Image Sampling

Michael Kazhdan

(601.457/657)

Sampling Questions

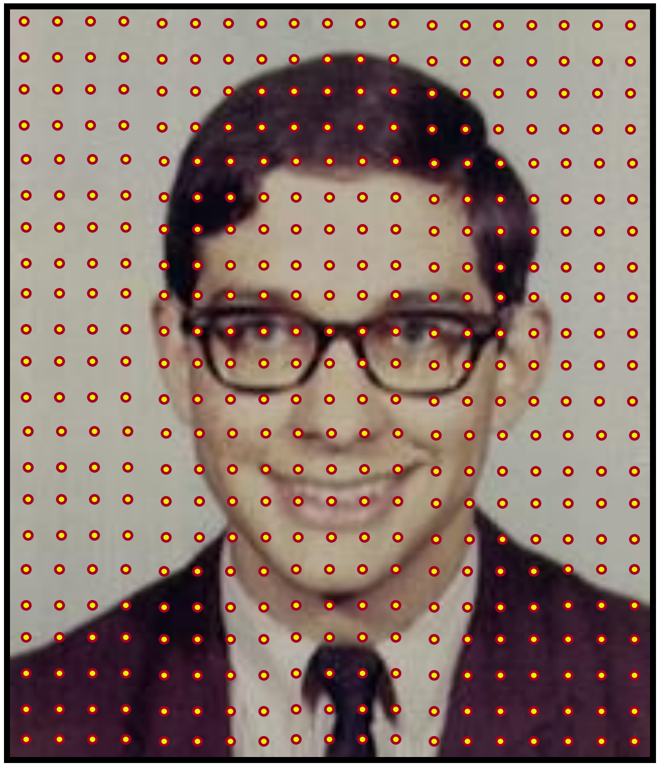
- How should we sample an image:
  - Nearest Point Sampling?
  - Bilinear Sampling?
  - Gaussian Sampling?
  - Something Else?

Image Representation

What is an image?

An image is a discrete collection of pixels, each representing the value(s) of a continuous function.

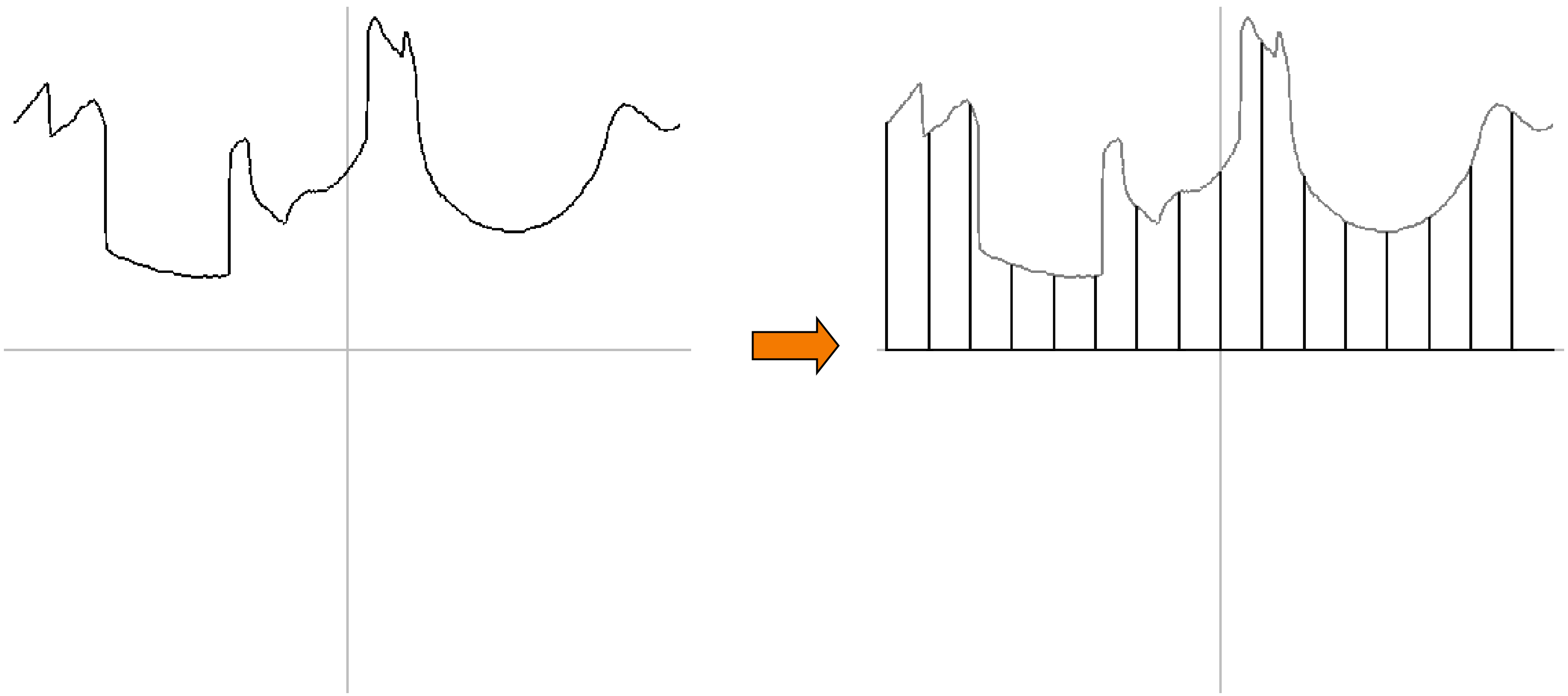
Continuous image



Digital image

Sampling

Consider a 1D example:

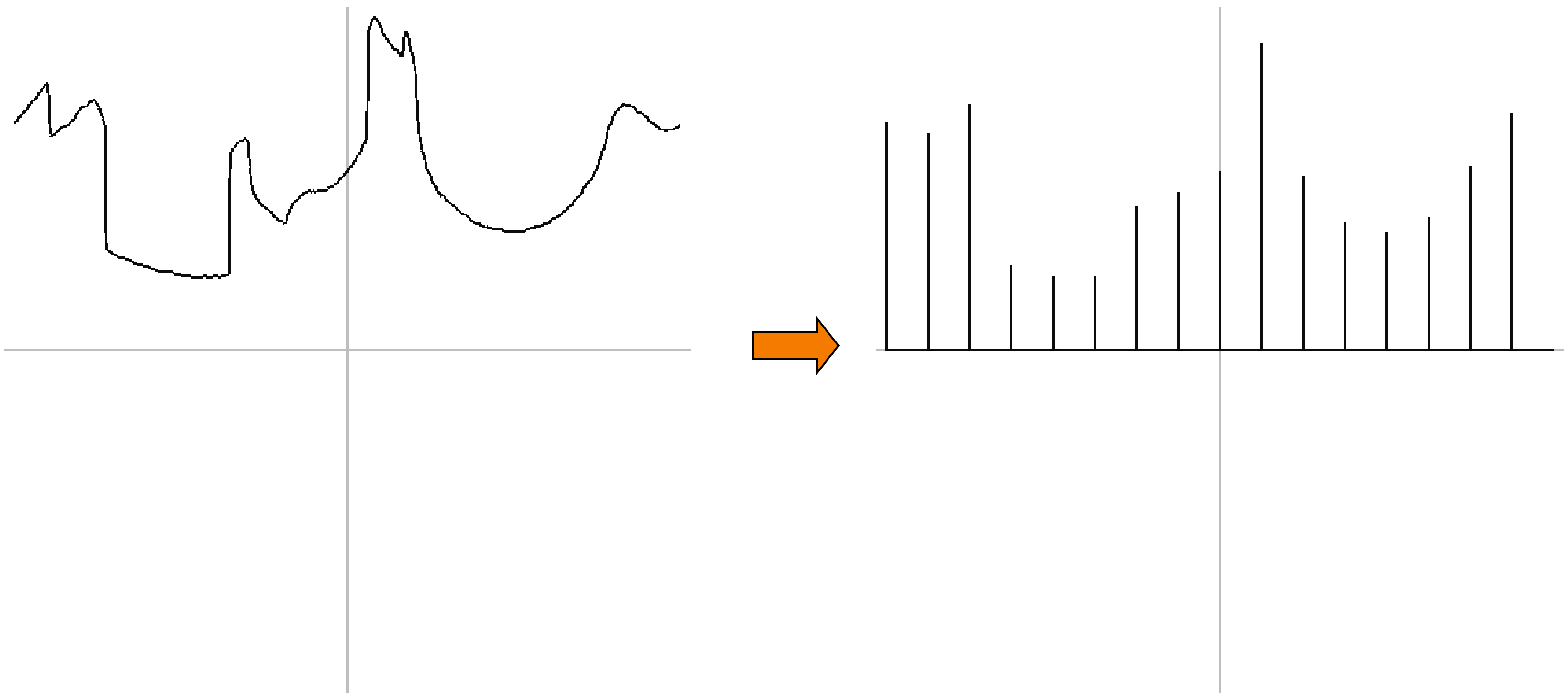


Continuous Function

Discrete Samples

Sampling

Consider a 1D example:



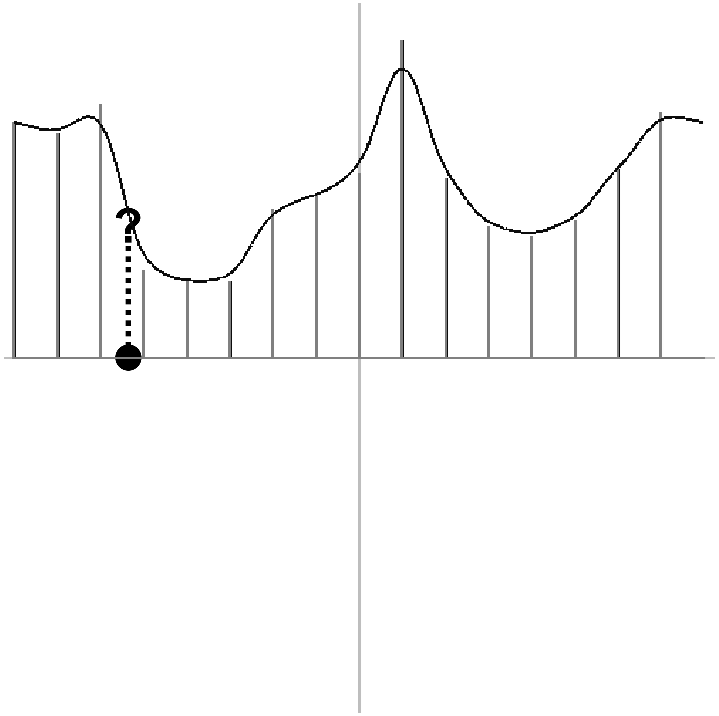
Continuous Function   Discrete Samples

Sampling

At in-between positions, values are undefined.

How do we determine the value of a sample?

