EEO 401
Digital Signal Processing

Prof. Mark Fowler

Note Set #18

- Introduction to DFT (via the DTFT)
- Reading Assignment: Sect. 7.1 of Proakis & Manolakis

<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

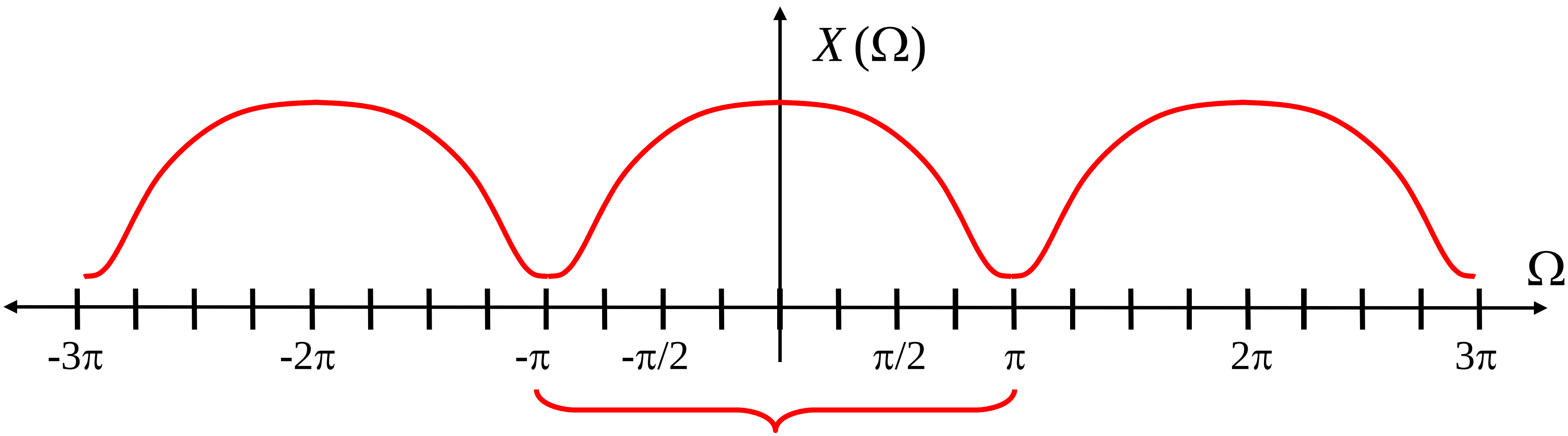
Suppose we have a finite duration signal: $x[n] = 0$ for $n < 0$ and $n \geq N$

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

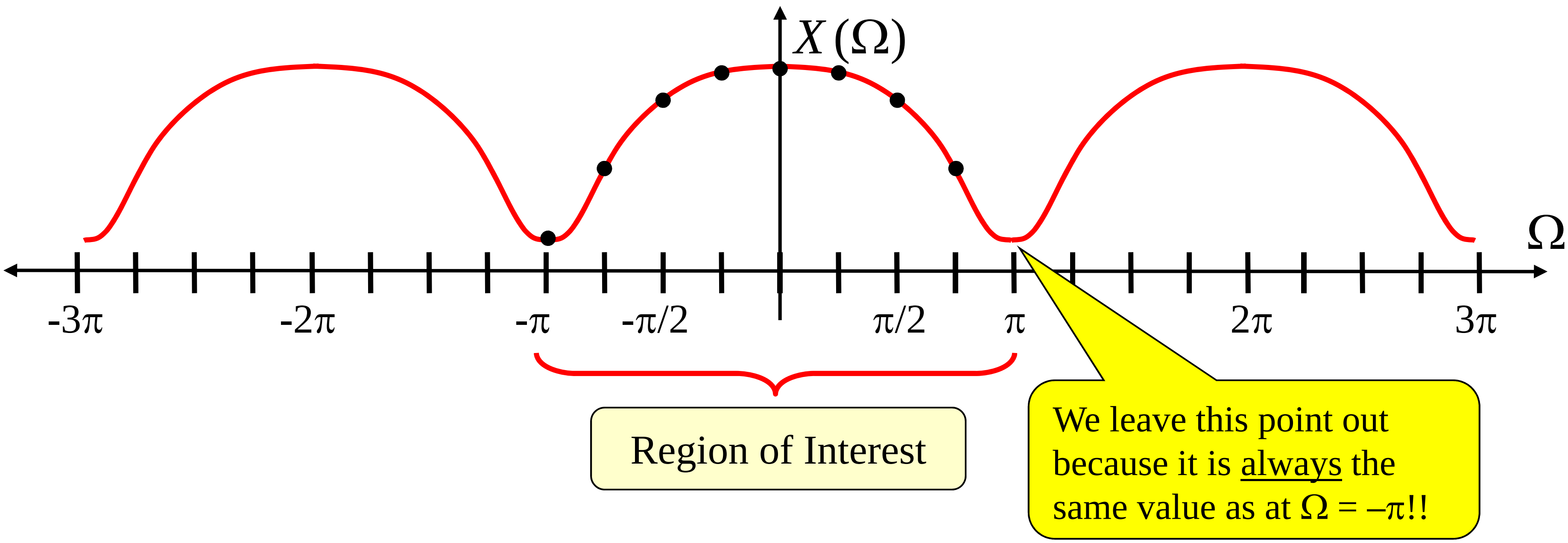
If we could compute this at every Ω value… it might look like this:



