BINGHAMTON UNIVERSITY
State University of New York

EECE 301
Signals & Systems
Prof. Mark Fowler

Note Set #20

- D-T Signals: Definition of DFT – Numerical FT Analysis

<u>Discrete Fourier Transform (DFT)</u>

We've seen that the DTFT is a good <u>analytical</u> tool for D-T signals (and systems – as we'll see later)

Namely $X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n}$ (DTFT) can be <u>computed analytically</u>

(at least in principle) when we have an <u>equation</u> model for $x[n]$

Q: **Well… why can't we use a <u>computer</u> to compute the DTFT <u>from Data</u>?**

A: There are <u>two reasons</u> why <u>we can't</u>!!

1. The DTFT requires an <u>infinite</u> number of terms to be summed over n = …, -3, -2, -1, 0, 1, 2, 3, …
2. The DTFT must be <u>evaluated</u> at an <u>infinite</u> number of points over the interval $\Omega \in (-\pi, \pi]$

-The first one ("infinite # of terms")… isn't a problem if $x[n]$ has "finite duration"

-The second one ("infinitely many points")… is always a problem!!

Well… <u>maybe</u> we can just compute the DTFT at a <u>finite</u> set of points!!

Let's explore this possibility… it will lead us to the **D**iscrete **F**ourier **T**ransform

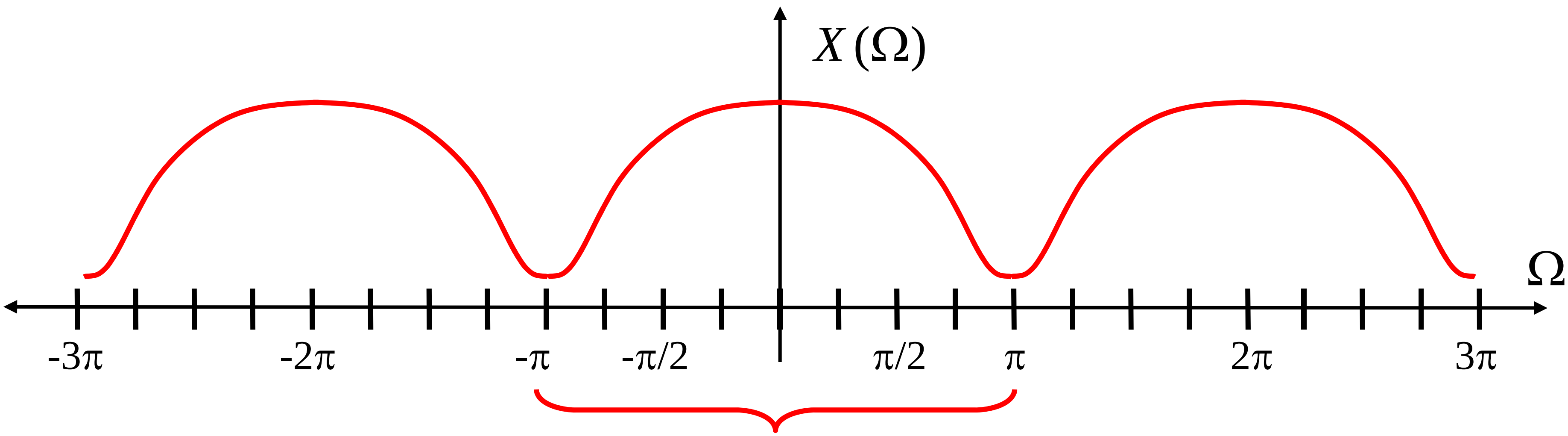
Suppose we have a finite duration signal: $x[n] = 0$ for $n < 0$ and $n \geq N$

Then the DTFT of this finite duration signal is:

$$X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n} = \sum_{n=0}^{N-1} x[n]e^{-j\Omega n}$$