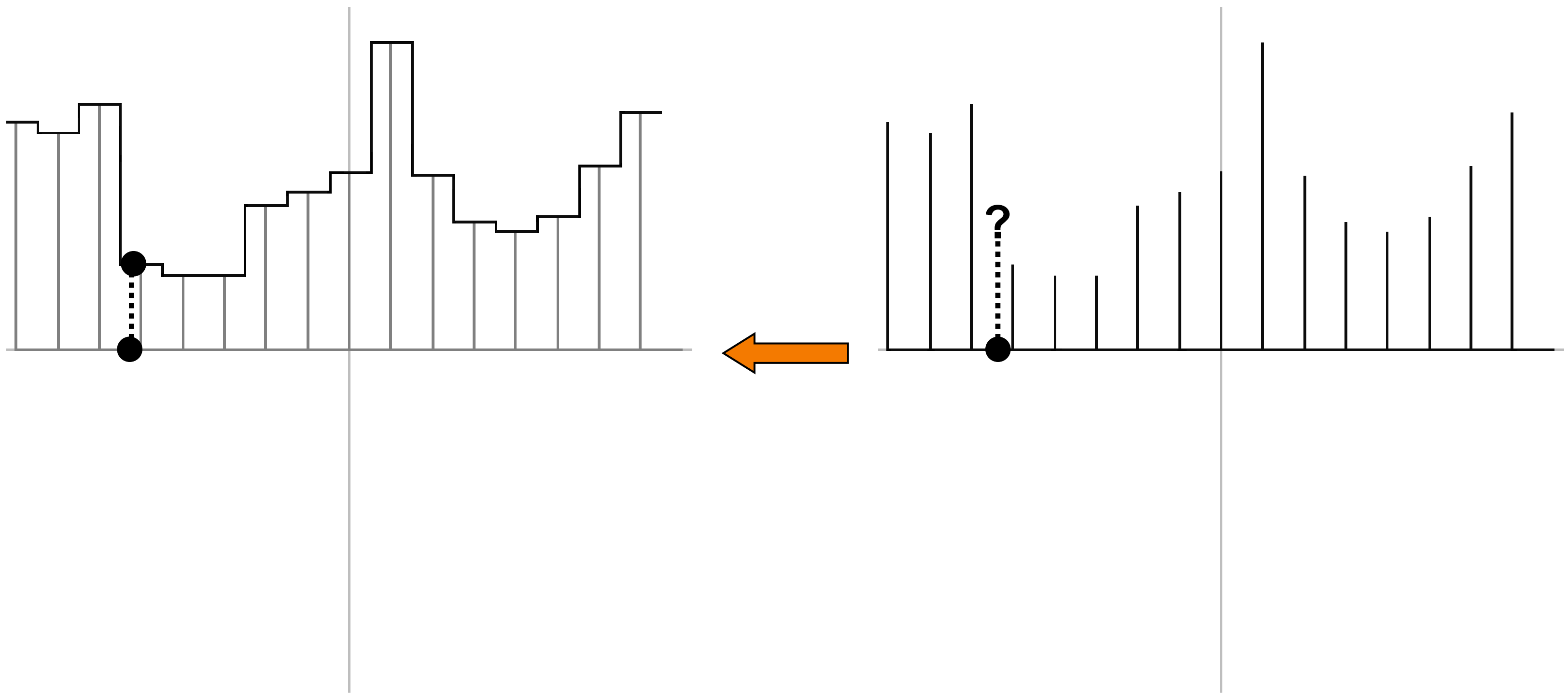
Turn the discrete collection of samples into a continuous function that can be sampled at arbitrary locations.



Discrete Samples

Nearest Point Sampling

The value at a point is the value of the closest discrete sample.

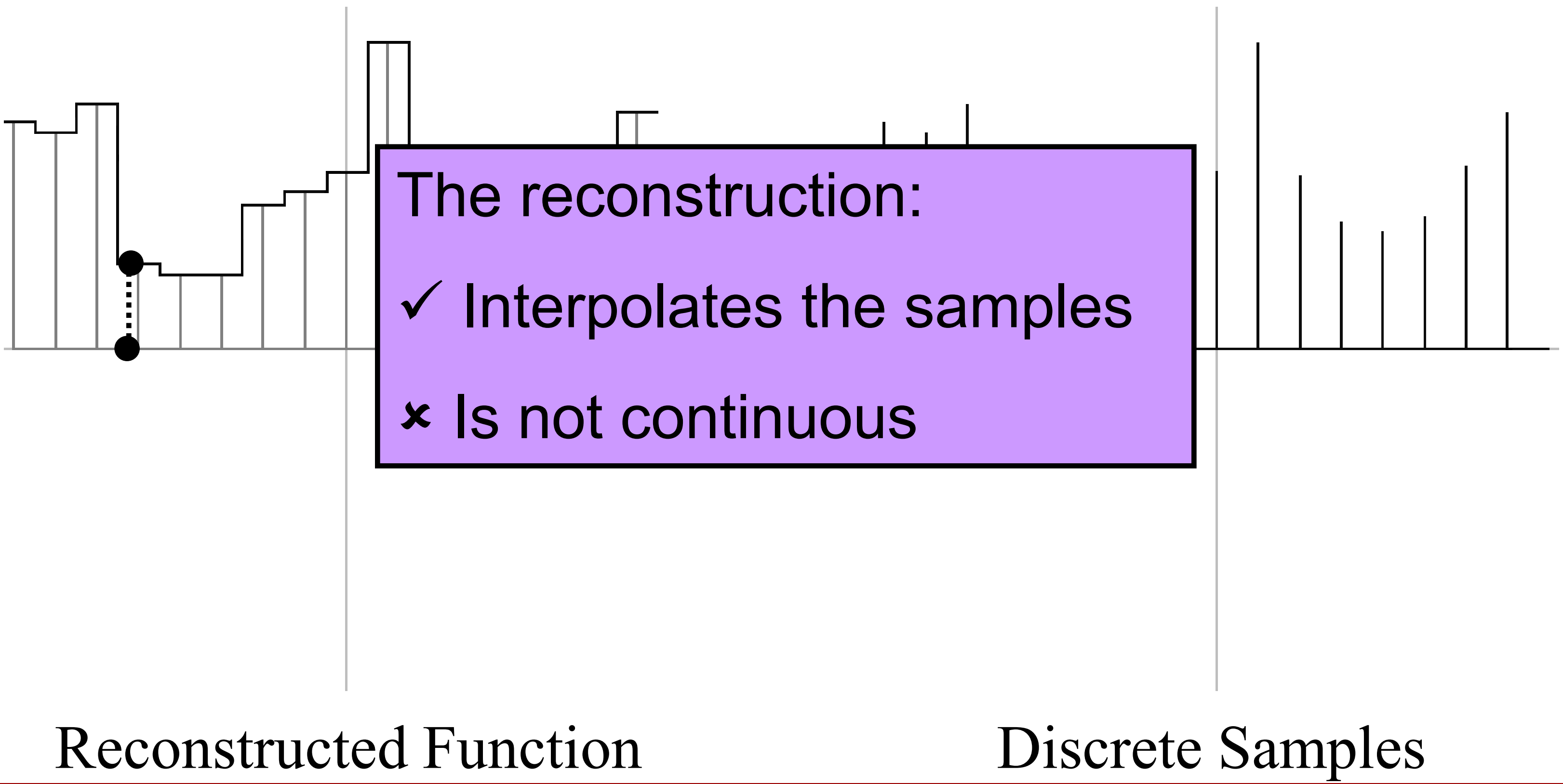


Reconstructed Function

Discrete Samples

Nearest Point Sampling

The value at a point is the value of the closest discrete sample.



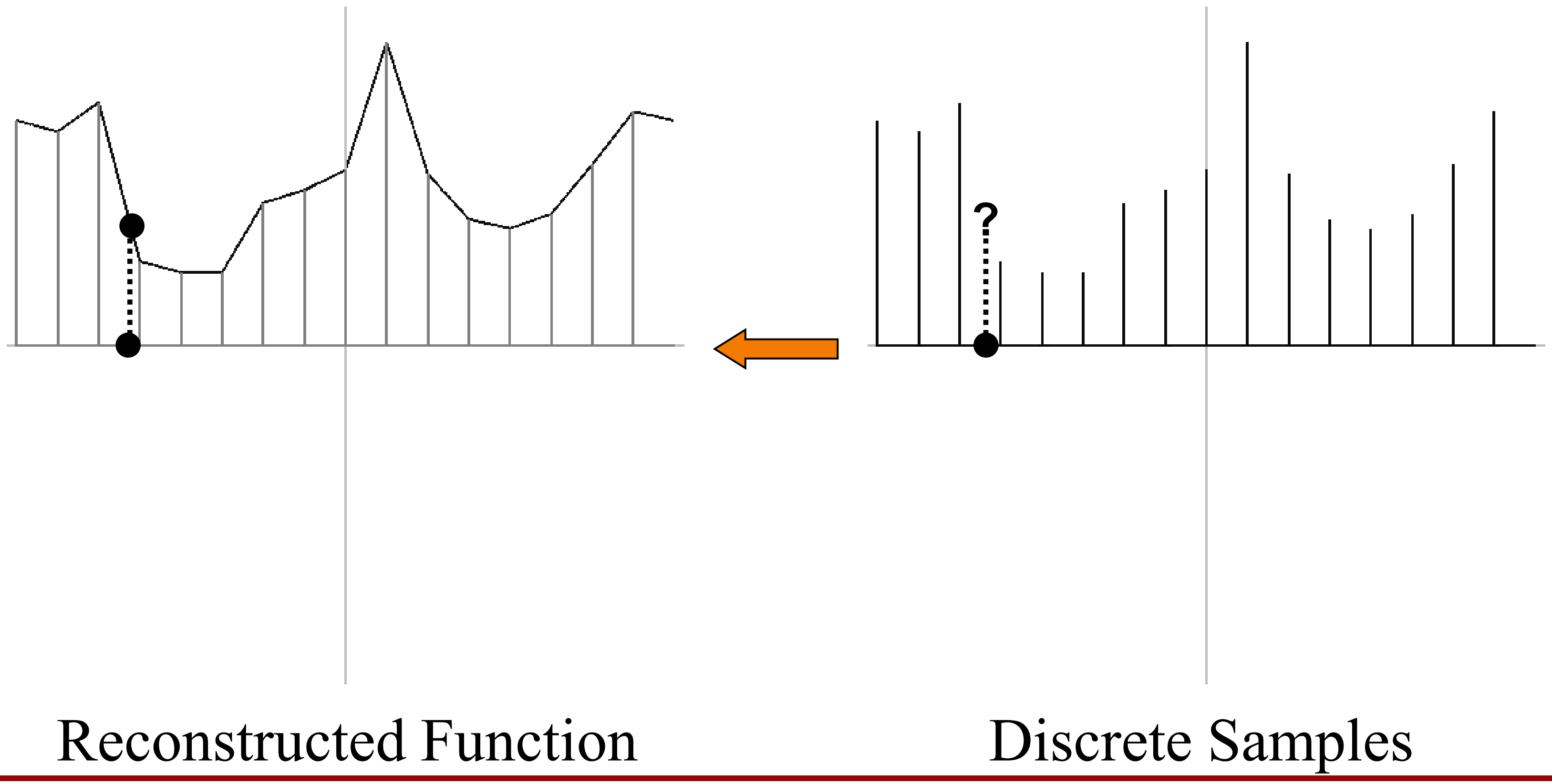
The reconstruction:

✓ Interpolates the samples

✗ Is not continuous

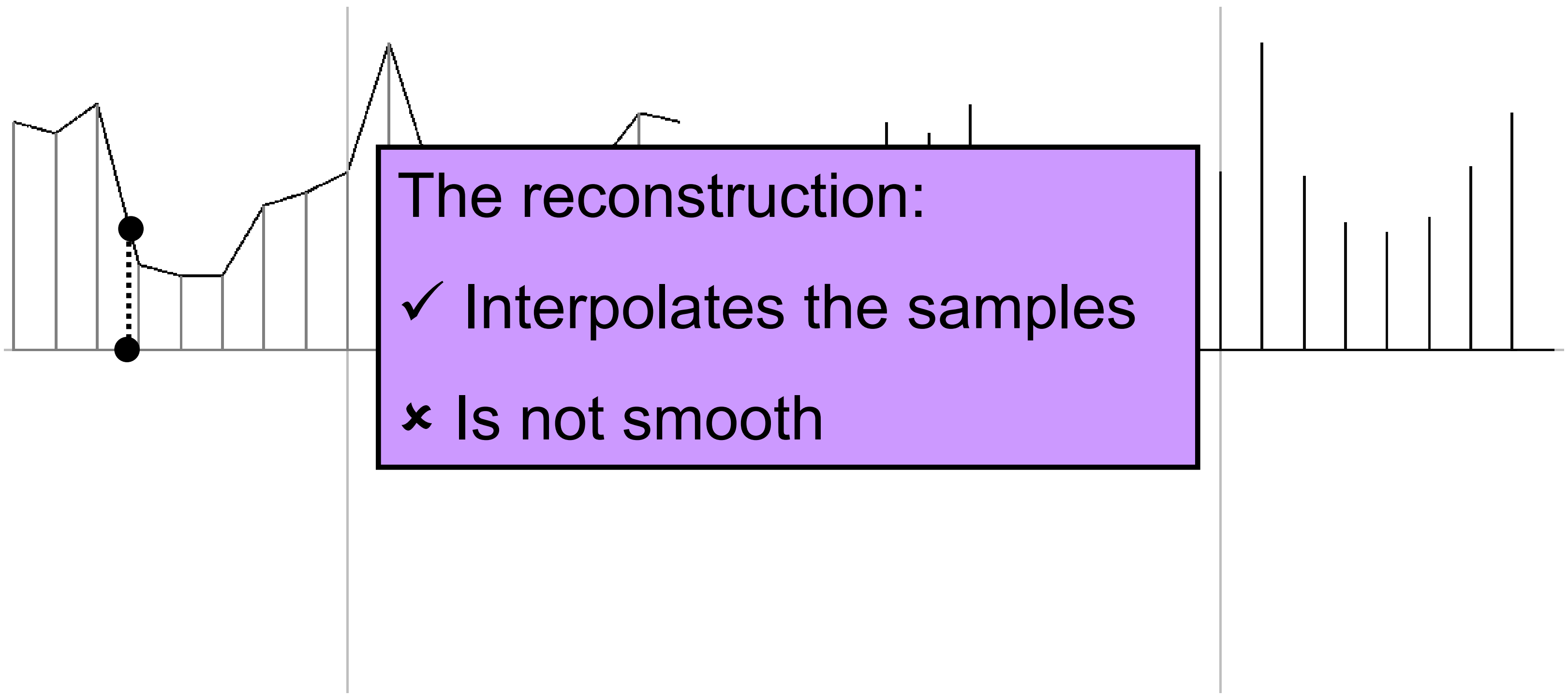
(Bi)linear Sampling

The value at a point is the (bi)linear interpolation of the two surrounding samples.