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

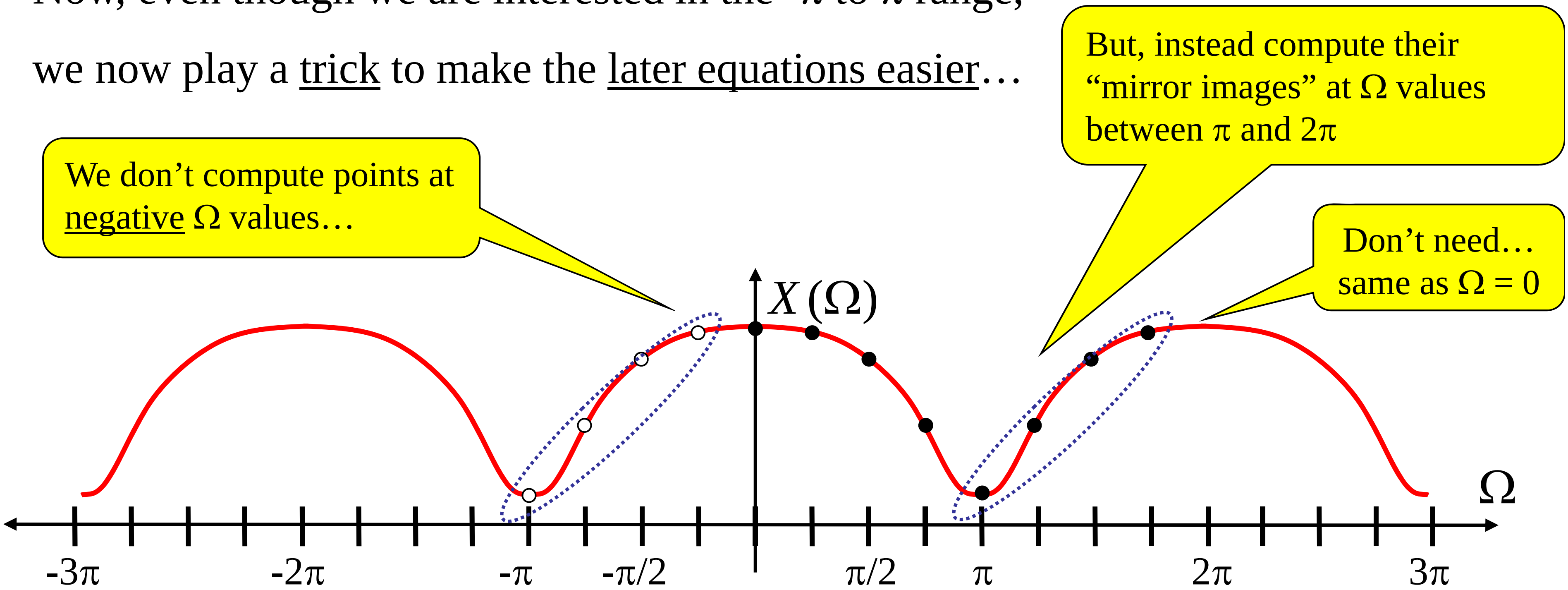
Now suppose we take the numerical data $x[n]$ for $n = 0, \ldots, N\text{-}1$

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:

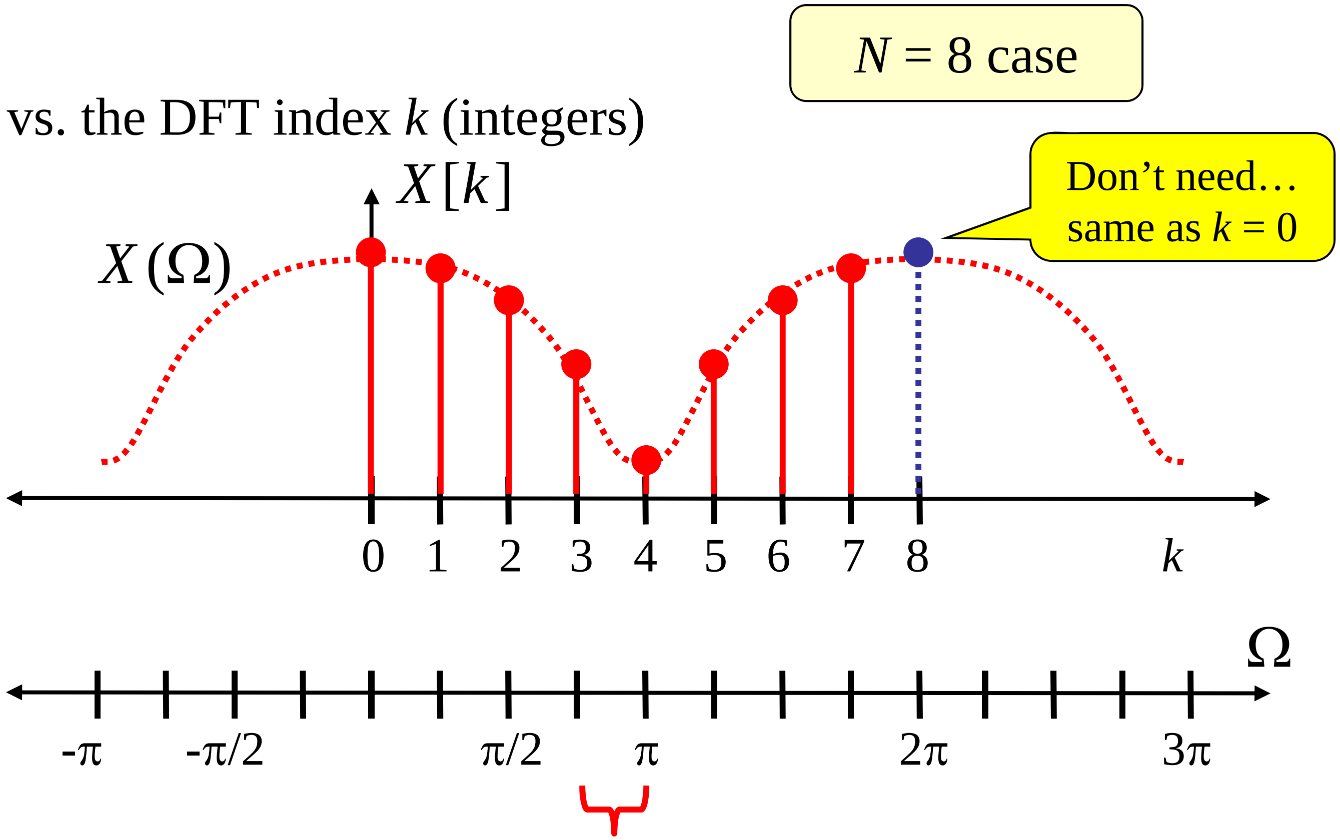
$$\boxed{X[k] = \sum_{n=0}^{N-1} x[n] e^{-j2\pi kn/N} \qquad k = 0, 1, 2, \ldots, N-1}$$

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

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

Inverse DFT

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:

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

Inverse DFT (IDFT)

Compare to the DFT… a remarkably similar structure:

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

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 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!!

