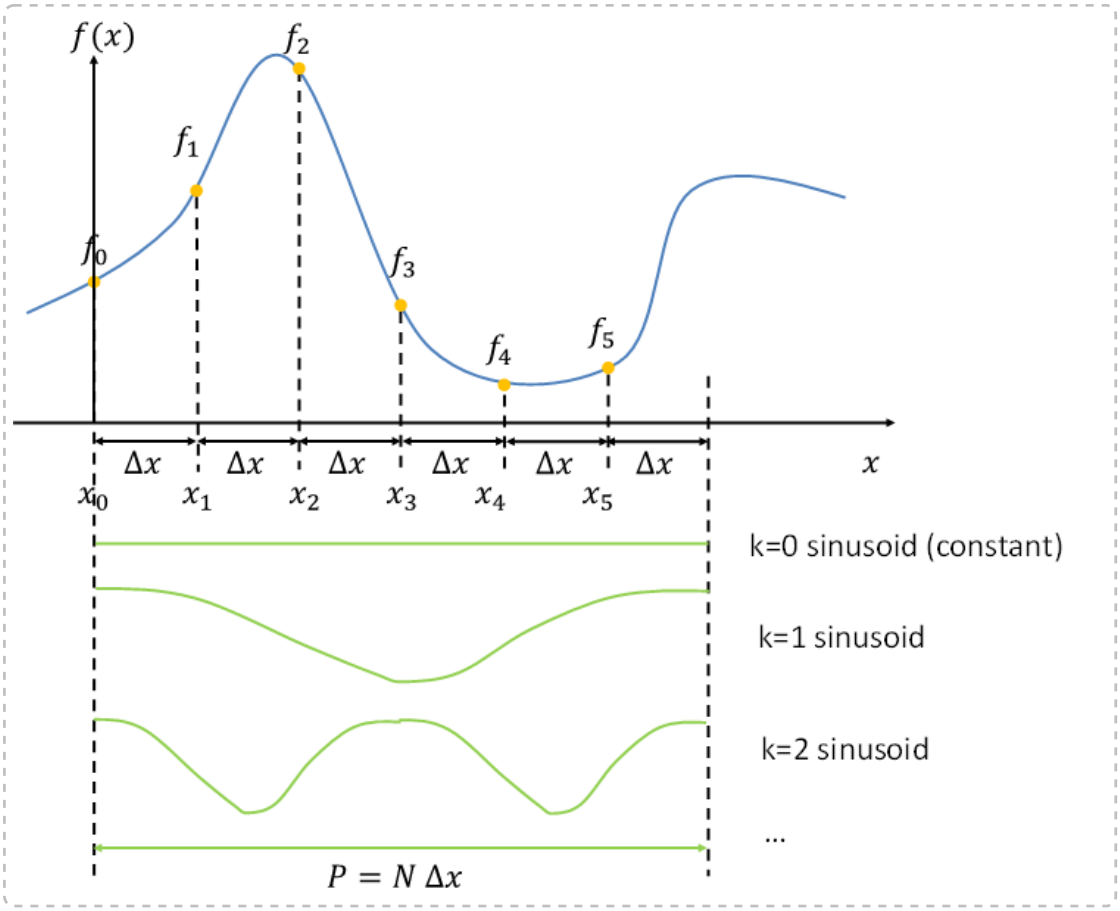
Note: from the figure above, we see that the period is actually $P = N\,\Delta x$, not $P = (N - 1)\,\Delta x$.

The lowest frequency sinusoid that we would consider fitting to this discrete sequence has 1 cycle over the period P. When written in

$$\xi = \frac{1\ \text{cycle}}{P\ \text{sec}} = \frac{1}{N\,\Delta x} \qquad \text{(ordinary frequency of in units of Hz)}$$

Or in terms of angular frequency we have

$$\omega = 2\,\pi\,\xi = \frac{2\,\pi}{N\,\Delta x}\ \text{(angular frequency in units of rad/s)}$$



We note that the real part of the k^{th} fundamental/basis sinusoid can be represented as

$$\text{Re}(s_k(x)) = \cos\left(\frac{2\pi}{P} k\, x\right) = \cos\left(\frac{2\pi}{N\Delta x} k\, x\right)$$

We can represent an equivalent complex sinusoid pair (with real and imaginary component) as

$$s_k(x) = e^{i\frac{2\pi}{P} k\, x}$$

$$= e^{i\frac{2\pi}{N\Delta x} k\, x}$$

$$= \left(e^{i\frac{2\pi}{N}}\right)^{\frac{k\, x}{\Delta x}} \qquad \text{recall: } \omega_N \triangleq \left(e^{2\pi i}\right)^{\frac{1}{N}} = e^{\frac{2\pi i}{N}}$$

$$s_k(x) = \omega_N^{\frac{k\, x}{\Delta x}} \qquad \textbf{(Eq.3.2.A)}$$

```
In[•]:= sk[x_] = ωN^(k x / Δx)
Out[•]= (E^(2 i π / Nn))^(k x / Δx)
```

In[•]:=
```
(*This should work for N=1 but for some reason Mathematica cannot simplify*)
Simplify[Exp[I 2 π / (Nn Δx) k x] == sk[x], Nn > 2]
```