(Bi)linear Sampling

The value at a point is the (bi)linear interpolation of the two surrounding samples.



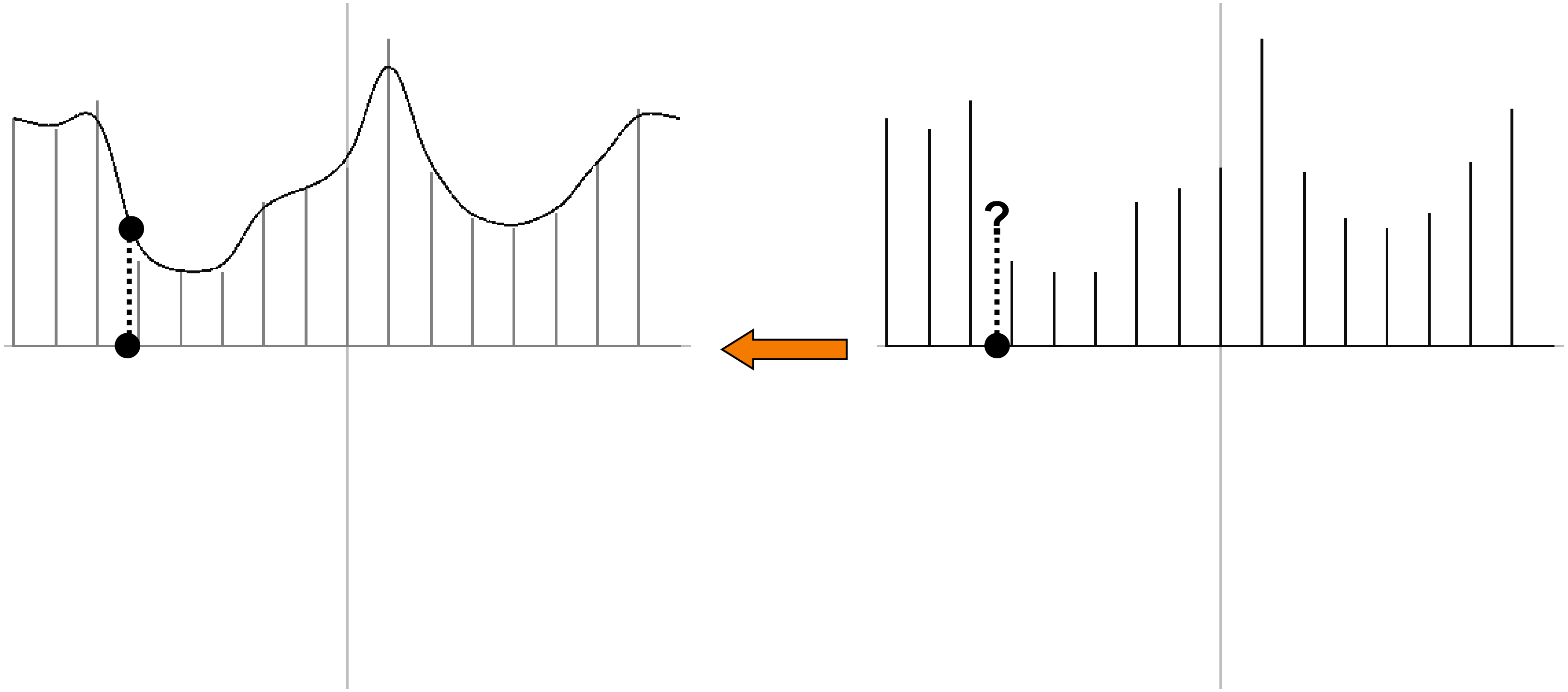
The reconstruction:

✓ Interpolates the samples

✗ Is not smooth

Gaussian Sampling

The value at a point is the Gaussian average of the surrounding samples.



Reconstructed Function

Discrete Samples

Gaussian Sampling

The value at a point is the Gaussian average of the surrounding samples.

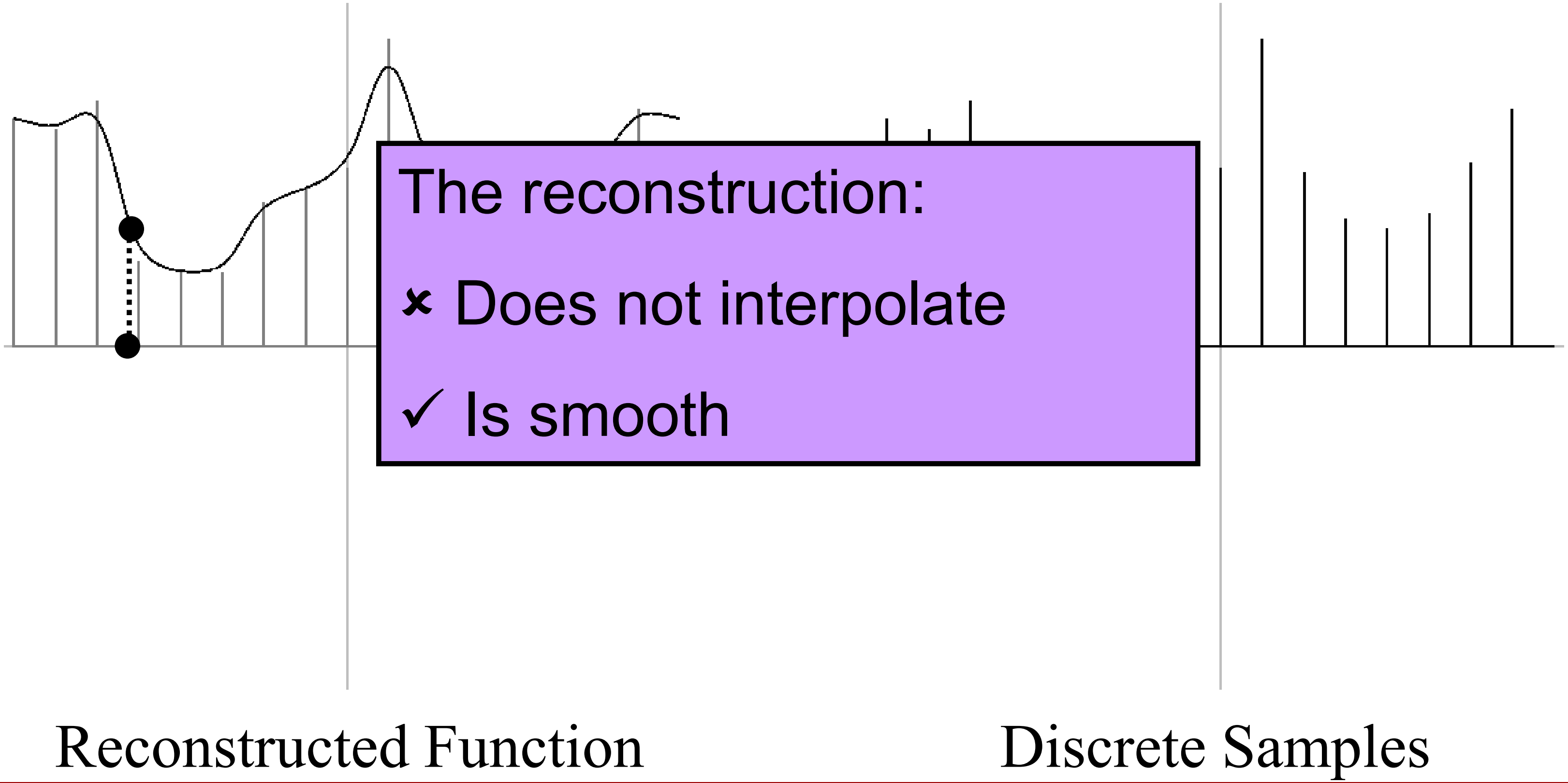
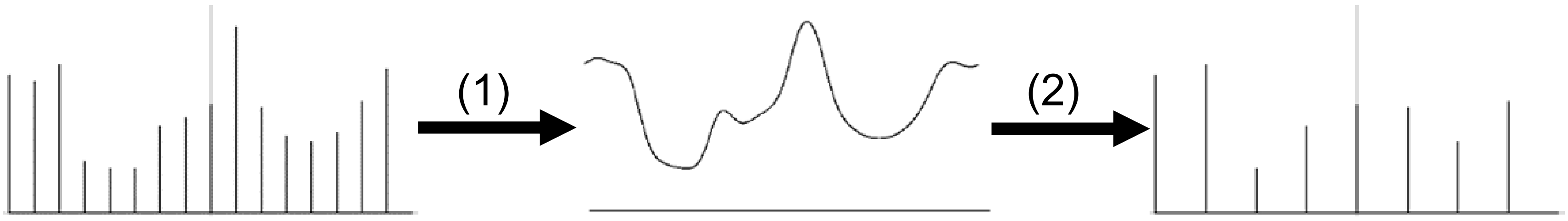


Image Sampling

Conceptually, this is done in two steps:

1. Reconstruct a continuous function from input samples.
2. Sample the continuous function at the new sample positions.



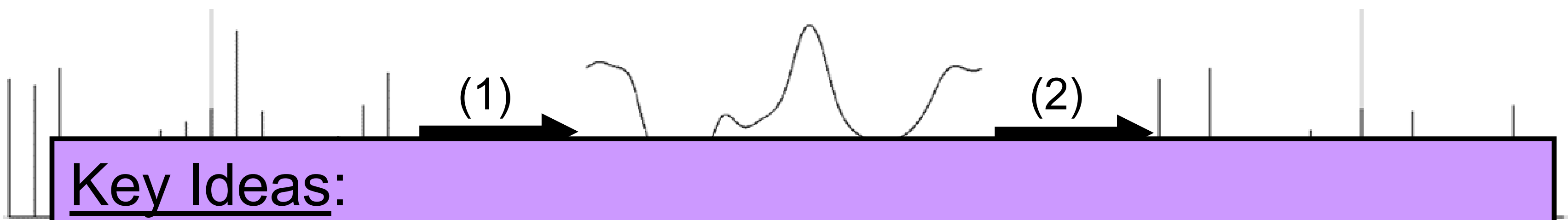
Challenge:

Reconstruction is an under-constrained problem
(i.e. there are many functions fitting the samples.)
⇒ Need to define what makes a good reconstruction.