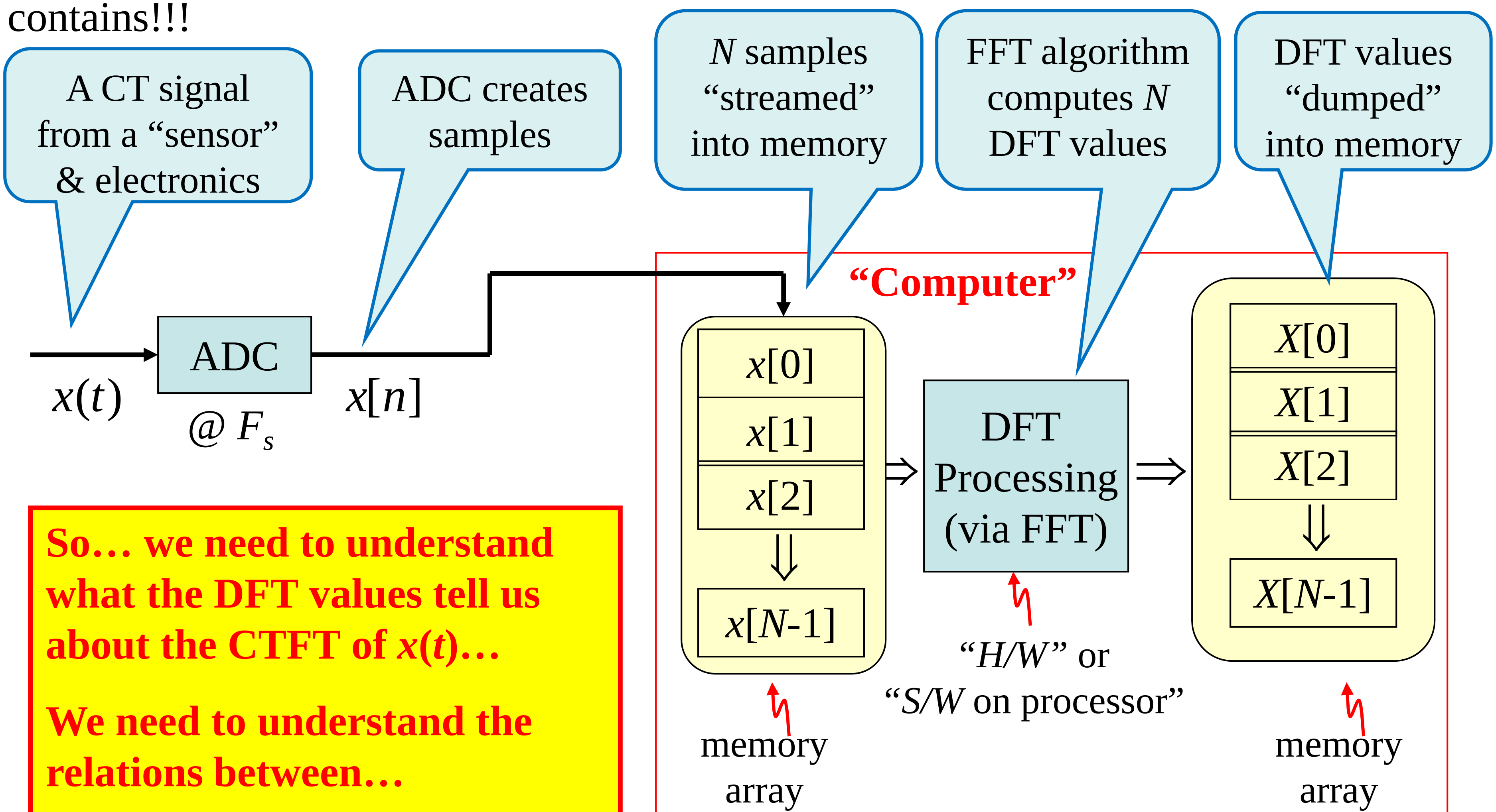
Now…

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

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

We can use the DFT to implement numerical FT processing

This enables us to numerically analyze a signal to find out what frequencies it contains!!!



So… we need to understand what the DFT values tell us about the CTFT of $x(t)$…

We need to understand the relations between…

CTFT, DTFT, and DFT

We'll mathematically explore the link between DTFT & DFT in two cases:

1. For $x[n]$ of **finite duration**:

$$\ldots 0 \;\; 0 \;\; \underbrace{x[0] \;\; x[1] \;\; x[2] \ldots \;\; X[N-1]}_{N \text{ "non-zero" terms}} \;\; 0 \;\; 0 \ldots$$

This case hardly ever happens… but it's easy to analyze and provides a perspective for the 2nd case

(of course, we could have some of the interior values = 0)

For this case… we'll assume that the signal ***is*** zero outside the range that we have captured.

So… we have all of the meaningful signal data.

This is the practical case.

2. For $x[n]$ of **infinite duration** …or at least of duration longer than what we can get into our "DFT Processor" inside our "computer".

So… we don't have all the meaningful signal data.

What effect does that have? How much data do we need for a given goal?

DFT & DTFT: Finite Duration Case

If $x[n] = 0$ for $n < 0$ and $n \geq N$ then the DTFT is:

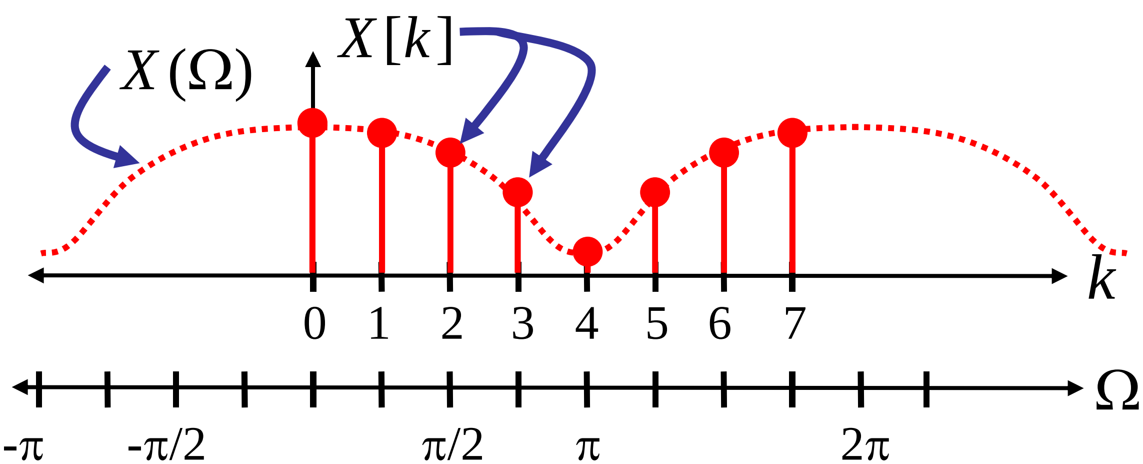
$$\boxed{X(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n} = \sum_{n=0}^{N-1} x[n]e^{-j\Omega n}}$$

we can leave out terms that are zero

Now… if we take these N samples and compute the DFT (using the FFT, perhaps) we get:

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

Comparing these we see that for the finite-duration signal case: $\boxed{X[k] = X\left(k\,\frac{2\pi}{N}\right)}$



DTFT & DFT :

DFT points lie exactly on the finite-duration signal's DTFT!!!