Out[•]=
```
True
```

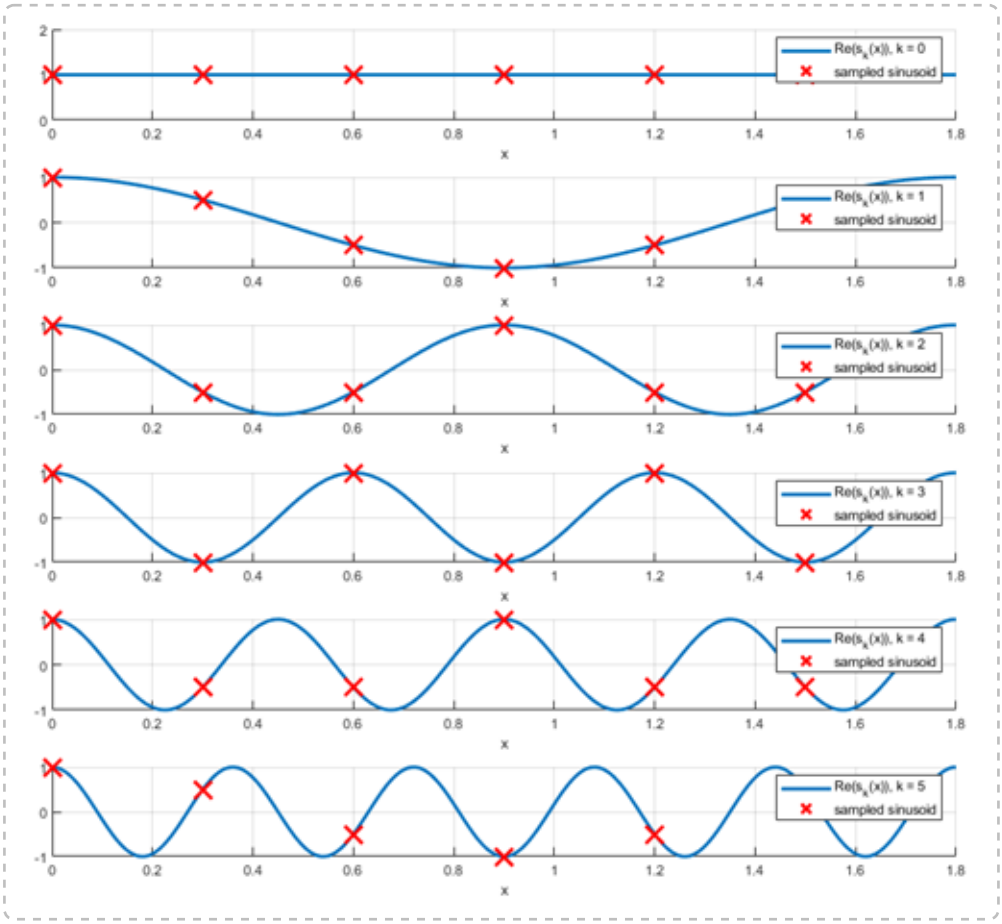
For completeness, we note that this can be expressed using sin and cos as

$$s_k(x) = \cos\left(\frac{2\pi k}{N\Delta x}\, x\right) + i \sin\left(\frac{2\pi k}{N\Delta x}\, x\right) \qquad \textbf{(Eq.3.2.B)}$$

In[•]:=
```
t1 = ExpToTrig[sk[x]];
t2 = Cos[2 π k / (Nn Δx) x] + I Sin[2 π k / (Nn Δx) x];
FullSimplify[t1 == t2, Nn > 2]
```

Out[•]=
```
True
```

However, instead of continuous sinusoids, $s_k(x)$, we instead samples these functions at the same sample times as $\{f_n\}$. Both the continuous and sampled sinusoids are shown below



We can now describe the sampled sinusoids by evaluating $s_k(x)$ at the points

$$x = \{0\cdot\Delta x,\ 1\cdot\Delta x,\ 2\cdot\Delta x,\ \ldots,\ (N-1)\cdot\Delta x\}$$

Evaluating for our example (recall that $s_k(x) = \omega_N^{\frac{k\,x}{\Delta x}}$)