Image Sampling

Conceptually, this is done in two steps:

1. Reconstruct a continuous function from input samples.
2. Sample the continuous function at the new sample positions.



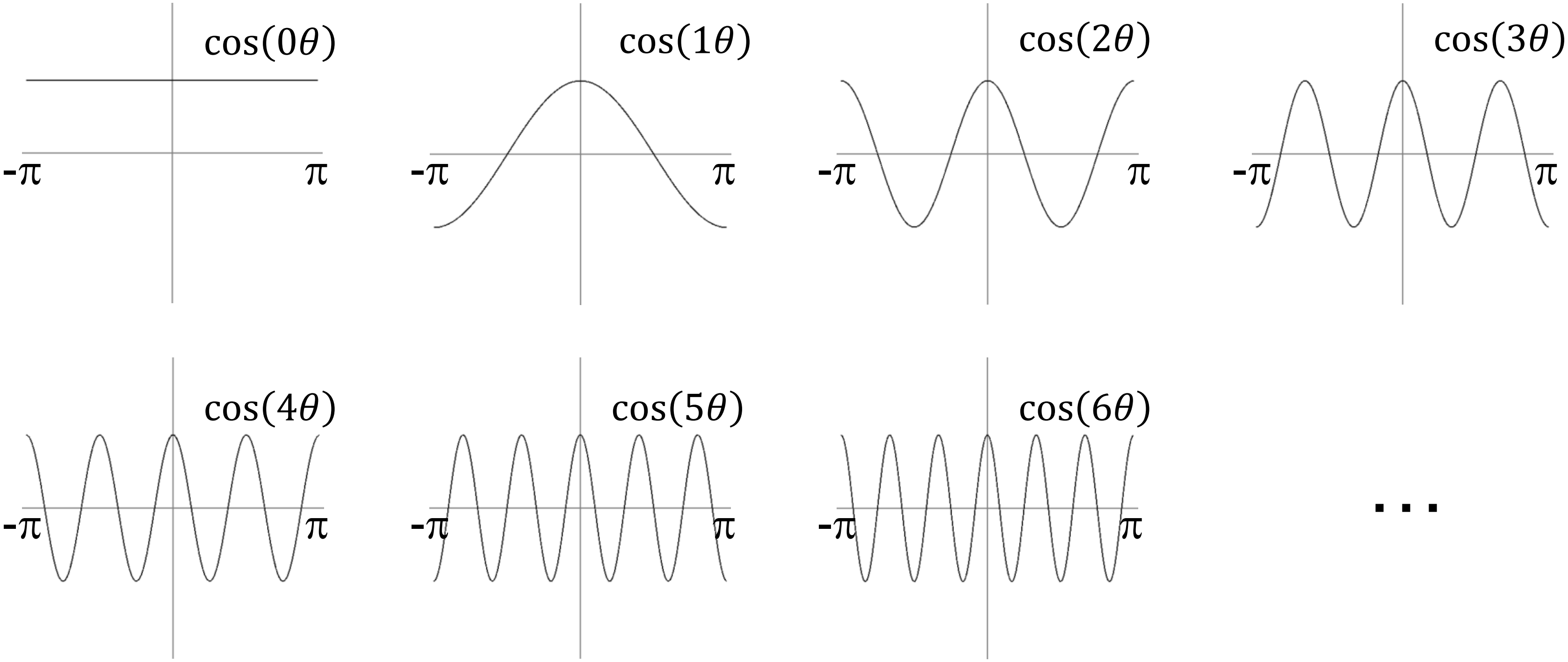
Key Ideas:

1. Of all possible reconstructions, we want the one that is smoothest (has lowest frequencies).
2. It turns out… How we reconstruct should also depend on how we intend to sample.

Signal processing helps us formulate this precisely.

Fourier Analysis

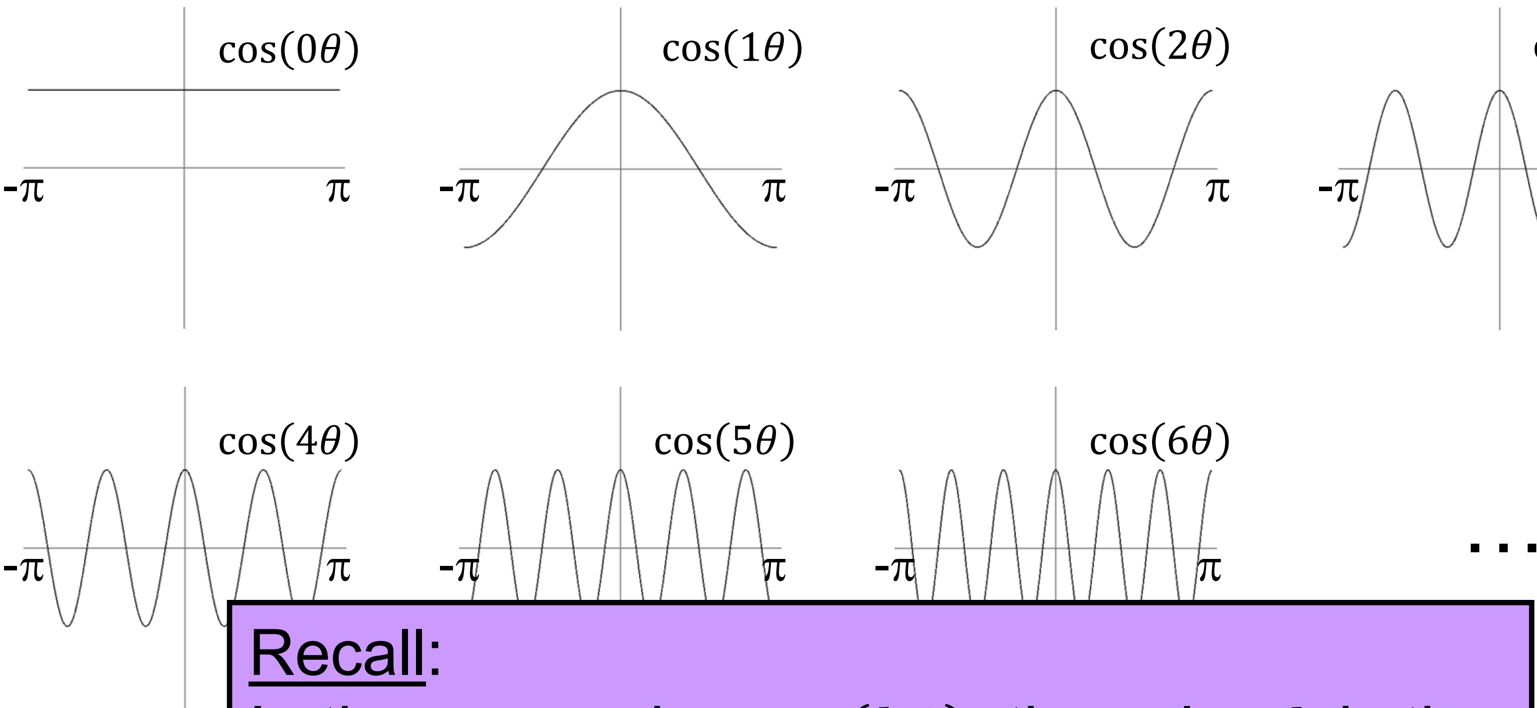
Uniquely describes a signal as a sum of scaled and shifted cosine functions.



The Building Blocks for the Fourier Decomposition

Fourier Analysis

Uniquely describes a signal as a sum of <u>scaled</u> and <u>shifted</u> cosine functions.



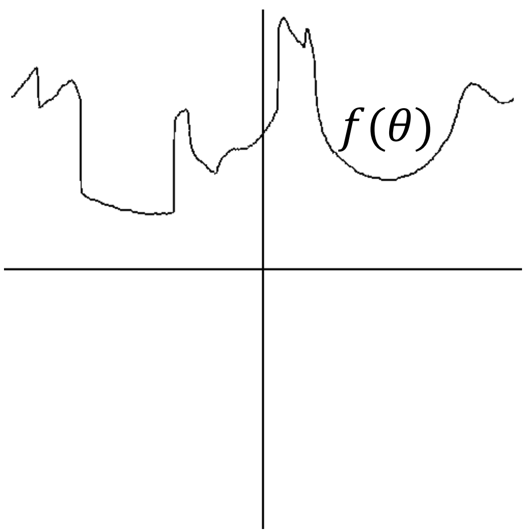
<u>Recall</u>:
In the expression $\cos(k\theta)$, the value k is the *frequency* of the function.

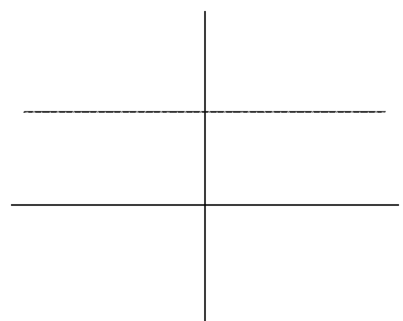
Fourier Analysis

Uniquely describes a signal as a sum of <u>scaled</u> and <u>shifted</u> cosine functions.

- As higher frequency components are added, finer details are captured.

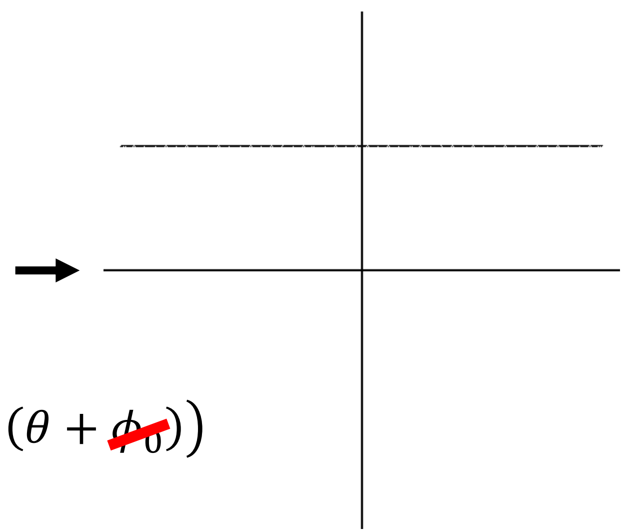
Initial Function	0^{th} Order Approximation





$$f_0(\theta) = a_0 \cdot \cos\big(0 \cdot (\theta + \cancel{\phi_0})\big)$$

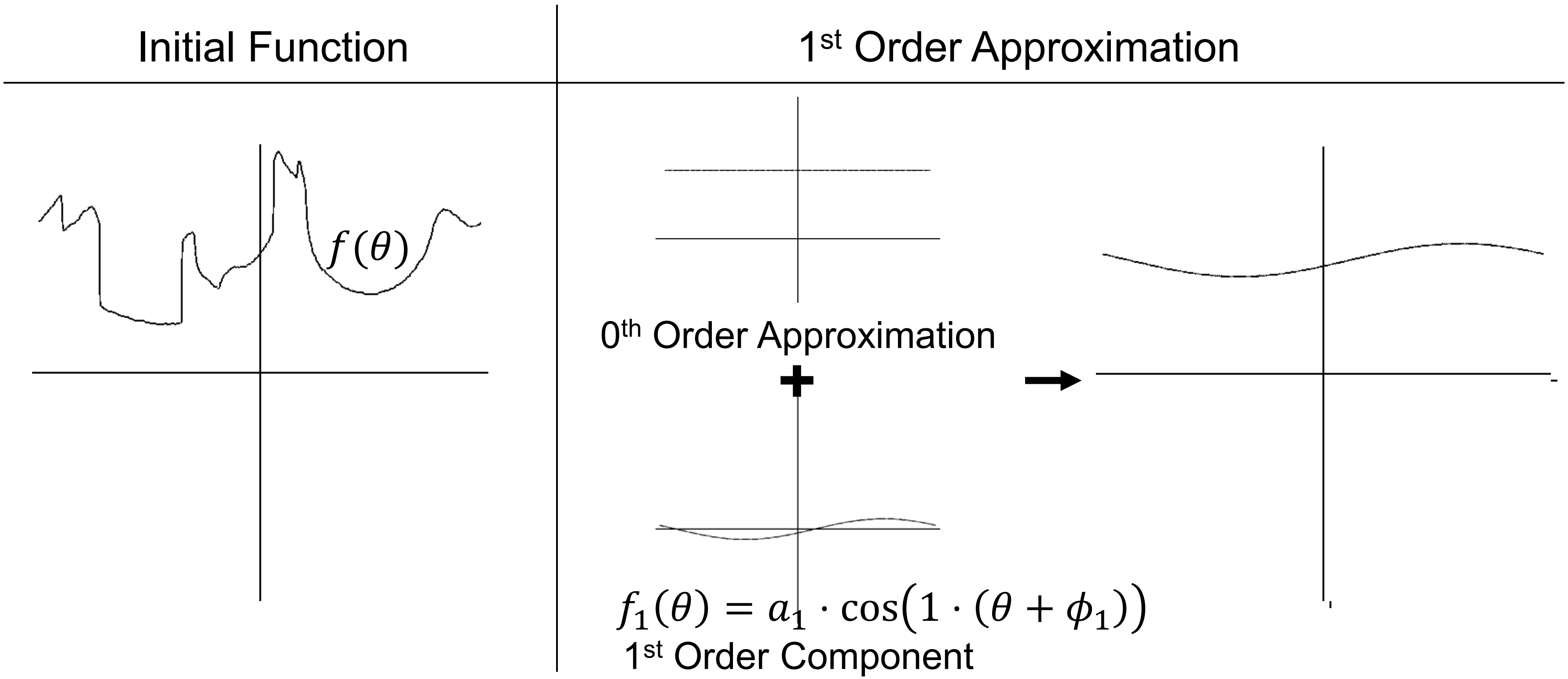
0^{th} Order Component



Fourier Analysis

Uniquely describes a signal as a sum of <u>scaled</u> and <u>shifted</u> cosine functions.