we can leave out terms that are zero

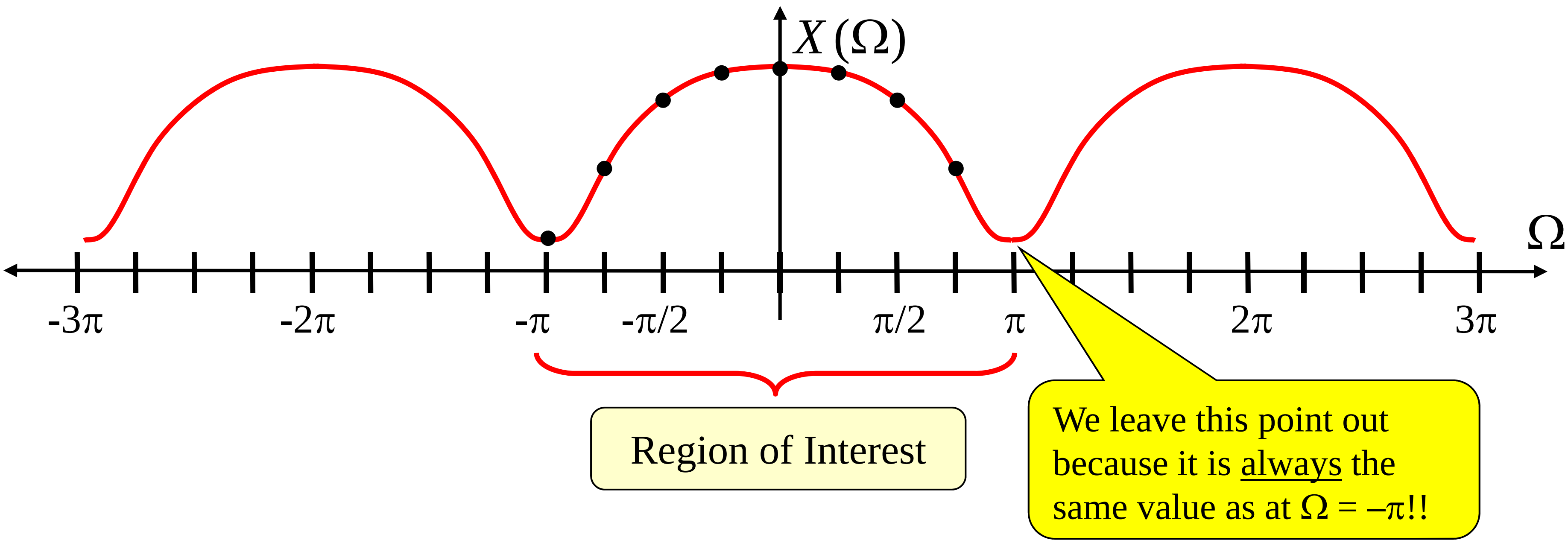
If we could compute this at every Ω value… it might look like this:



We are only interested in this range…
Everywhere else it just repeats periodically

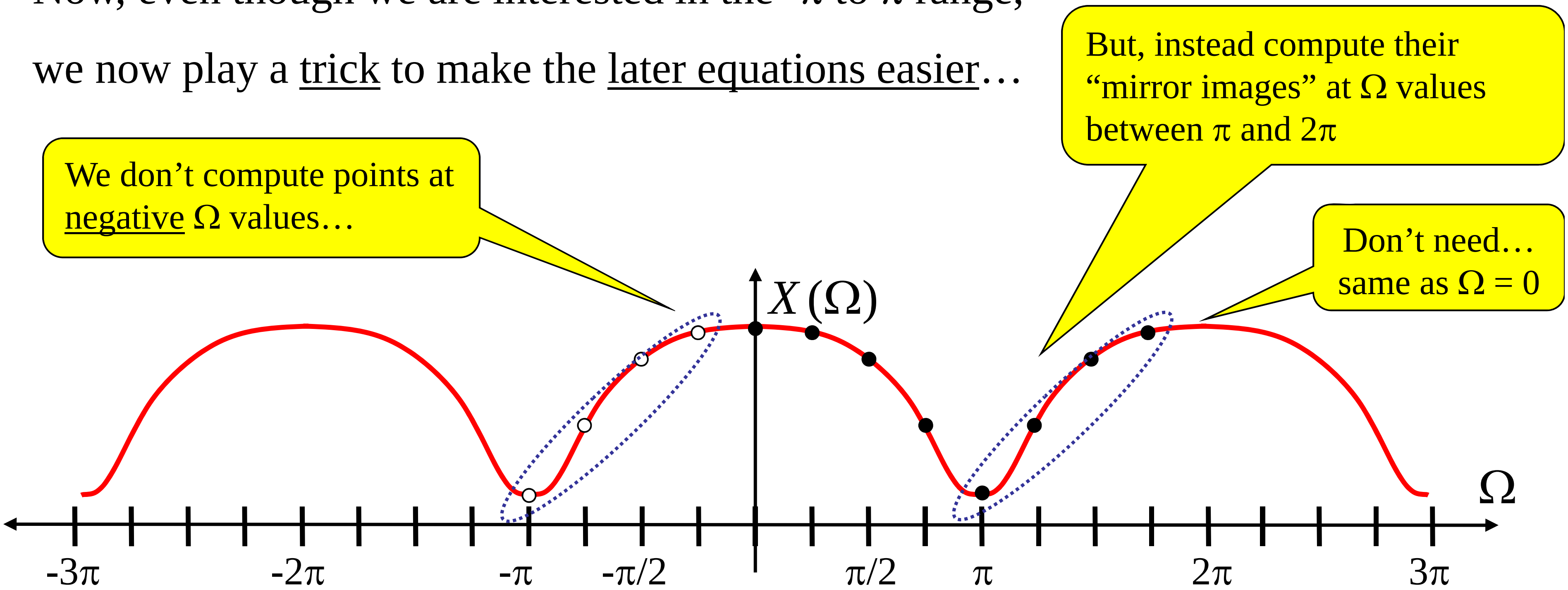
Now suppose we take the numerical data $x[n]$ for $n = 0, \ldots, N-1$