Sample 1

$$s_k(0\cdot\Delta x) = \omega_N^{\frac{k\cdot 0\cdot\Delta x}{\Delta x}}$$

$$s_k(0\cdot\Delta x) = \omega_N^{0\cdot k}$$

Sample 2

$$s_k(1\cdot\Delta x) = \omega_N^{\frac{k\cdot 1\cdot\Delta x}{\Delta x}}$$

$$s_k(1\cdot\Delta x) = \omega_N^{1\cdot k}$$

Sample 3

$$s_k(2\cdot\Delta x) = \omega_N^{\frac{k\cdot 2\cdot\Delta x}{\Delta x}}$$

$$s_k(2\cdot\Delta x) = \omega_N^{2\,k}$$

Sample 4

$$s_k(3\cdot\Delta x) = \omega_N^{\frac{k\cdot 3\cdot\Delta x}{\Delta x}}$$

$$s_k(3\cdot\Delta x) = \omega_N^{3\,k}$$

Sample 5

$$s_k(4\cdot\Delta x) = \omega_N^{\frac{k\cdot 4\cdot\Delta x}{\Delta x}}$$

$$s_k(4\cdot\Delta x) = \omega_N^{4\,k}$$

Sample 6

$$s_k(5\cdot\Delta x) = \omega_N^{\frac{k\cdot 5\cdot\Delta x}{\Delta x}}$$

$$s_k(5\cdot\Delta x) = \omega_N^{5\,k}$$

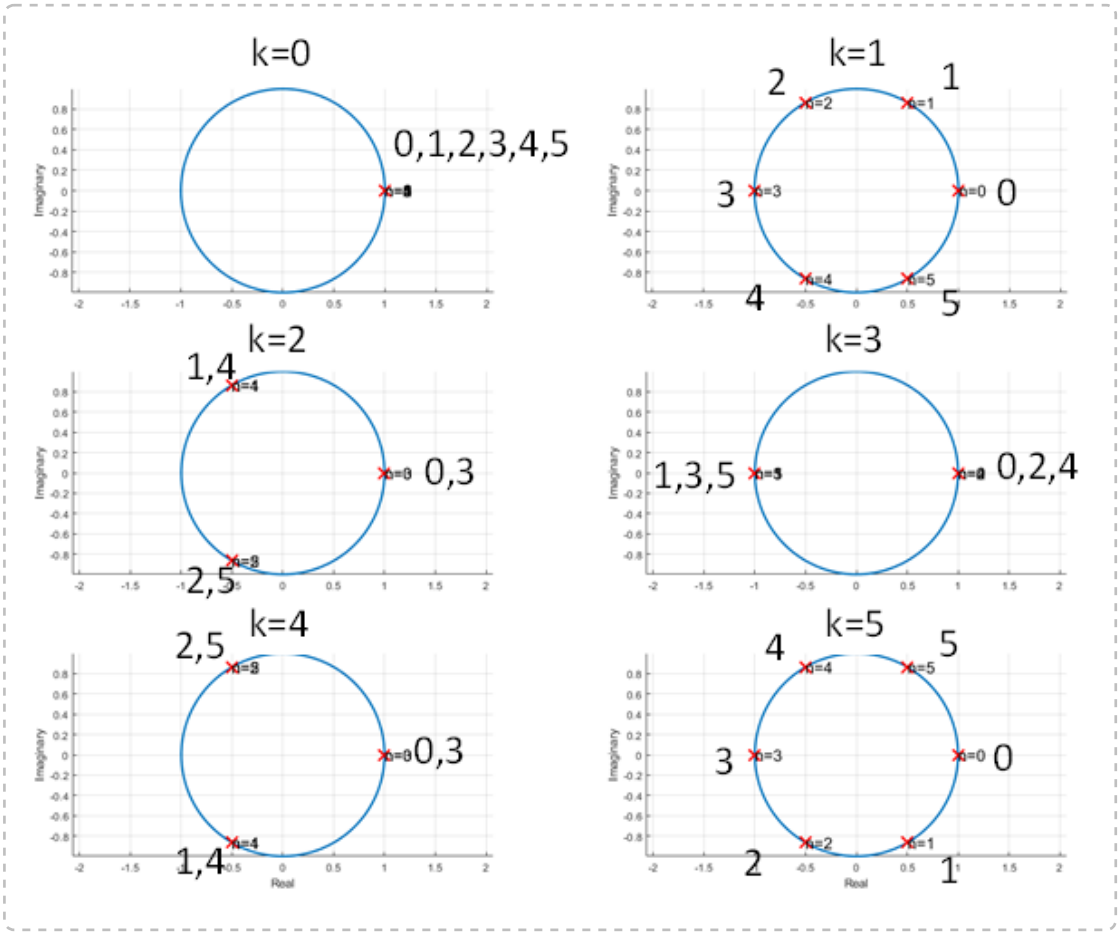
So in general we have

$$\{s_k(n)\} = \omega_N^{n\,k} = \left\{\omega_N^{0\cdot k},\ \omega_N^{1\cdot k},\ \ldots,\ \omega_N^{(N-1)\cdot k}\right\}$$

From a notation perspective, we note that k is the number of the fundamental/basis sinusoid whereas n is the number of the sample. In essence, $\{s_k(n)\}$ is the sequence of samples of the k^{th} basis sinusoid. Note that we can generate these samples from the N^{th} roots of unity by recalling the definition of $\omega_N = e^{\frac{2\pi i}{N}}$ so we have

$$\{s_k(n)\} = \omega_N^{n\,k} = e^{\frac{2\pi n k}{N} i} \qquad \textbf{(Eq.3.3)}$$

So we see that the n^{th} sample of the k^{th} basis sinusoid is simply a complex number with angle of $\frac{2\pi nk}{N}$. These samples in the complex plane are shown below



Recall from the discussion on Fourier Series that another way to think of the coefficients of the Fourier series is that they are the inner product of the function in question with orthogonal basis functions which are sinusoids of increasing frequency.

We now perform the same operation of calculating the coefficients of series by taking the inner product of the sequence $\{f_n\}$ with the orthogonal basis sinusoids of $\{s_k(n)\}$ to obtain the coefficients of the discrete sampled signal in the Fourier domain

$\hat{f}_k = <\{f_n\}, \{s_k(n)\}>$ note: recall that $\{s_k(n)\}$ is a vector of complex values so the inner product uses a complex conjugate

$$= \sum_{n=0}^{N-1} f_n \overline{s_k(n)}$$

$$= \sum_{n=0}^{N-1} f_n \overline{\omega_N^{nk}}$$

$$= \sum_{n=0}^{N-1} f_n \overline{e^{\frac{2\pi nk}{N} i}}$$

$$= \sum_{n=0}^{N-1} f_n \, e^{-\frac{2\pi n k}{N} i}$$

$$= \sum_{n=0}^{N-1} f_n \left(e^{\frac{2\pi}{N} i} \right)^{-n k}$$

$$\hat{f}_k = \sum_{n=0}^{N-1} f_n \, e^{-i\,2\pi k n/N} = \sum_{n=0}^{N-1} f_n \, \omega_N^{-n k} \qquad k = 0,\ 1,\ \ldots,\ N-1 \qquad \textbf{(Eq.3.4)}$$