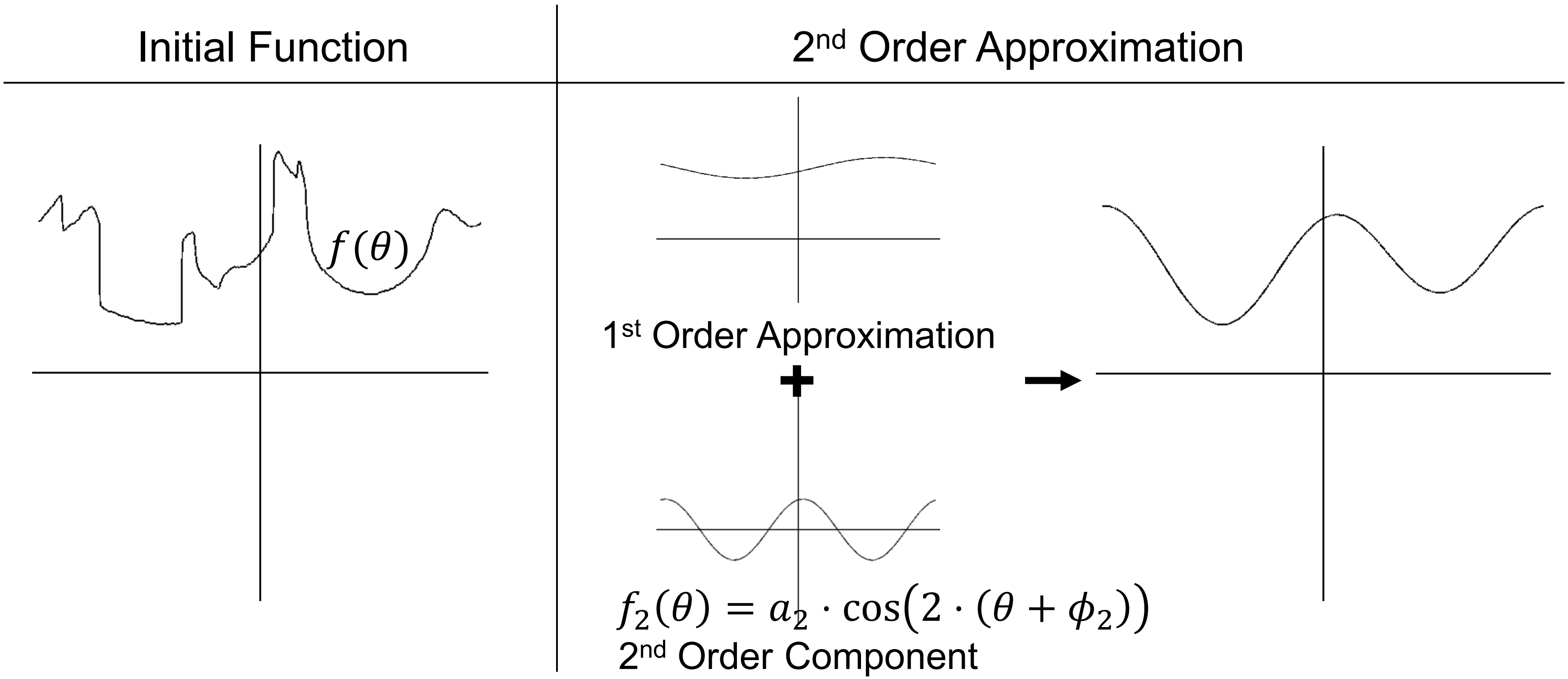
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of <u>scaled</u> and <u>shifted</u> cosine functions.

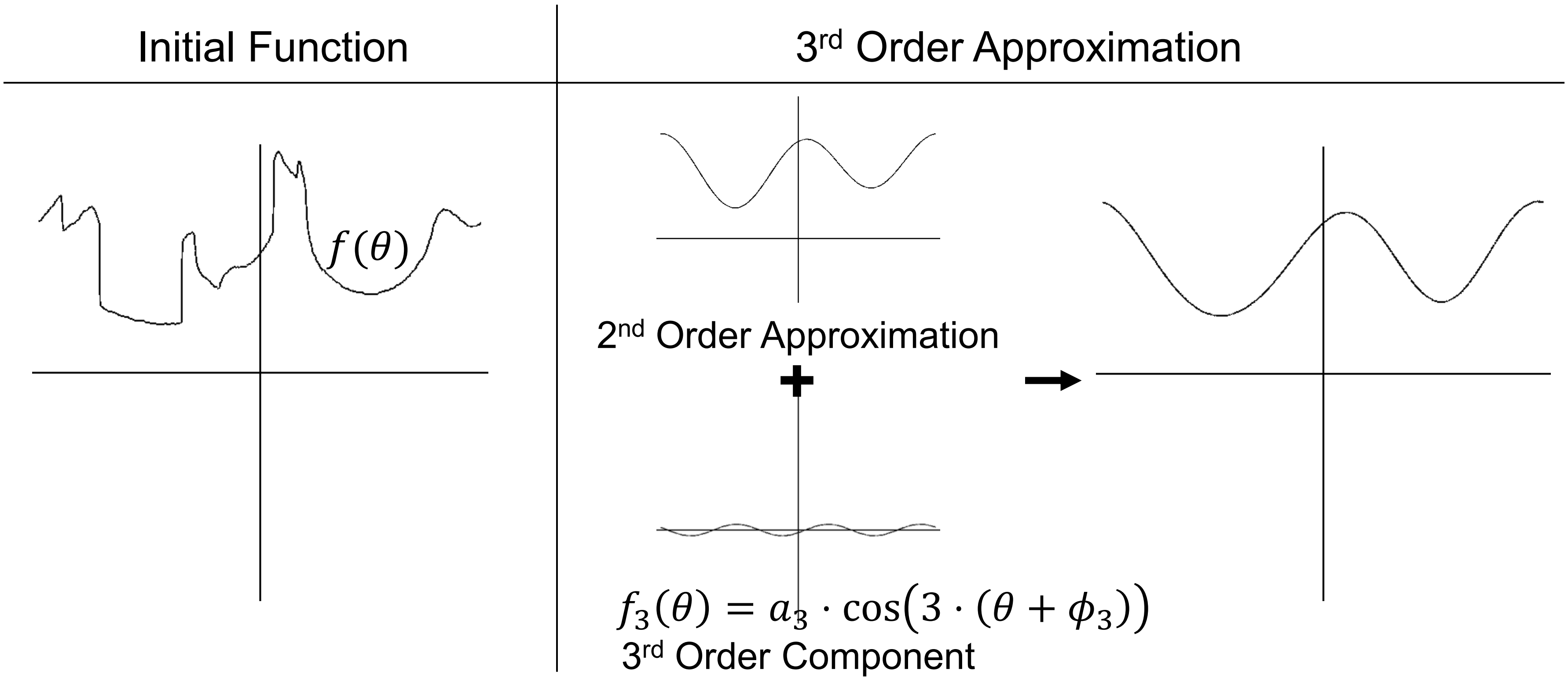
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of <u>scaled</u> and <u>shifted</u> cosine functions.

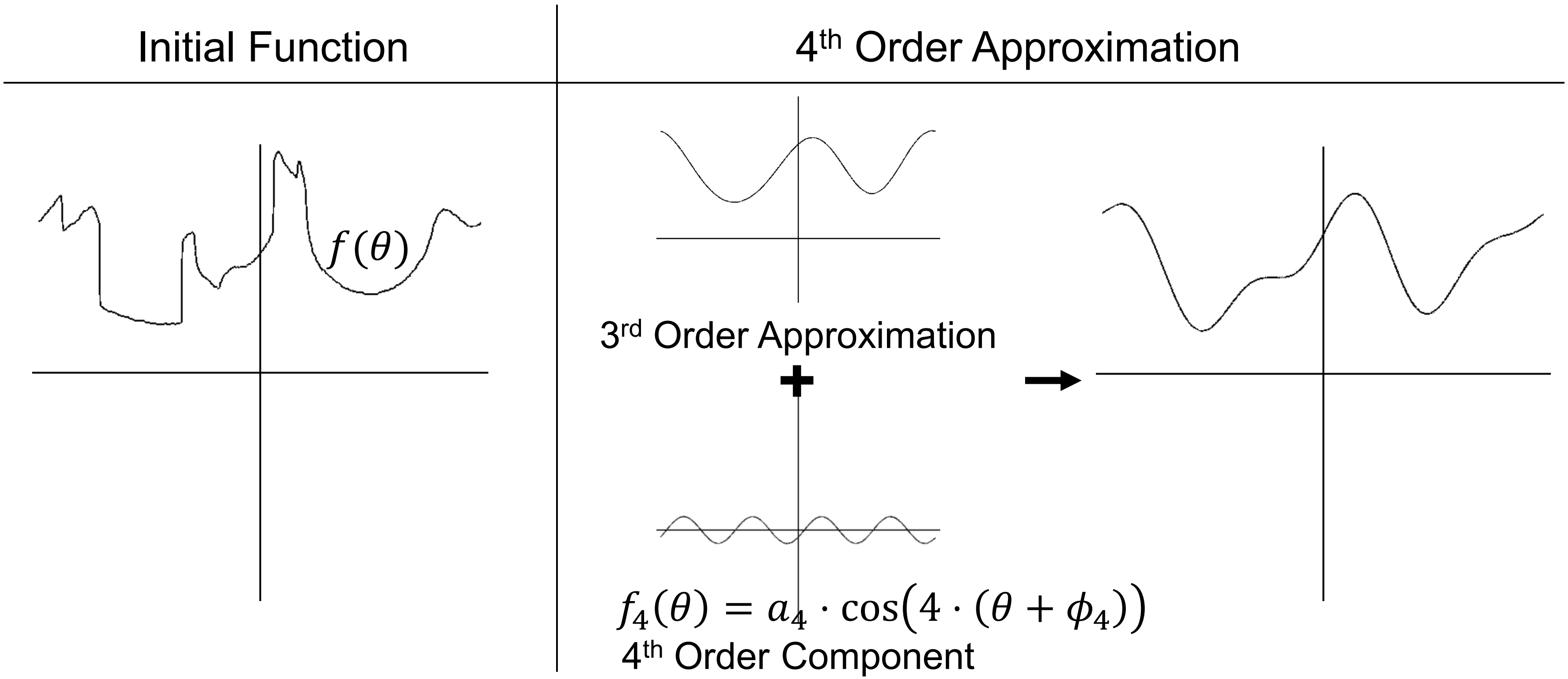
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of scaled and shifted cosine functions.