and just compute this DTFT at a <u>finite number of Ω values</u> (8 points here… but in practice we'd do it MANY more points… thousands of points!)



Now, even though we are interested in the -π to π range,

we now play a <u>trick</u> to make the <u>later equations easier</u>…



So say we want to compute the DTFT at M points, then choose

$$\Omega_k = k\boxed{\frac{2\pi}{M}}, \quad for \;\; k = 0,1,2,\ldots,M-1$$

Spacing between computed Ω values

In otherwords:

$$\boxed{\Omega_0 = 0,} \quad \boxed{\Omega_1 = \frac{2\pi}{M},} \quad \boxed{\Omega_2 = 2\frac{2\pi}{M},} \quad \ldots \quad \boxed{,\Omega_{M-1} = (M-1)\frac{2\pi}{M}}$$

Thus… mathematically what we have computed for our finite-duration signal is:

$$X(\Omega_k) = \sum_{n=0}^{N-1} x[n] e^{-jn\Omega_k} = \sum_{n=0}^{N-1} x[n] e^{-jnk\frac{2\pi}{M}}, \quad for \quad k = 0, 1, 2, \ldots, M-1$$

There is just one last step to get the "official" definition of

the **D**iscrete **F**ourier **T**ransform **(DFT):**

We must set $M = N$…

Done for a few mathematical reasons… later we'll learn a trick called "zero-padding" to get around this!

In other words: Compute as many "frequency points" as "signal points"

So… Given N signal data points $x[n]$ for $n = 0, \ldots, N$-1
Compute N DFT points using:

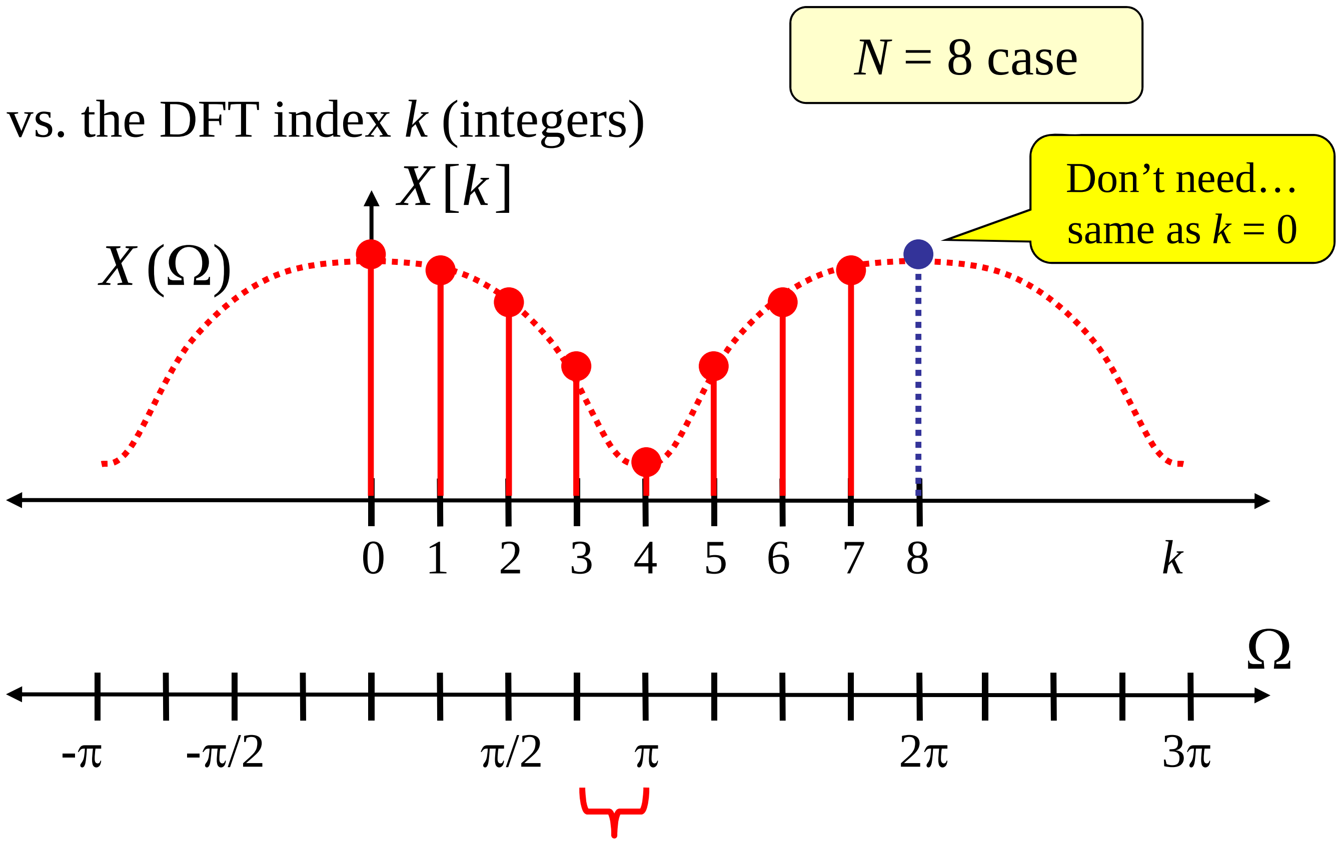
$$\boxed{X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N} \qquad k = 0, 1, 2, \ldots, N-1}$$

Definition of the DFT

$$\Omega_k = k\frac{2\pi}{N}$$

Plotting the DFT (we'll say more about this later..)

We often plot the DFT vs. the DFT index k (integers)



But… we know that these points can be tied back to the true D-T frequency Ω:

Spacing between computed Ω values

$$\frac{2\pi}{N} \Rightarrow \frac{2\pi}{8} = \frac{\pi}{4}$$

1. So far… we've defined the DFT
   a. We based this on the motivation of wanting to compute the DTFT at a finite number of points
   b. Later… we'll look more closely at the general relationship between the DFT and the DTFT
2. For now… we want to understand a few properties of the DFT
   a. There a more properties… if you take a DSP class you'll learn them there.

Properties of the DFT

1. **Symmetry of the DFT**

We arrived at the DFT via the DTFT so it should be no surprise that the DFT inherits some sort of symmetry from the DTFT.

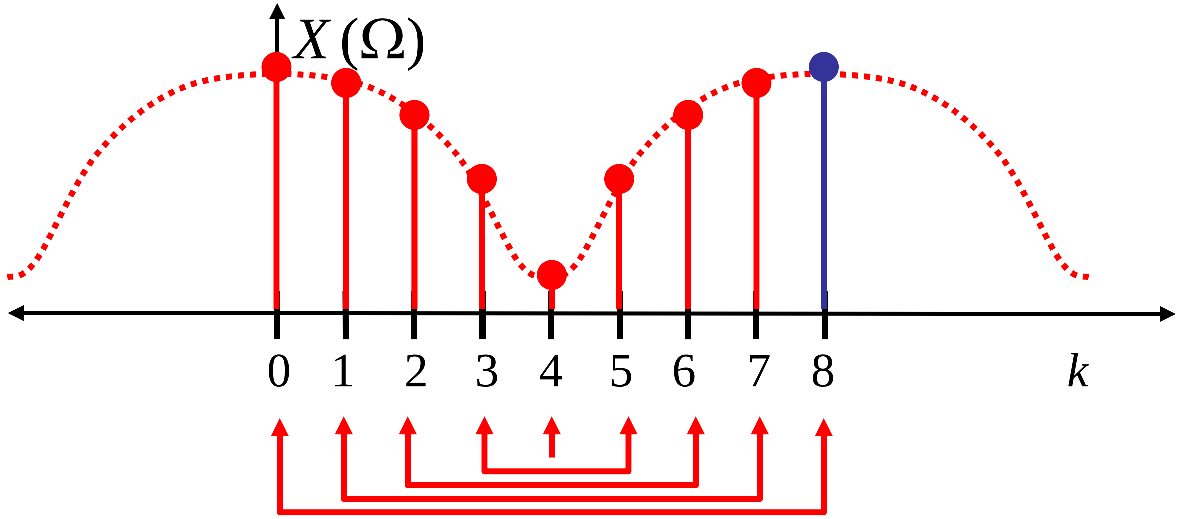
$$\boxed{X[N-k] = \overline{X}[k], \quad k = 0,1,2,\ldots,N-1}$$