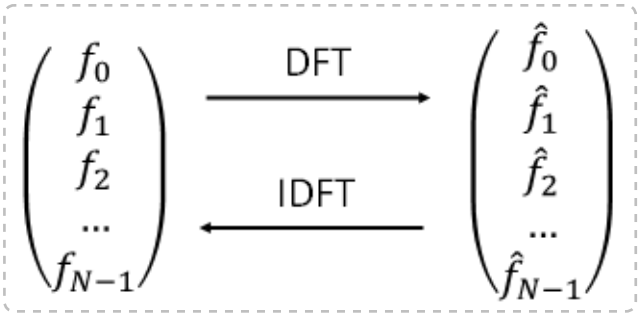
where $\omega_N = e^{2\pi i/N}$ (primitive N^{th} root of unity)

Eq.3.4 is referred to as the **Discrete Fourier Transform** (DFT). This transforms a of N numbers $f_0, f_1, \ldots, f_{N-1}$ into an N-periodic sequence of complex numbers $\hat{f}_0, \hat{f}_1, \ldots, \hat{f}_{N-1}$. Each value $\hat{f}_k$ represents how much of the basis sinusoid is present in the original sampled sequence $\{f_n\}$.

Similarly, the **Inverse Discrete Fourier Transform** (IDFT) is given as

$$f_n = \frac{1}{N} \sum_{k=0}^{N-1} \hat{f}_k \, e^{i\,2\pi k n/N} = \frac{1}{N} \sum_{k=0}^{N-1} \hat{f}_k \, \omega_N^{k n} \qquad n = 0,\ 1,\ \ldots,\ N-1 \qquad \textbf{(Eq.3.5)}$$

Note the normalizing factor of $\frac{1}{N}$ in the IDFT.



Example : Compute DFT By Hand With 6 samples

DFT

Consider a function

$$f(x) = 5 + 2\cos\left(2\pi x - \frac{\pi}{2}\right) + 3\cos(4\pi x)$$

Notice that this is composed of 3 signals

1. DC gain of 5
2. 1 Hz signal shifted by 90^o with amplitude of 2
3. 2 Hz signal with amplitude of 3

In[•]:=
```
f[t_] = 5 + 2 Cos[2 π t - π/2] + 3 Cos[4 π t];
```

Let us sample $f(t)$ at a rate of 5 Hz (which is more than twice as fast the highest frequency in the signal to satisfy the Nyquist criterion) and obtain $N = 6$ samples. In other words, the time spacing between samples is

$$\Delta x = 1/5$$

In[•]:=
```
Δx = 1 / 5;
```

We can now sample the signal with this temporal spacing

In[•]:=
```
(*How many elements will be in the list
 (note: we use the variable P because N is reserved in Mathematica*)
Nn = 6;

(*Create a container list of the appropriate size
 filled with zeros (note: use ConstantArray instead of Array)*)
fn = ConstantArray[0, Nn];
xn = ConstantArray[0, Nn];

(*Loop through and fill the array as appropriate*)
For[n = 0, n ≤ Nn - 1, n++,

 (*compute the current time*)
 currentX = n Δx;
 xn[[n + 1]] = currentX;
 fn[[n + 1]] = f[currentX];
]

(*clear loop variable*)
Clear[Nn, n, currentX]

Print["x"]
xn // N
Print[""]

Print["Samples"]
fn // N
```

x

Out[•]=
```
{0., 0.2, 0.4, 0.6, 0.8, 1.}
```

Samples

Out[•]=
```
{8., 4.47506, 7.10262, 4.75148, 0.670836, 8.}
```

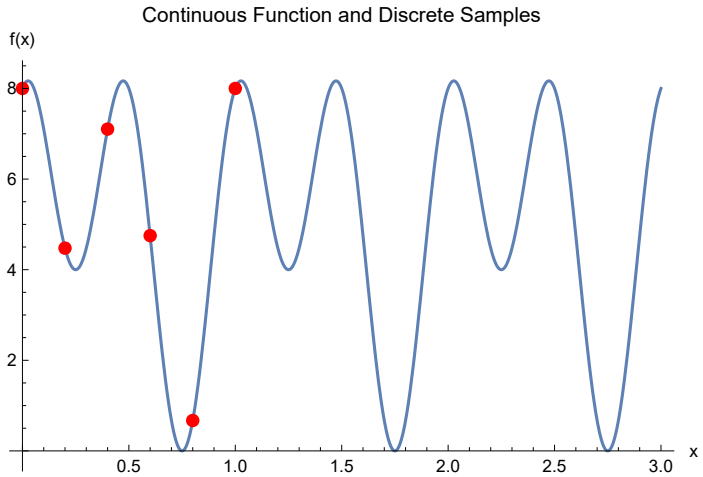
So we have

$\{f_n\} \approx \{8.0, 4.5, 7.1, 4.8, 0.7, 8.0\}$

We can now visualize the samples $\{f_n\}$ and the original function

In[•]:=
```
dataListPlot = Table[{xn[[n]], fn[[n]]}, {n, 1, Length[xn]}];
Show[
 (*Plot 1*)
 Plot[f[t], {t, 0, 3}],

 (*Plot 2*)
 ListPlot[dataListPlot,
  PlotStyle → {Red, PointSize[Large]}
 ],

 (*Plot Options*)
 PlotLabel → "Continuous Function and Discrete Samples",
 AxesLabel → {"x", "f(x)"}
]
```

Out[•]=



We can now compute the DFT using Eq.3.4 (repeated here for convenience)

$$\hat{f}_k = \sum_{n=0}^{N-1} f_n \, e^{-i\,2\pi\,k\,n/N} = \sum_{n=0}^{N-1} f_n \, \omega_N^{-n\,k} \qquad k = 0,\ 1,\ \ldots,\ N-1 \qquad \textbf{(Eq.3.4)}$$

where $\omega_N = e^{2\pi i/N}$ (primitive N^{th} root of unity)