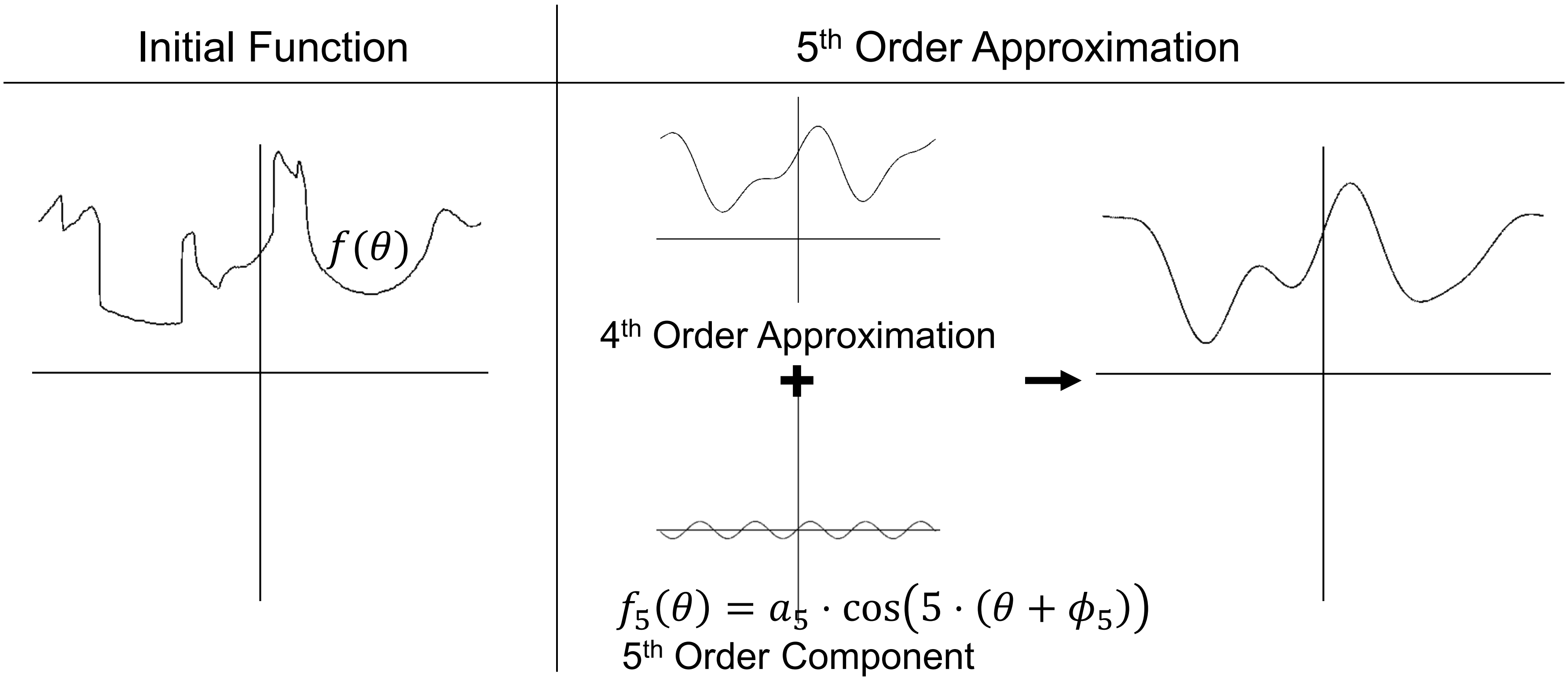
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of <u>scaled</u> and <u>shifted</u> cosine functions.

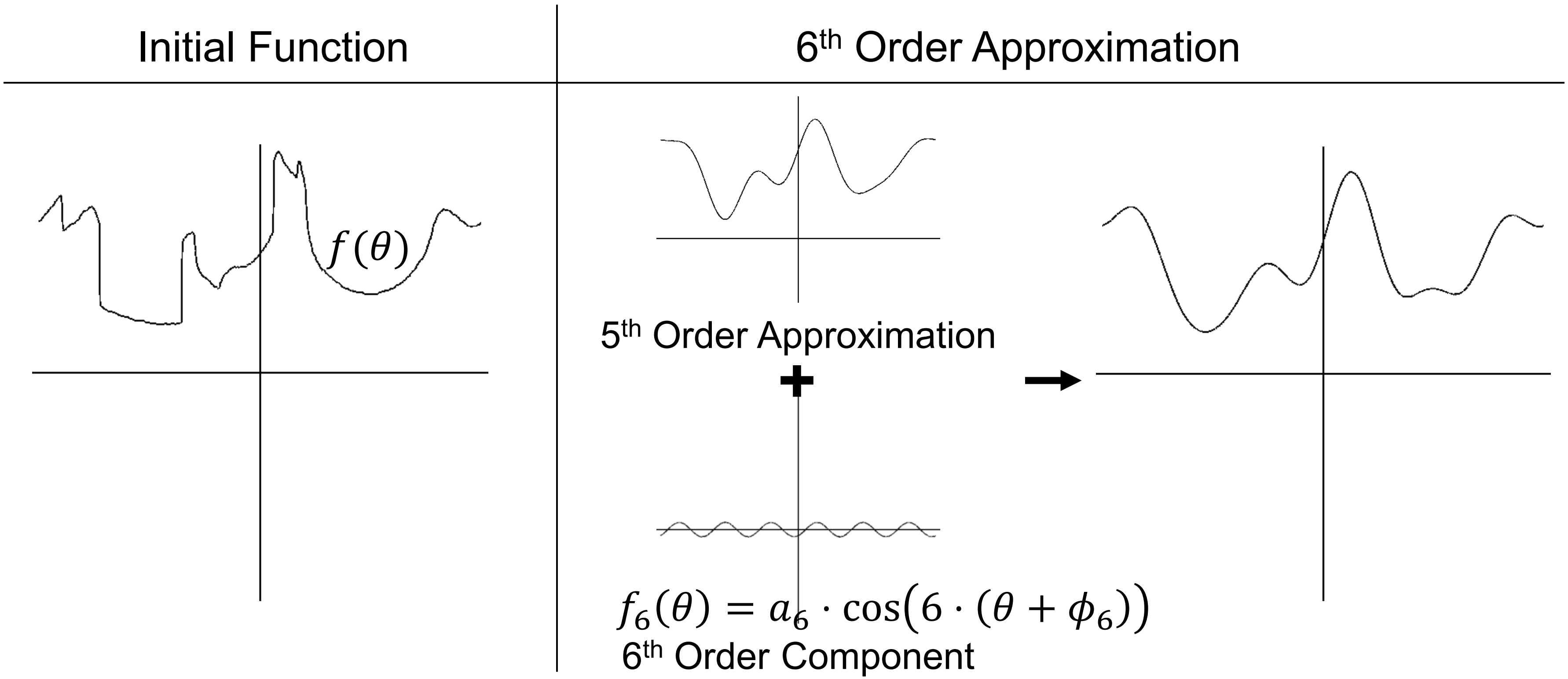
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of scaled and shifted cosine functions.

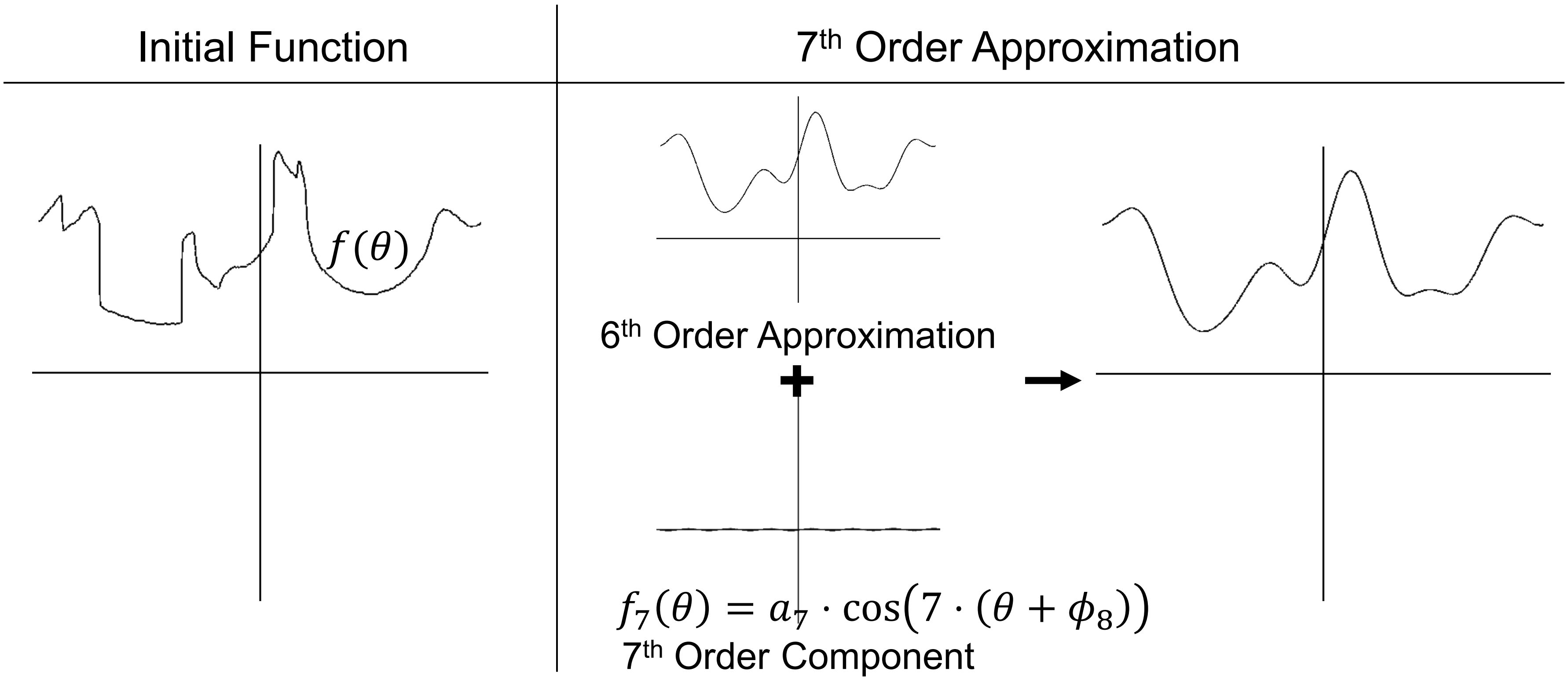
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of <u>scaled</u> and <u>shifted</u> cosine functions.

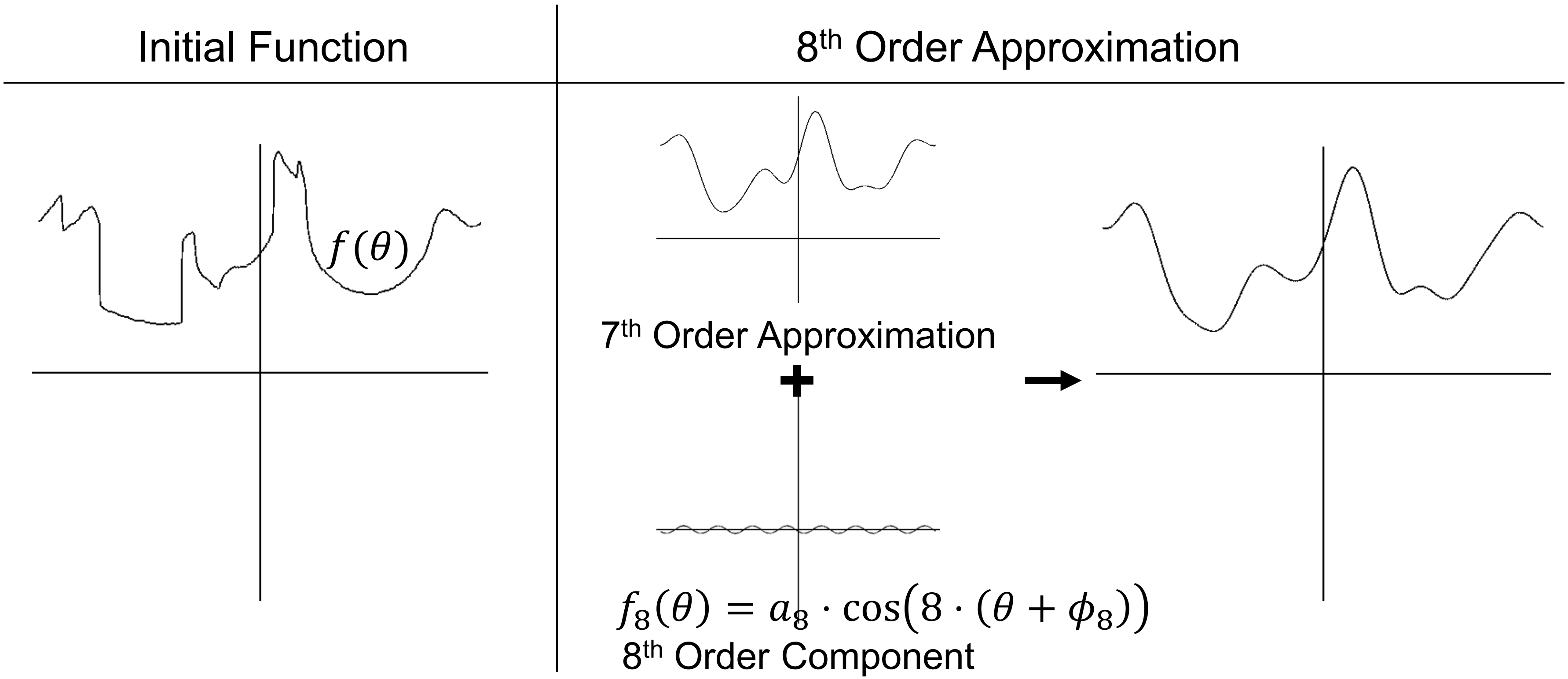
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of scaled and shifted cosine functions.