Summary of DFT & DTFT for a ***finite*** duration $x[n]$

$$x[n] \xrightarrow{\text{DTFT}} X(\Omega)$$

$$x[n] \xrightarrow{\text{DFT}} X[k] = X\left(k\frac{2\pi}{N}\right)$$

Points of DFT are "samples" of DTFT of $x[n]$

The number of samples N sets how closely spaced these "samples" are on the DTFT… seems to be a limitation.

"Zero-Padding Trick"

After we collect our N samples, we tack on some additional zeros at the end to trick the "DFT Processing" into thinking there are really more samples.

(Since these are zeros tacked on they don't change the values in the DFT sums)

If we now have a total of N_Z "samples" (including the tacked on zeros), then the spacing between DFT points is $2\pi/N_Z$ which is smaller than $2\pi/N$

Ex. DTFT & DFT of pulse

$$x[n] = \begin{cases} 1, & n = 0,1,2,\ldots 2q \\ 0, & otherwise \end{cases} \qquad \text{Recall}: \; p_q[n] = \begin{cases} 1, & n = -q, \ldots, -1, 0, 1, \ldots, q \\ 0, & otherwise \end{cases}$$

Then… $x[n] = p_q[n-q]$

Note: we'll need the delay property for DTFT

From DTFT Table: $\boxed{p_q[n] \leftrightarrow P_q(\Omega) = \dfrac{\sin[(q+0.5)\Omega]}{\sin[\Omega/2]}}$

From DTFT Property Table

(Delay Property): $\boxed{X(\Omega) = \dfrac{\sin[(q+0.5)\Omega]}{\sin[\Omega/2]} e^{-jq\Omega}}$

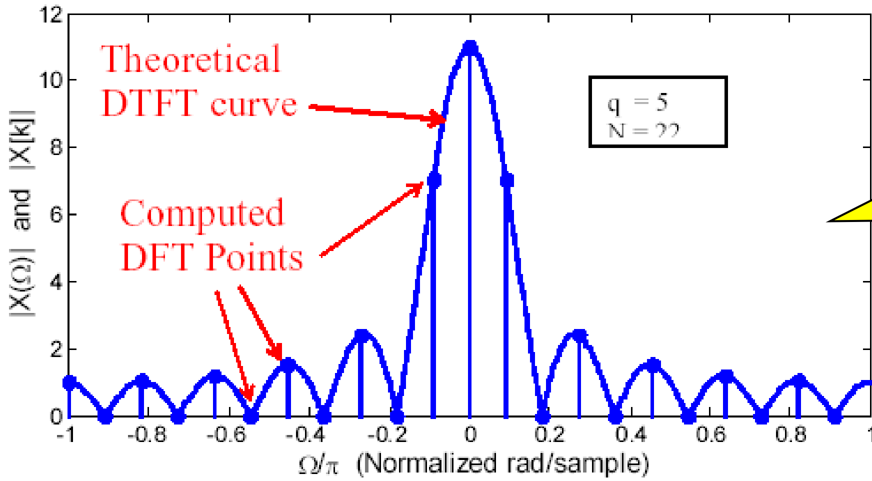
Since $x[n]$ is a finite-duration signal then the DFT of the $N = 2q+1$ non-zero samples is just samples of the DTFT: $\boxed{X[k] = X\left(k\dfrac{2\pi}{N}\right)}$

$$\boxed{X[k] = \frac{\sin[(q+.5)2\pi k/N]}{\sin[\pi k/N]} e^{-jq2\pi k/N}}$$

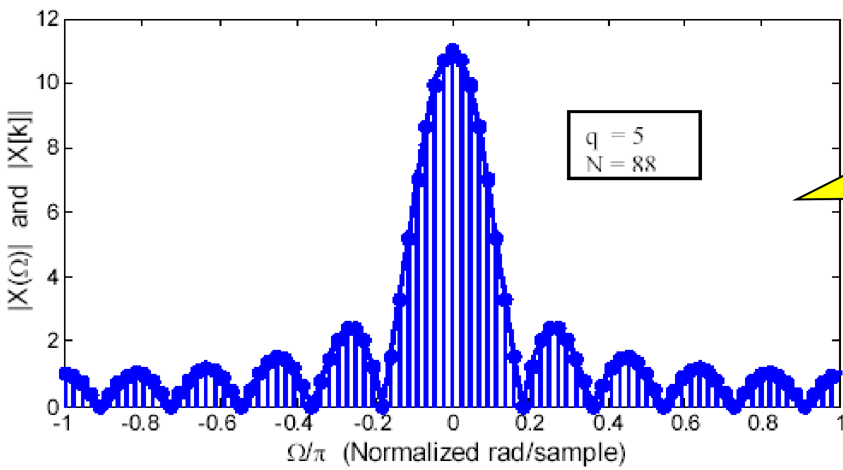
Note that if we don't zero pad, then all but the $k = 0$ DFT values are zero!!!