We first compute the primitive N^{th} root of unity

$\omega_N = e^{2\pi i/N}$ (primitive N^{th} root of unity)

$= e^{2\pi i/6}$

$\omega_N = e^{\pi i/3}$

In[•]:= **ωNGiven = ωN /. {Nn → 6}**

Out[•]= $e^{\frac{i\pi}{3}}$

$\{f_n\} \approx \{8.0, 4.5, 7.1, 4.8, 0.7, 8.0\}$

For $k = 0$

$$\hat{f}_0 = \sum_{n=0}^{N-1} f_n \, \omega_N^{-n \cdot 0}$$

$$= f_0 \, \omega_N^{-0 \cdot 0} + f_1 \, \omega_N^{-1 \cdot 0} + f_2 \, \omega_N^{-2 \cdot 0} + f_3 \, \omega_N^{-3 \cdot 0} + f_4 \, \omega_N^{-4 \cdot 0} + f_5 \, \omega_N^{-5 \cdot 0}$$

$$= f_0 \cdot 1 + f_1 \cdot 1 + f_2 \cdot 1 + f_3 \cdot 1 + f_4 \cdot 1 + f_5 \cdot 1$$

$$= 8.0 \cdot 1 + 4.5 \cdot 1 + 7.1 \cdot 1 + 4.8 \cdot 1 + 0.7 \cdot 1 + 8.0 \cdot 1$$

$$\hat{f}_0 = 33$$

For $k = 1$

$$\hat{f}_1 = \sum_{n=0}^{N-1} f_n \, \omega_N^{-n \cdot 1}$$

$$= f_0 \, \omega_N^{-0 \cdot 1} + f_1 \, \omega_N^{-1 \cdot 1} + f_2 \, \omega_N^{-2 \cdot 1} + f_3 \, \omega_N^{-3 \cdot 1} + f_4 \, \omega_N^{-4 \cdot 1} + f_5 \, \omega_N^{-5 \cdot 1}$$

$$= f_0 \cdot 1 + f_1 \, \omega_N^{-1} + f_2 \, \omega_N^{-2} + f_3 \, \omega_N^{-3} + f_4 \, \omega_N^{-4} + f_5 \, \omega_N^{-5}$$

$$= 8.0 \cdot 1 + 4.5 \cdot \left(e^{\pi i/3}\right)^{-1} + 7.1 \cdot \left(e^{\pi i/3}\right)^{-2} + 4.8 \cdot \left(e^{\pi i/3}\right)^{-3} + 0.7 \cdot \left(e^{\pi i/3}\right)^{-4} + 8.0 \cdot \left(e^{\pi i/3}\right)^{-5}$$

$$\hat{f}_1 = 5.6 - 2.5\, i$$

For $k = 2$

$$\hat{f}_2 = \sum_{n=0}^{N-1} f_n \, \omega_N^{-n \cdot 2}$$

$$= f_0 \, \omega_N^{-0 \cdot 2} + f_1 \, \omega_N^{-1 \cdot 2} + f_2 \, \omega_N^{-2 \cdot 2} + f_3 \, \omega_N^{-3 \cdot 2} + f_4 \, \omega_N^{-4 \cdot 2} + f_5 \, \omega_N^{-5 \cdot 2}$$

$$= f_0 \cdot 1 + f_1 \, \omega_N^{-2} + f_2 \, \omega_N^{-4} + f_3 \, \omega_N^{-6} + f_4 \, \omega_N^{-8} + f_5 \, \omega_N^{-10}$$

$$= 8.0 \cdot 1 + 4.5 \cdot \left(e^{\pi i/3}\right)^{-2} + 7.1 \cdot \left(e^{\pi i/3}\right)^{-4} + 4.8 \cdot \left(e^{\pi i/3}\right)^{-6} + 0.7 \cdot \left(e^{\pi i/3}\right)^{-8} + 8.0 \cdot \left(e^{\pi i/3}\right)^{-10}$$

$$\hat{f}_2 = 2.6 + 8.6\, i$$

For $k = 3$

$$\hat{f}_3 = \sum_{n=0}^{N-1} f_n \, \omega_N^{-n \cdot 3}$$

$$= f_0 \, \omega_N^{-0 \cdot 3} + f_1 \, \omega_N^{-1 \cdot 3} + f_2 \, \omega_N^{-2 \cdot 3} + f_3 \, \omega_N^{-3 \cdot 3} + f_4 \, \omega_N^{-4 \cdot 3} + f_5 \, \omega_N^{-5 \cdot 3}$$

$$= f_0 \cdot 1 + f_1 \, \omega_N^{-3} + f_2 \, \omega_N^{-6} + f_3 \, \omega_N^{-9} + f_4 \, \omega_N^{-12} + f_5 \, \omega_N^{-15}$$

$$= 8.0 \cdot 1 + 4.5 \cdot \left(e^{\pi i/3}\right)^{-3} + 7.1 \cdot \left(e^{\pi i/3}\right)^{-6} + 4.8 \cdot \left(e^{\pi i/3}\right)^{-9} + 0.7 \cdot \left(e^{\pi i/3}\right)^{-12} + 8.0 \cdot \left(e^{\pi i/3}\right)^{-15}$$

$$\hat{f}_3 = -1.5$$

For $k = 4$

$$\hat{f}_4 = \sum_{n=0}^{N-1} f_n \, \omega_N^{-n \cdot 4}$$

$$= f_0 \, \omega_N^{-0 \cdot 4} + f_1 \, \omega_N^{-1 \cdot 4} + f_2 \, \omega_N^{-2 \cdot 4} + f_3 \, \omega_N^{-3 \cdot 4} + f_4 \, \omega_N^{-4 \cdot 4} + f_5 \, \omega_N^{-5 \cdot 4}$$