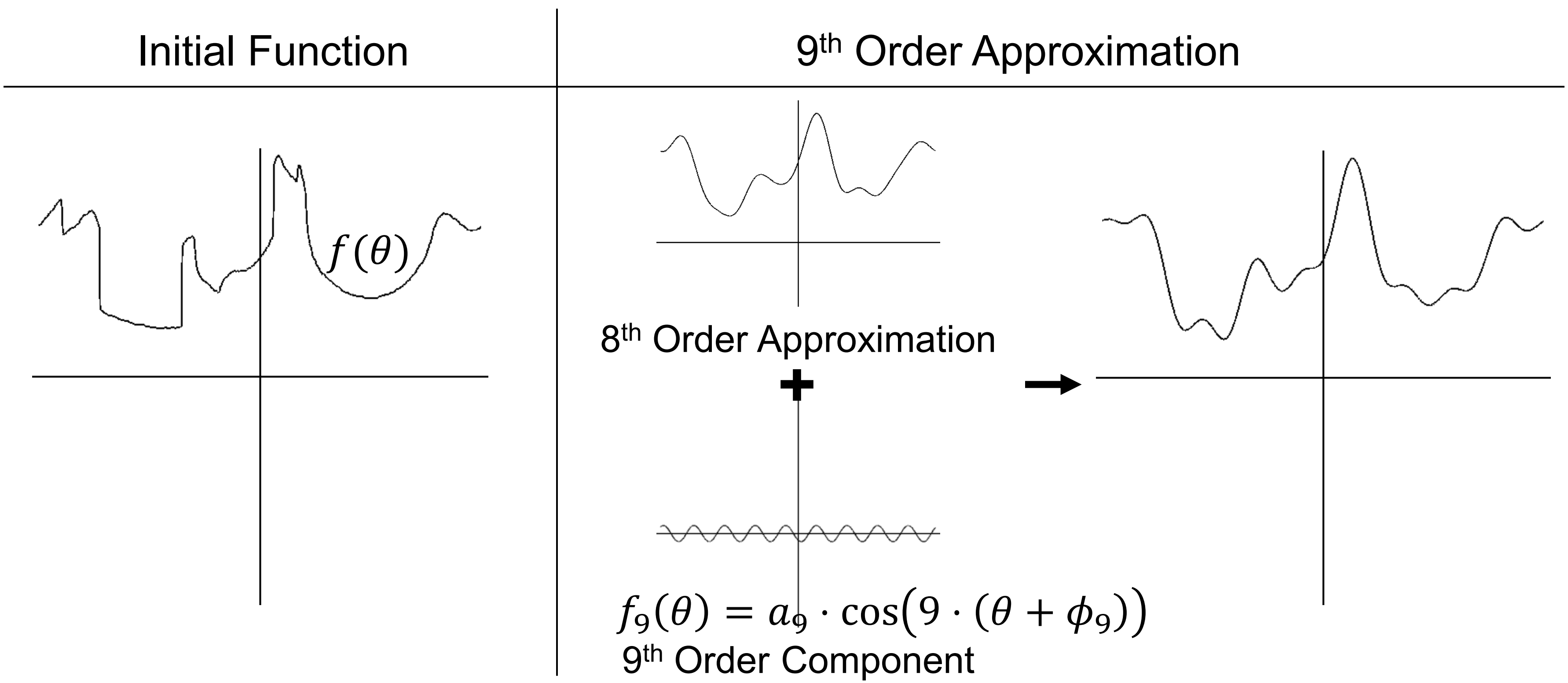
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of <u>scaled</u> and <u>shifted</u> cosine functions.

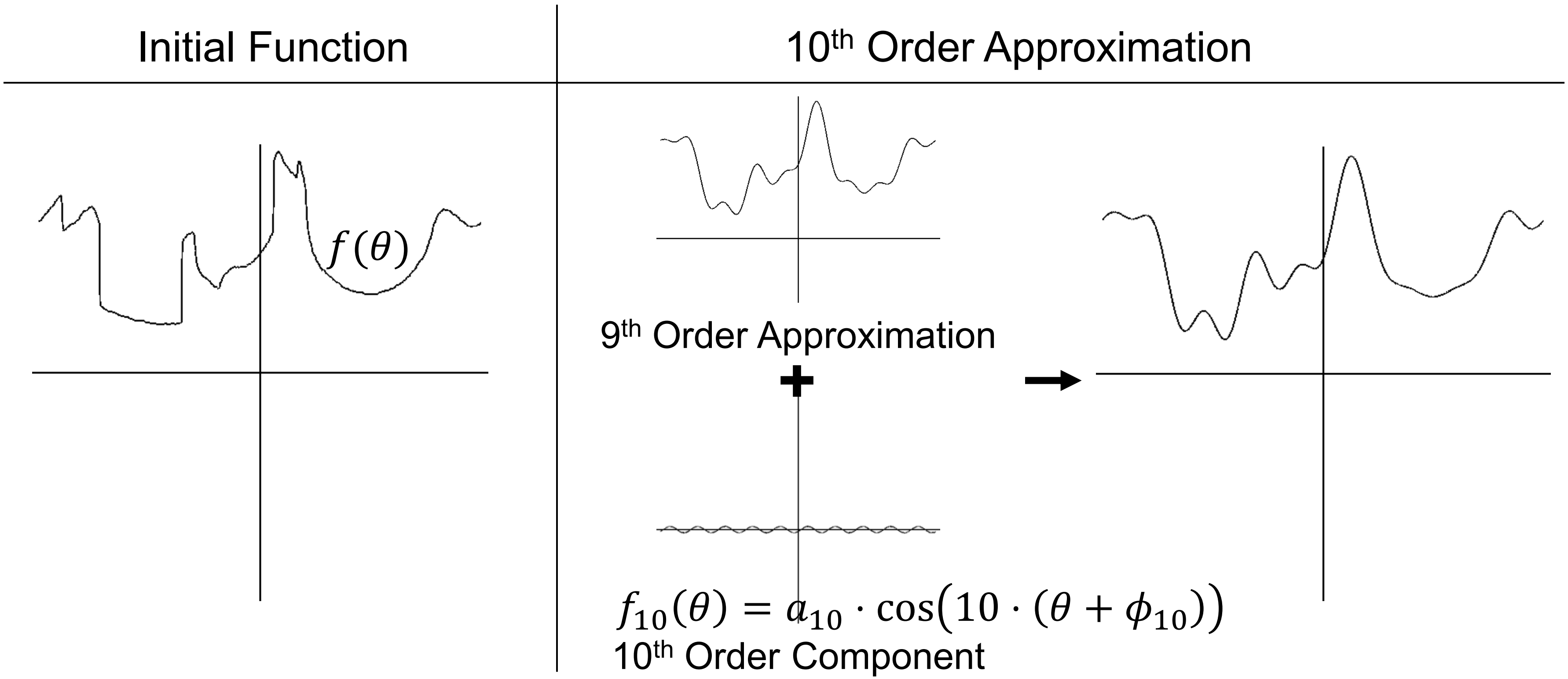
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of scaled and shifted cosine functions.

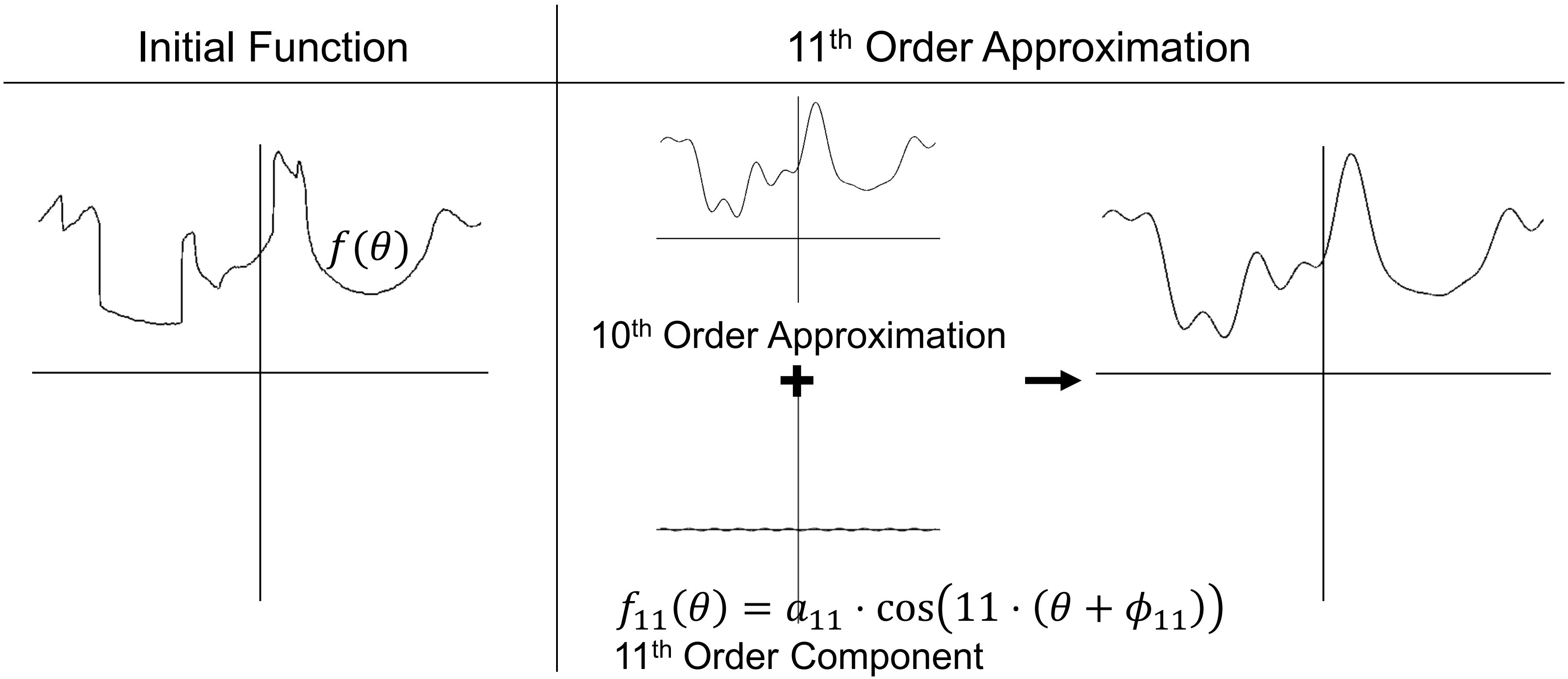
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of scaled and shifted cosine functions.