Illustration of DFT Symmetry

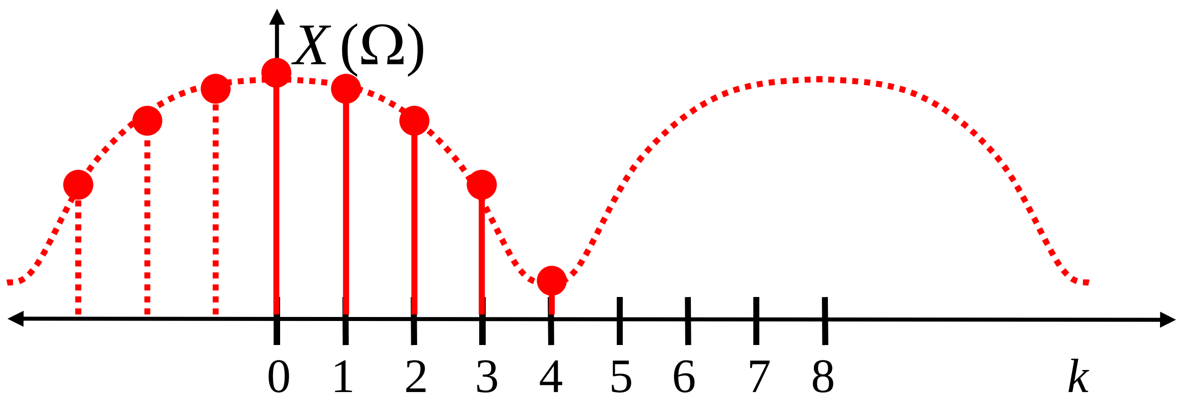
$$\boxed{X[N-k] = \overline{X}[k], \quad k = 0,1,2,\ldots,N-1}$$

In this example we don't see the effect of the conjugate because we made the DFT real-valued for ease

$N = 8$ case



Because the "upper" DFT points are just like the "negative index" DFT points… this DFT symmetry property is exactly the same as the DTFT symmetry around the origin:



2. **<u>Inverse DFT</u>**

Recall that the DTFT can be inverted… given $X(\Omega)$ you can find the signal $x[n]$

Because we arrived at the DFT via the DTFT… it should be no surprise that the DFT inherits an inverse property from the DTFT.

Actually, we needed to force $M = N$ to enable the DFT inverse property to hold!!

So… Given N DFT points $X[k]$ for $k = 0, \ldots, N-1$
Compute N signal data points using:

$$x[n] = \frac{1}{N}\sum_{n=0}^{N-1} X[k]e^{j2\pi kn/N} \qquad n = 0,1,2,\ldots,N-1$$

Inverse DFT (IDFT)

Compare to the DFT… a remarkably similar structure:

$$X[k] = \sum_{n=0}^{N-1} x[n]e^{-j2\pi kn/N} \qquad k = 0,1,2,\ldots,N-1$$

DFT

FFT Algorithm (We won't delve into HOW/WHY this works)

"FFT" = **F**ast **F**ourier **T**ransform

The FFT is not a new "thing" to compute (the DFT is a "thing" we compute)

The FFT is just an efficient algorithm for computing the DFT

If you code the DFT algorithm in the obvious way it takes:

about N^2 multiples, and N^2 additions.