That doesn't show what the DTFT looks like! So we need to use zero-padding.

Here are two numerically computed examples, both for the case of $q = 5$:



For the case of zero-padding 11 zeros onto the end of the signal… the DFT points still don't really show what the DTFT looks like!



For the case of zero-padding 77 zeros onto the end of the signal… **NOW** the DFT points really show what the DTFT looks like!

DFTs were computed using matlab's fft command… see code on next slide

Compute the DTFT Equation derived for the pulse. Using *eps* adds a very small number to avoid getting $\Omega = 0$ and then dividing by 0

```
omega=eps+(-1:0.0001:1)*pi;
q=5;    % used to set pulse length to 11 points
X=sin((q+0.5)*omega)./sin(omega/2);
subplot(2,1,1)
plot(omega/pi,abs(X));   % plot magn of DTFT
xlabel('\Omega/\pi  (Normalized rad/sample)')
ylabel('|X(\Omega)|  and  |X[k]|')
hold on
x=zeros(1,22);  % Initially fill x with 22 zeros
x(1:(2*q+1))=1;   % Then fill first 11 pts with ones
Xk=fftshift(fft(x));    % fft computes the DFT and fftshift re-orders points
                        % to between  -pi and pi
omega_k=(-11:10)*2*pi/24;  % compute DFT frequencies, except make them
                        % between  -pi and pi
stem(omega_k/pi,abs(Xk));  % plot DFT vs. normalized frequencies
hold off
subplot(2,1,2)
plot(omega/pi,abs(X));
xlabel('\Omega/\pi  (Normalized rad/sample)')
ylabel('|X(\Omega)|  and  |X[k]|')
hold on
x=zeros(1,88);
x(1:(2*q+1))=1;
Xk=fftshift(fft(x));
omega_k=(-44:43)*2*pi/88;
stem(omega_k/pi,abs(Xk));
hold off
```

Make the zero-padded signal

Compute the DFT

Compute the DFT point's frequency values and plot the DFT

Important Points for *Finite-Duration* Signal Case

- DFT points lie on the DTFT curve… perfect view of the DTFT
  - But… only if the DFT points are spaced closely enough
- Zero-Padding doesn’t change the shape of the DFT…
- It just gives a denser set of DFT points… all of which lie on the true DTFT
  - Zero-padding provides a better view of this “perfect” view of the DTFT

DFT & DTFT: Infinite Duration Case

As we said… in a computer we cannot deal with an infinite number of signal samples.

So say there is some signal that "goes on forever" (or at least continues on for longer than we can or are willing to grab samples)

$x[n] \quad n = \ldots, -3, -2, -1, 0, 1, 2, 3, \ldots$

We **only grab N samples**: $x[n]$, $n = 0, \ldots, N-1$ **We've lost some information!**

We can define an "imagined" finite-duration signal:

$$x_N[n] = \begin{cases} x[n], & n = 0, 1, 2, \ldots, N-1 \\ 0, & elsewhere \end{cases}$$

We can compute the DFT of the N collected samples:

$$X_N[k] = \sum_{n=0}^{N-1} x_N[n] e^{-j2\pi nk/N} \qquad k = 0, 1, \ldots, N-1$$

Q: How does this DFT of the "truncated signal" relate to the "true" DTFT of the full-duration $x[n]$? …which is what we really want to see!!

$$\text{"True" DTFT}: X_\infty(\Omega) = \sum_{n=-\infty}^{\infty} x[n]e^{-j\Omega n}$$

What we *want* to see

$$\text{DTFT of truncated signal}: X_N(\Omega) = \sum_{n=-\infty}^{\infty} x_N[n]e^{-j\Omega n} = \sum_{n=0}^{N-1} x[n]e^{-j\Omega n}$$

A *distorted* version of what we want to see

$$\text{DFT of collected signal data}: X_N[k] = \sum_{n=0}^{N-1} x[n]e^{-j2\pi kn/N}$$

What we *can* see

$$\text{DFT gives samples of } X_N(\Omega)$$

So… DFT of collected data gives "samples" of $\underbrace{\text{DTFT of \textit{truncated} signal}}_{\neq \text{"True" DTFT}}$

⇒DFT of collected data does not perfectly show DTFT of complete signal.