$$= f_0 \cdot 1 + f_1 \, \omega_N^{-4} + f_2 \, \omega_N^{-8} + f_3 \, \omega_N^{-12} + f_4 \, \omega_N^{-16} + f_5 \, \omega_N^{-20}$$

$$= 8.0 \cdot 1 + 4.5 \cdot \left(e^{\pi i/3}\right)^{-4} + 7.1 \cdot \left(e^{\pi i/3}\right)^{-8} + 4.8 \cdot \left(e^{\pi i/3}\right)^{-12} + 0.7 \cdot \left(e^{\pi i/3}\right)^{-16} + 8.0 \cdot \left(e^{\pi i/3}\right)^{-20}$$

$\hat{f}_4 = 2.6 - 8.6\,i$ (note this is complex conjugate of $\hat{f}_2$)

For $k = 5$

$$\hat{f}_5 = \sum_{n=0}^{N-1} f_n \, \omega_N^{-n \cdot 5}$$

$$= f_0 \, \omega_N^{-0 \cdot 5} + f_1 \, \omega_N^{-1 \cdot 5} + f_2 \, \omega_N^{-2 \cdot 5} + f_3 \, \omega_N^{-3 \cdot 5} + f_4 \, \omega_N^{-4 \cdot 5} + f_5 \, \omega_N^{-5 \cdot 5}$$

$$= f_0 \cdot 1 + f_1 \, \omega_N^{-5} + f_2 \, \omega_N^{-10} + f_3 \, \omega_N^{-15} + f_4 \, \omega_N^{-20} + f_5 \, \omega_N^{-25}$$

$$= 8.0 \cdot 1 + 4.5 \cdot \left(e^{\pi i/3}\right)^{-5} + 7.1 \cdot \left(e^{\pi i/3}\right)^{-10} + 4.8 \cdot \left(e^{\pi i/3}\right)^{-15} + 0.7 \cdot \left(e^{\pi i/3}\right)^{-20} + 8.0 \cdot \left(e^{\pi i/3}\right)^{-25}$$

$\hat{f}_5 = 5.6 + 2.5\,i$ (note this is complex conjugate of $\hat{f}_1$)

So we have

$$\hat{f}_k = \{33,\ 5.62 - 2.5\,i,\ 2.6 + 8.6\,i,\ -1.5,\ 2.6 - 8.6\,i,\ 5.6 + 2.5\,i\}$$

In[•]:=
```
(*Note: we need to be careful with indices as Mathematica is a 1-based system
   and therefore care must be taken to ensure the indexing is not off by 1*)
fhat = ConstantArray[0, Length[fn]];

For[k = 0, k ≤ Length[fn] - 1, k++,
 fhat[[k + 1]] = Sum[fn[[n + 1]] ωNGiven^-nk, {n, 0, Length[fn] - 1}]
]

Clear[k]
fhat // N
```

Out[•]=
```
{33., 5.59932 - 2.5174 i, 2.62722 + 8.62278 i, -1.45309, 2.62722 - 8.62278 i, 5.59932 + 2.5174 i}
```

Mathematica provides a function 'Fourier' to perform this calculation.

Note: The Mathematica Fourier function by default defines the DFT in a slightly different manner. Therefore, in order to match this definition, we need to user the FourierParameters argument (recall that we needed to do this with the FourierTransform function as well). You can use pass in FourierParameters as an additional argument to obtain the appropriate DFT

Fourier[fn, FourierParameters → {1, −1}] ⟺ match definition of DFT used here and also used by Matlab

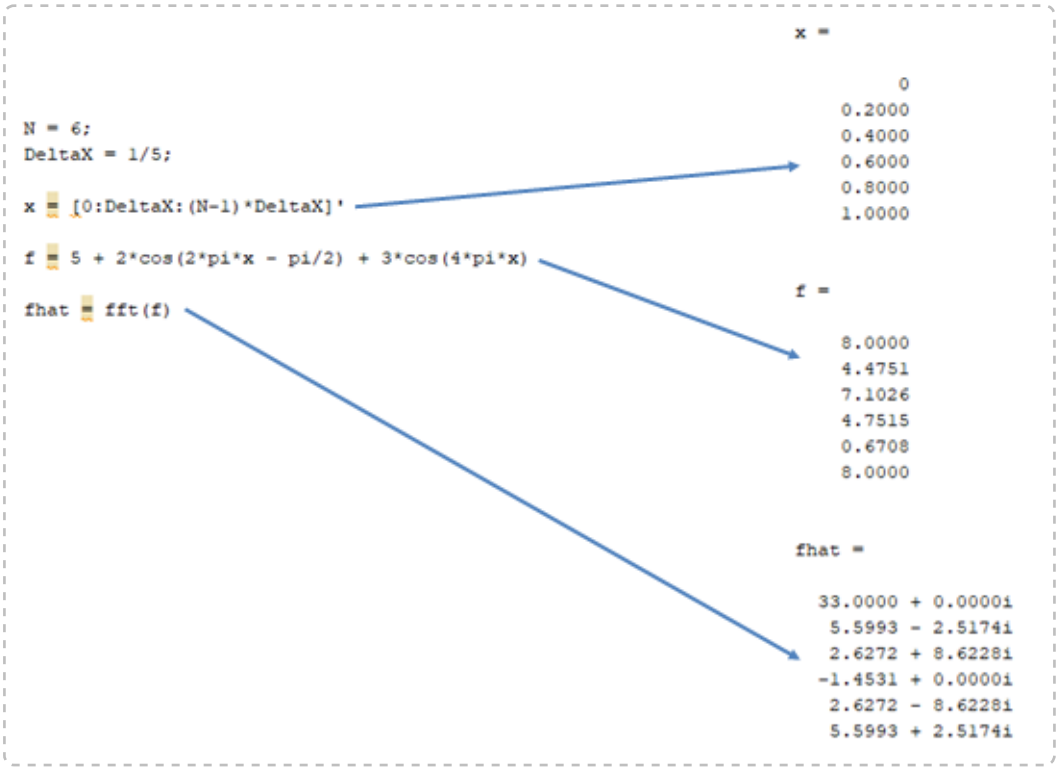
These same values need to be used for both forward and inverse transformations.