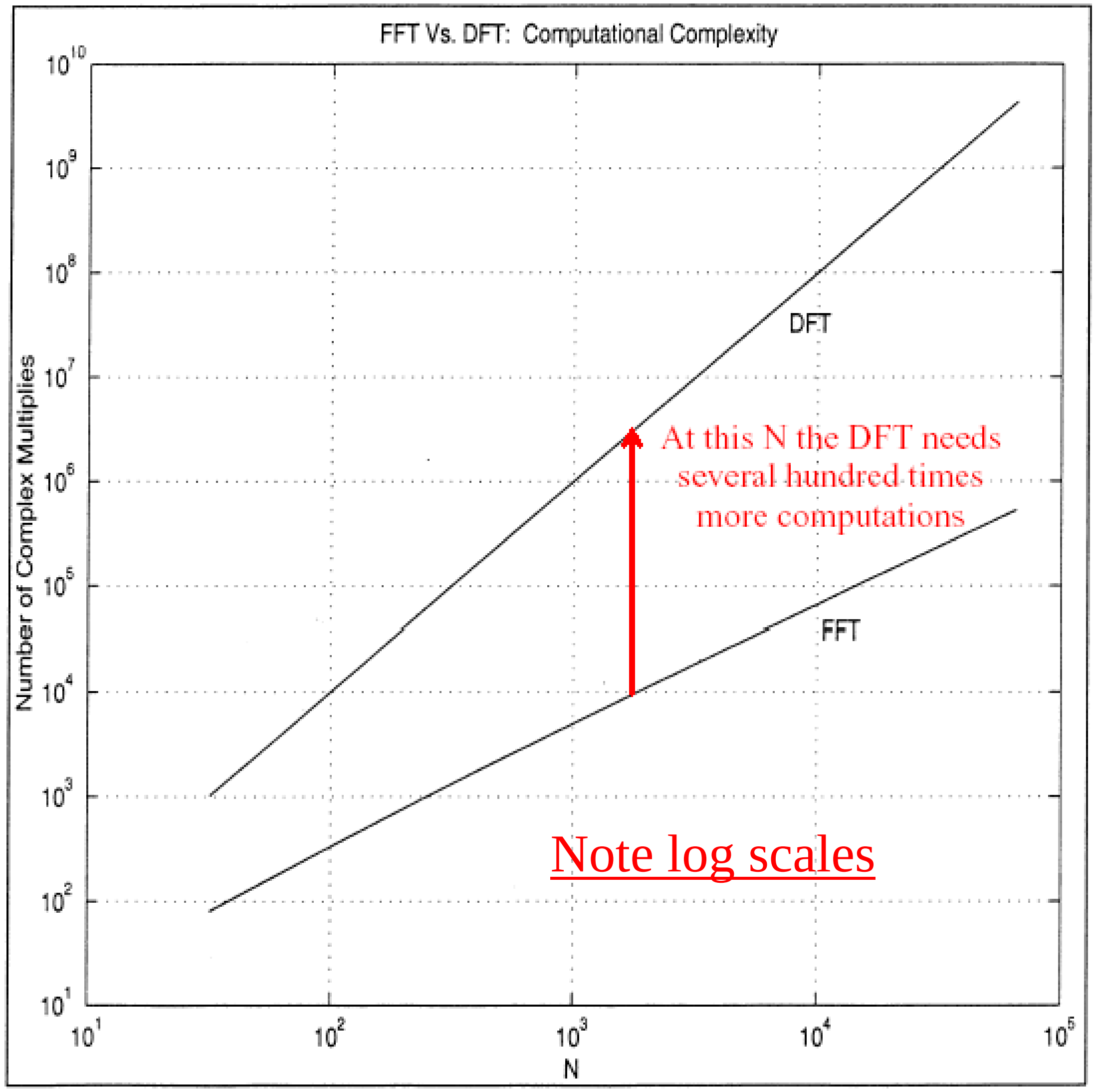
The FFT is a trick to compute the DFT more efficiently – it takes:

$$\text{about} \begin{cases} \dfrac{N}{2}\log_2(N) & \textit{multiples} \\ \dfrac{N}{2}\log_2(N) & \textit{additions} \end{cases}$$

Similar ideas hold for computing the inverse DFT.

Note: The most common FFT algorithm requires that the number of samples put into it be a power of two… later we'll talk about how to "zero-pad" up to a power of two!

The following plot shows the **drastic** improvement the FFT gives over the DFT:



DFT Summary… What We Know So Far!

- Given N signal data points… we can compute the DFT
  - And we can do this efficiently using the FFT algorithm
- Given N DFT points… we can get back the N signal data points
  - And we can do this efficiently using the IFFT algorithm
- We know that there is a symmetry property
- We know that we can move the "upper" DFT points down to represent the "negative" frequencies…
  - this will be essential in practical uses of the DFT
  - Remember… we ended up with the "upper" DFT points only to make the indexing by k easy!!!
    - It is just to make the DFT equation easy to write!!

Now…

• We need to explore the connections between the DFT and the DTFT

• Then… understand the relation between the CTFT, DTFT, & DFT