Instead, the DFT of the data shows the DTFT of the *truncated* signal…

So **our goal** is to understand what kinds of "errors" are in the "truncated" DTFT
…then we'll know what "errors" are in the computed DFT of the data

To see what the DFT <u>does</u> show we need to understand how

$X_N(\Omega)$ relates to $X_\infty(\Omega)$

First, we note that:

$$\underbrace{x_N[n] = x[n]p_q[n-q]}_{} \xrightarrow{\text{DTFT}} P_q(\Omega) = \frac{\sin[N\Omega/2]}{\sin[\Omega/2]} e^{-j(N-1)\Omega/2}$$

with $N=2q+1$

From "mult. in time domain" property in DTFT Property Table:

$$\boxed{X_N(\Omega) = \frac{1}{2\pi}\int_{-\pi}^{\pi} X_\infty(\lambda)P_q(\Omega-\lambda)d\lambda}$$

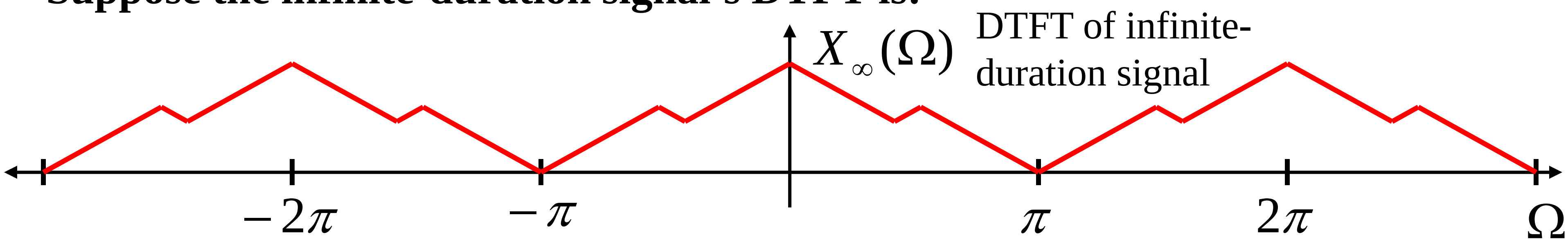
causes "smearing" of $X_\infty(\Omega)$

$\Rightarrow$ So… $X_N(\Omega)$ …which we can see via the DFT $X_N[k]$ …

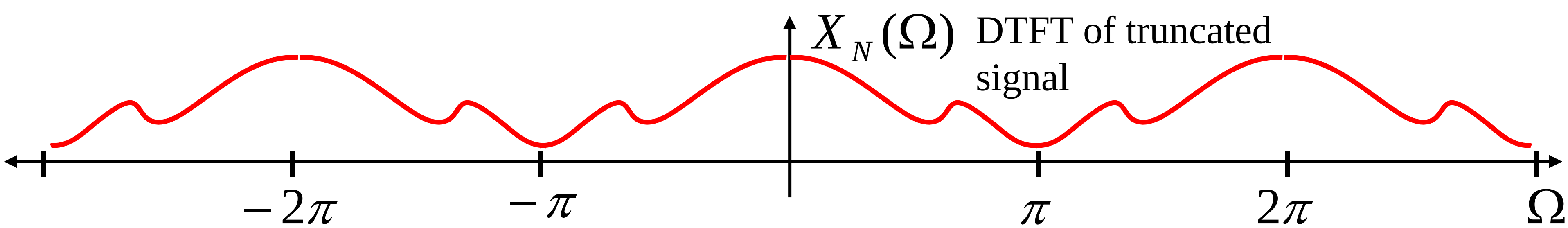
is a "smeared" version of $X_\infty(\Omega)$

"Fact": The more data you collect, the less smearing
… because $P_q(\Omega)$ becomes more like $\delta(\Omega)$

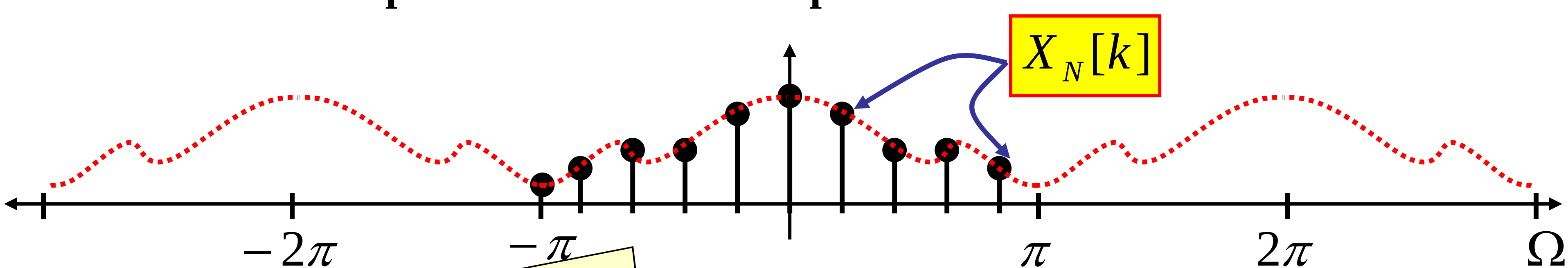
Suppose the infinite-duration signal's DTFT is:



Then it gets smeared into something that might look like this:



Then the DFT computed from the *N* data points is:



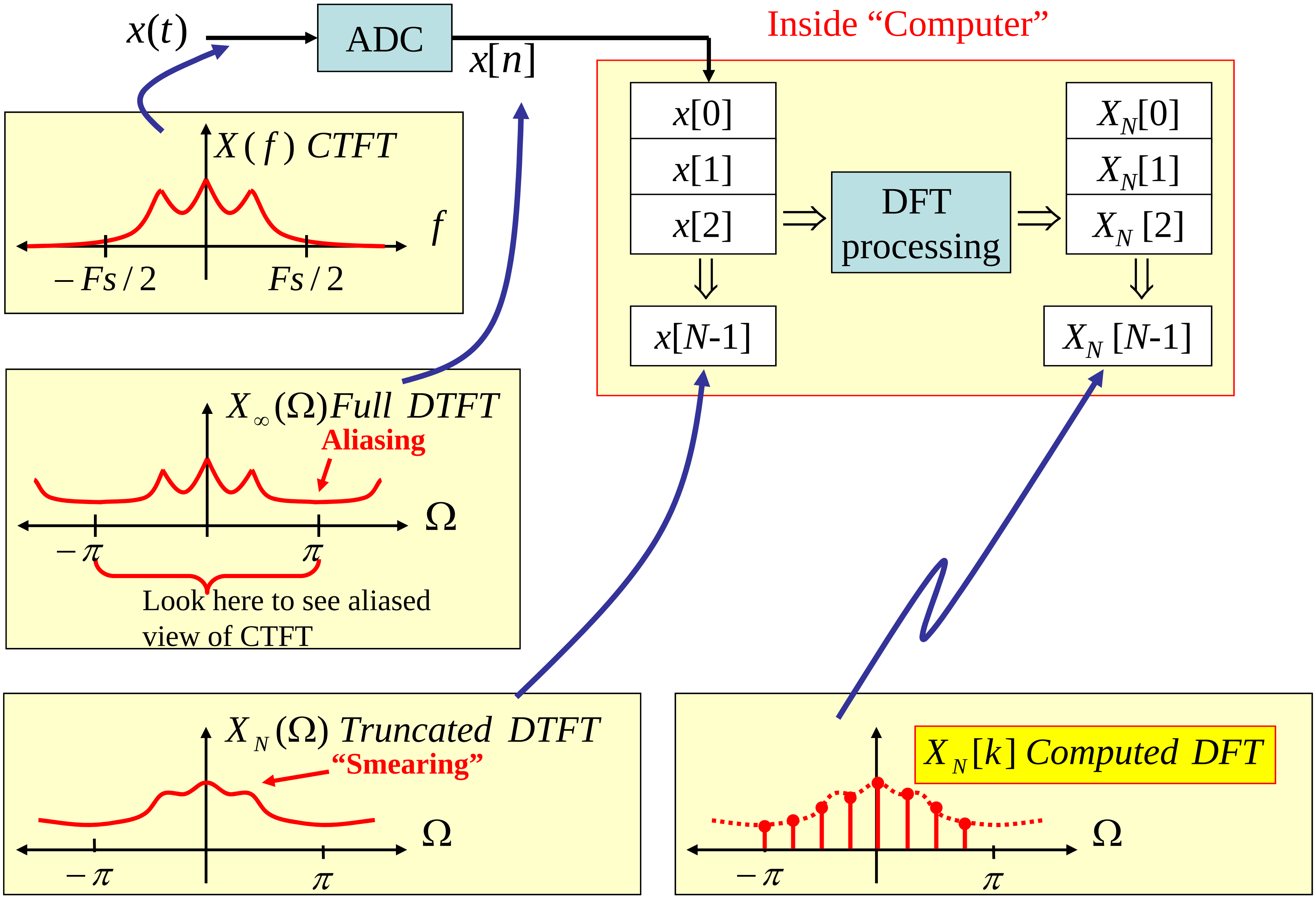
The DFT points are shown *after* "upper" points are moved (e.g., by MATLAB "fftshift")

The only case that ***really*** happens in practice!

Important points for Infinite-Duration Signal Case

1. DTFT of finite collected data is a "smeared" version of the DTFT of the infinite-duration data
2. The computed DFT points lie on the "smeared" DTFT curve… not the "true" DTFT
   a. This gives an imperfect view of the true DTFT!
3. "Zero-padding" gives denser set of DFT points… a better view of this imperfect view of the desired DTFT!!!

Connections between the CTFT, DTFT, & DFT



Errors in a Computed DFT

CTFT

↓

Aliasing Error – control through F_s choice (i.e. through proper sampling)

↓

DTFT_{∞}

↓

"Smearing" Error – control through N choice / "window" choice

See DSP course

↓

DTFT_N

↓

"Grid" Error – control through N choice / "zero padding"

↓

DFT

This is the only thing we can compute from data… and it has all these "errors" in it!! The theory covered here allows an engineer to understand how to control the amount of those errors!!!

Zero padding trick

Collect N samples → defines $X_N(\Omega)$

Tack M zeros on at the end of the samples

Take $(N + M)$pt. DFT → gives points on $X_N(\Omega)$ spaced by $2\pi/(N+M)$
(rather than $2\pi/N$)