In[•]:=
```
fhatMathematica = Fourier[fn, FourierParameters → {1, -1}]

fhat == fhatMathematica
```

Out[•]=
```
{33. + 0. i, 5.59932 - 2.5174 i, 2.62722 + 8.62278 i,
 -1.45309 + 0. i, 2.62722 - 8.62278 i, 5.59932 + 2.5174 i}
```

Out[•]=
```
True
```

Matlab provides the fft function to compute the DFT. We can verify that Matlab provides the same output. We will discuss this more when we examine the Fast Fourier Transform.



Note that the absolute magnitude of the entries of $\hat{f}$ can grow larger simply by taking more samples. (show this in Matlab)

Interpretation of Results

Recall that this stemmed from the fact that we used continuous, complex basis sinusoids of the form described in Eq.3.2.A or equivalently Eq.3.2.B (repeated here for convenience)

$$s_k(x) = \omega_N^{\frac{kx}{\Delta x}} \qquad \textbf{(Eq.3.2.A)}$$

$$s_k(x) = \cos\left(\frac{2\pi k}{N\Delta x}\, x\right) + i \sin\left(\frac{2\pi k}{N\Delta x}\, x\right) \qquad \textbf{(Eq.3.2.B)}$$

Each $\hat{f}_k$ is the coefficients of the associated $s_k(x)$ (recall that $\hat{f}_k$ is a projection of the sampled signal onto the sampled version of the basis sinusoid described in Eq.3.2.B)

Each $\hat{f}_k$ is a complex number that encodes both amplitude and phase of the complex sinusoid that contributes to the overall signal. This sinusoid's frequency is $\frac{2\pi k}{N\,\Delta x}$ rad/s. Its amplitude, phase, and angular frequency are given by (https://pythonnumericalmethods.berkeley.edu/notebooks/chapter24.02-Discrete-Fourier-Transform.html)

$$A_k = \frac{|\hat{f}_k|}{N} = \sqrt{\mathrm{Re}(\hat{f}_k)^2 + \mathrm{Im}(\hat{f}_k)^2} \Big/ N \qquad \text{(amplitude)} \qquad \textbf{(Eq.3.5.A)}$$

$$\theta_k = \text{atan2}\left(\text{Im}(\hat{f}_k), \text{Re}(\hat{f}_k)\right) \qquad \text{(phase)} \qquad \textbf{(Eq.3.5.B)}$$

$$\omega_k = \frac{2\pi k}{N \Delta x} \qquad \text{(angular frequency, rad/s)} \qquad \textbf{(Eq.3.5.C)}$$

where N =number of samples
Δx =sample time
$k = 0, 1, 2, \ldots, N-1$

Note that the amplitude is normalized by N to take into account the fact that the magnitudes can grow as N grows.

Also note that we assume that atan2 takes input arguments in the order of y, x (see video entitled 'The 4 Quadrant Inverse Tangent (atan2) and Other Inverse Trigonometric Functions' at https://y-outu.be/UWrkh_N1bfE)

So roughly speaking, the approximation of the function which should have a similar frequency content as $\{f_n\}$ is given as

$$g(x) = \sum_{k=0}^{N-1} A_k \cos(\omega_k x + \theta_k) \qquad \textbf{(Eq.3.6)}$$

Note that we have combined the real and imaginary sinusoids into a single sinusoid with amplitude and phase shift.

In[ ]:=
```
Ak[a_, numSamples_] := Abs[a]/numSamples

θk[a_] := ArcTan[Re[a], Im[a]]   (*note order of input arguments,
Mathematica is different from aforementioned convention*)

ωk[k_, sampleTime_, numSamples_] := 2 π k/(numSamples * sampleTime)
```

Interpretation of Results

$k = 0$

In[ ]:=
```
k = 0;
Ak[fhat[[k + 1]], Length[fhat]] // N
θk[fhat[[k + 1]]] // N
ωk[k, Δx, Length[fhat]] // N
```

Out[ ]= 5.5

Out[ ]= 0.

Out[ ]= 0.

The first value $\hat{f}_1$ is the coefficient of the first fundamental sinusoid of frequency 0. In other words, this gives a measure of the DC gain

$$\text{DC gain} = \frac{1}{N}\hat{f}_0 = \frac{1}{6} \times 33 = 5.5$$

$$g_0(x) = 5.5$$

$k = 1$

```
In[ ]:= k = 1;
        Ak[fhat[[k + 1]], Length[fhat]] // N
        θk[fhat[[k + 1]]] // N
        ωk[k, Δx, Length[fhat]] // N
```

```
Out[ ]= 1.0232

Out[ ]= -0.422514

Out[ ]= 5.23599
```

So we see that there should be a component

$$g_1(x) = 1.0 \cos(5.2\, x - 0.4)$$

$k = 2$

```
In[ ]:= k = 2;
        Ak[fhat[[k + 1]], Length[fhat]] // N
        θk[fhat[[k + 1]]] // N
        ωk[k, Δx, Length[fhat]] // N
```

```
Out[ ]= 1.50236

Out[ ]= 1.27505

Out[ ]= 10.472
```

So we see that there should be a component

$$g_2(x) = 1.5 \cos(10.5\, x + 1.3)$$

$k = 3$

```
In[ ]:= k = 3;
        Ak[fhat[[k + 1]], Length[fhat]] // N
        θk[fhat[[k + 1]]] // N
        ωk[k, Δx, Length[fhat]] // N
```

```
Out[ ]= 0.242181

Out[ ]= 3.14159

Out[ ]= 15.708
```

So we see that there should be a component

$$g_3(x) = 0.2 \cos(15.7\, x + 3.1)$$

$k = 4$

```
In[•]:= k = 4;
        Ak[fhat[[k + 1]], Length[fhat]] // N
        θk[fhat[[k + 1]]] // N
        ωk[k, Δx, Length[fhat]] // N
```

```
Out[•]= 1.50236
```

```
Out[•]= -1.27505
```

```
Out[•]= 20.944
```

So we see that there should be a component

$$g_4(x) = 1.5 \cos(20.9\, x - 1.3)$$

$k = 5$

```
In[•]:= k = 5;
        Ak[fhat[[k + 1]], Length[fhat]] // N
        θk[fhat[[k + 1]]] // N
        ωk[k, Δx, Length[fhat]] // N
```

```
Out[•]= 1.0232
```

```
Out[•]= 0.422514
```

```
Out[•]= 26.1799
```

So we see that there should be a component

$$g_4(x) = 1.0 \cos(26.2\, x + 0.4)$$

So summing all these we have

```
In[•]:= g = 0;
        For[k = 0, k ≤ Length[fhat] - 1, k++,
         amplitude = Ak[fhat[[k + 1]], Length[fhat]];
         phaseAngle = θk[fhat[[k + 1]]];
         angularFrequency = ωk[k, Δx, Length[fhat]];

         g = g + amplitude * Cos[angularFrequency * x + phaseAngle];
        ]
        Clear[k]

        Print["g(x)"]
        g // N
```

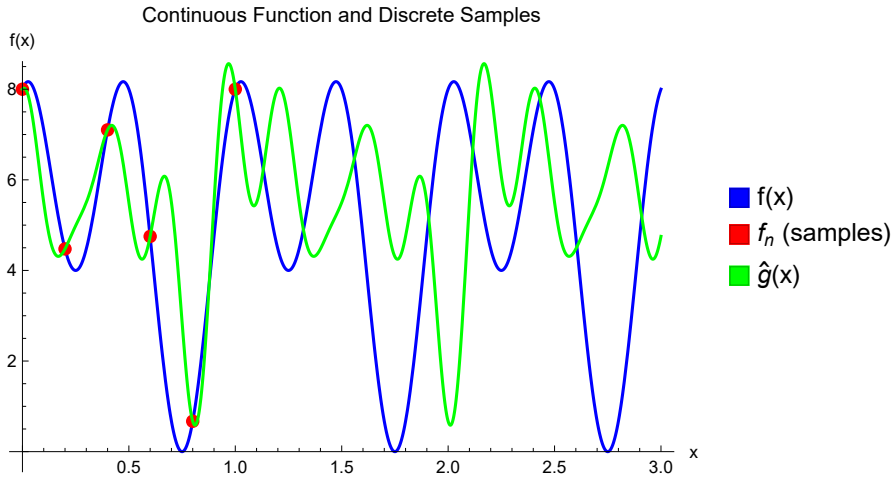
```
g(x)
```

```
Out[•]= 5.5 + 1.50236 Cos[1.27505 - 20.944 x] + 1.0232 Cos[0.422514 - 5.23599 x] -
         0.242181 Cos[15.708 x] + 1.50236 Cos[1.27505 + 10.472 x] + 1.0232 Cos[0.422514 + 26.1799 x]
```

Plotting this yields

In[ ]:=
```
Legended[
 Show[
  (*Plot 1: original function*)
  Plot[f[x], {x, 0, 3},
   PlotStyle → {Blue}],

  (*Plot 2: fₙ samples*)
  ListPlot[dataListPlot,
   PlotStyle → {Red, PointSize[Large]}
  ],

  (*Plot 3: g*)
  Plot[g, {x, 0, 3},
   PlotStyle → Green],

  (*Plot Options*)
  PlotLabel → "Continuous Function and Discrete Samples",
  AxesLabel → {"x", "f(x)"}
 ],

 (*Add the legend information*)
 SwatchLegend[{Blue, Red, Green}, {"f(x)", "fₙ (samples)", "ĝ(x)"}]
]
```

Out[ ]=



IDFT

We can now verify that the inverse transformation yields $\{f_n\}$

In[ ]:=
```
fMathematica = InverseFourier[fhat, FourierParameters → {1, -1}]

Print["Differences"]
fn - fMathematica

Print["Norm of different"]
Norm[fn - fMathematica]
```

Out[ ]= $\{8., 4.47506, 7.10262, 4.75148, 0.670836, 8.\}$

Differences

Out[ ]= $\{-1.77636 \times 10^{-15}, 6.66134 \times 10^{-16}, 1.33227 \times 10^{-15}, -4.44089 \times 10^{-16}, 1.9984 \times 10^{-15}, 0.\}$

Norm of different

Out[ ]= 3.09272×10^{-15}