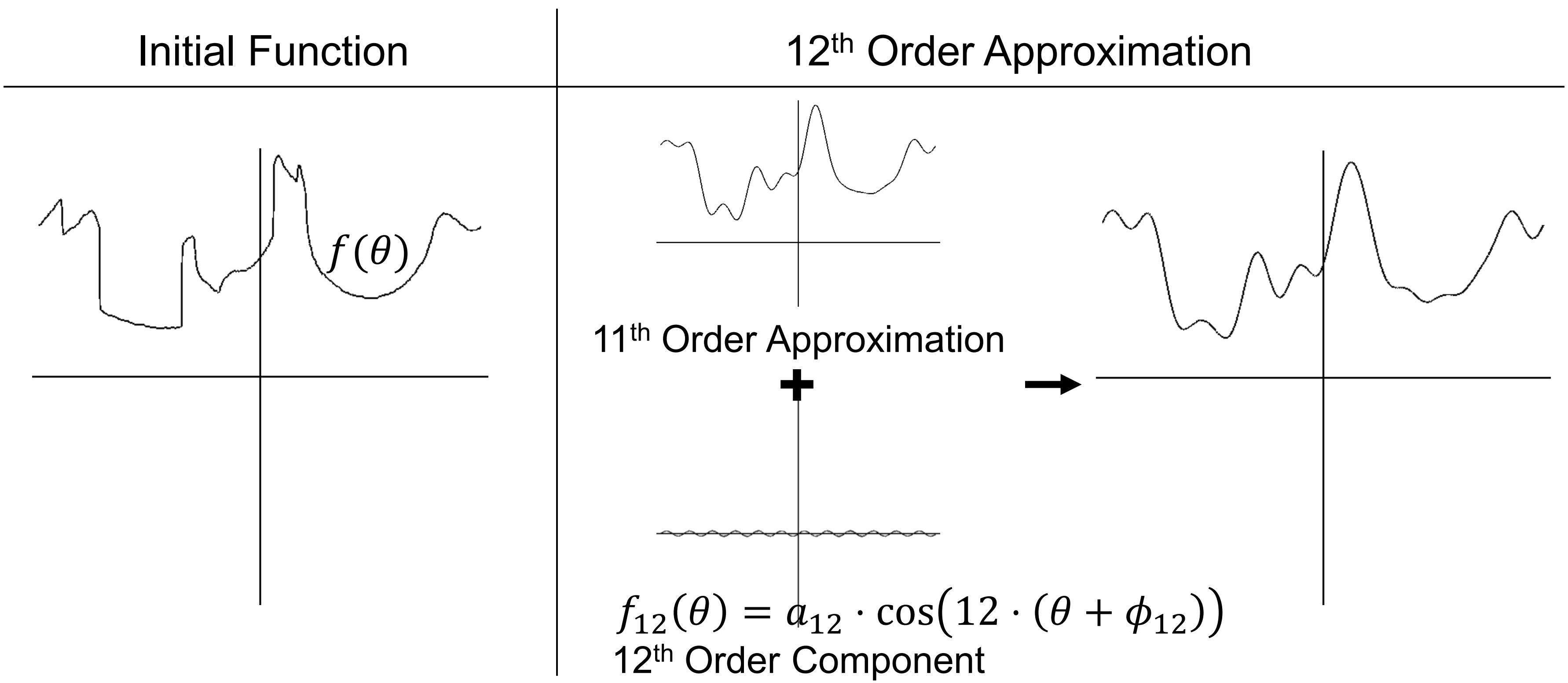
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of <u>scaled</u> and <u>shifted</u> cosine functions.

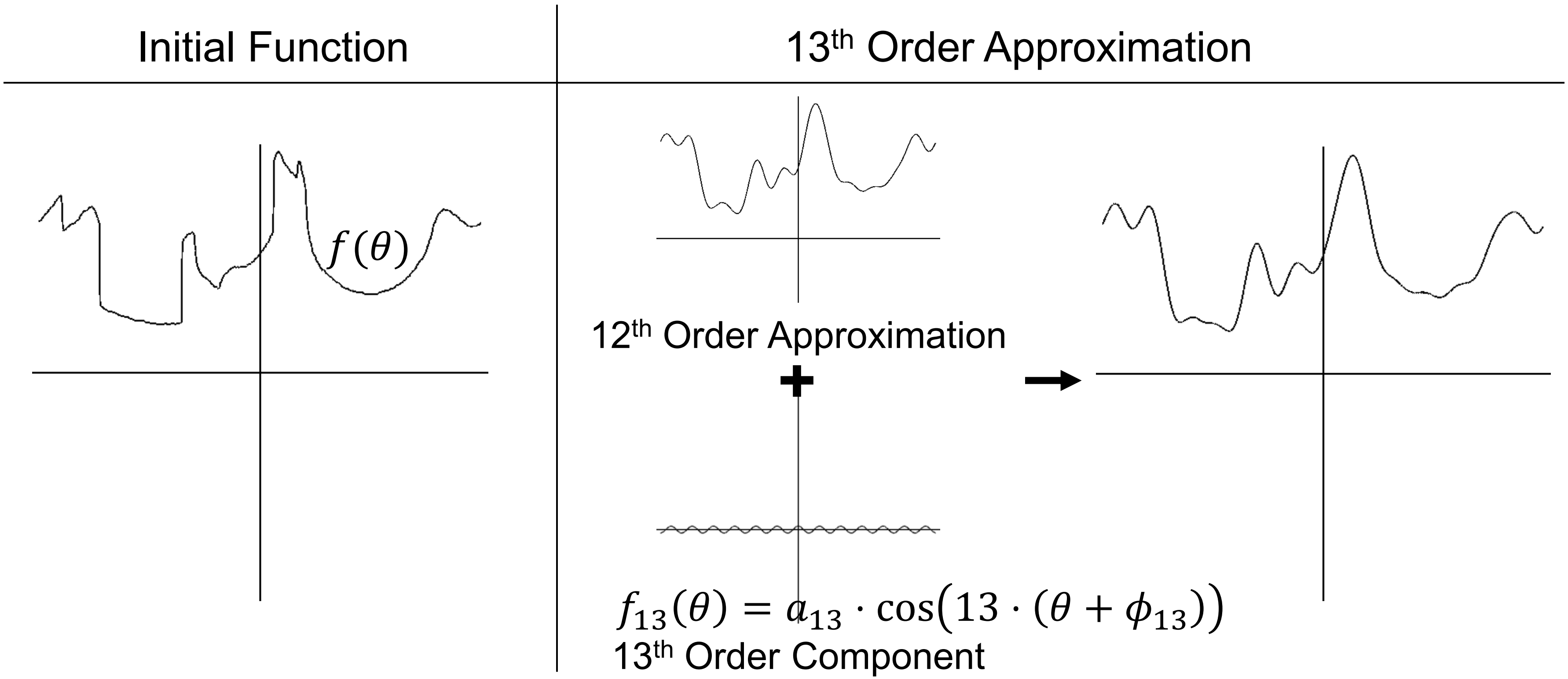
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of scaled and shifted cosine functions.

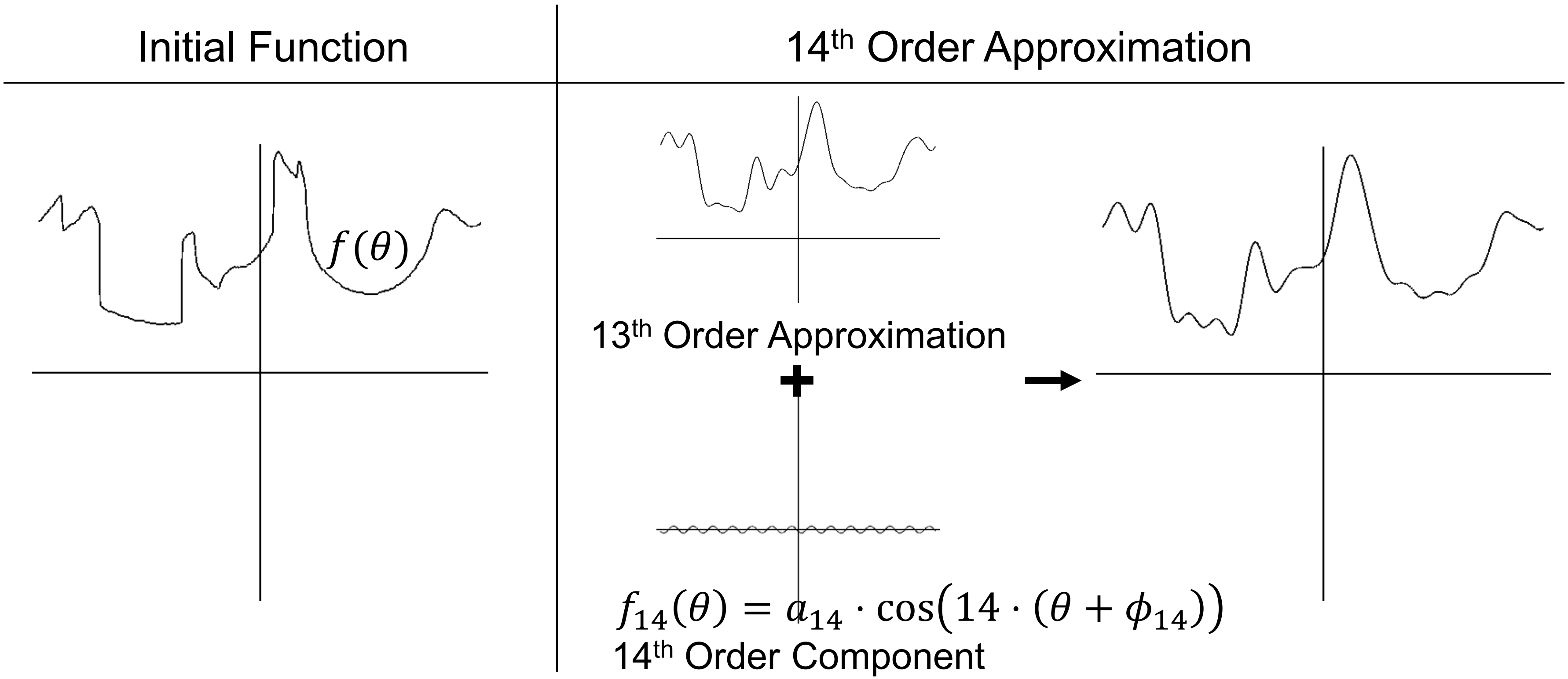
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of <u>scaled</u> and <u>shifted</u> cosine functions.

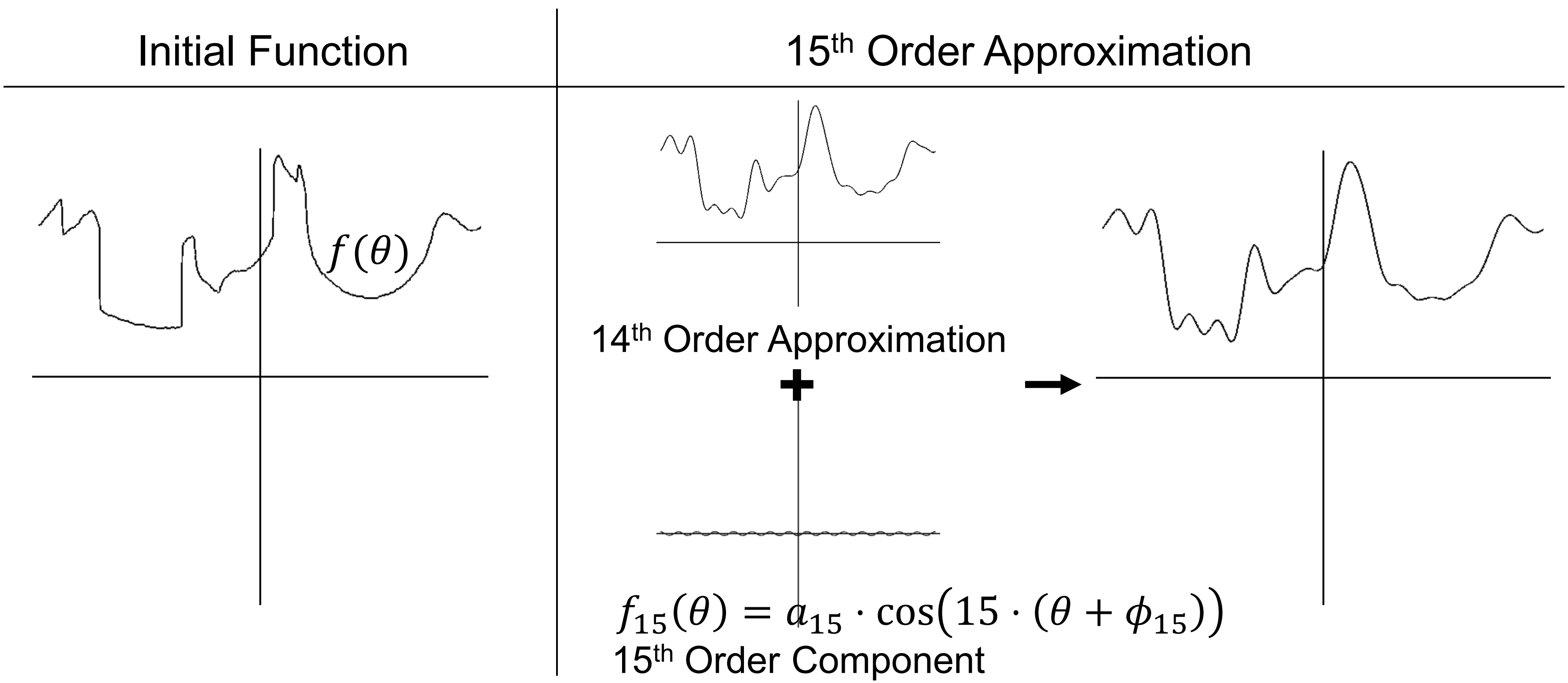
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of scaled and shifted cosine functions.

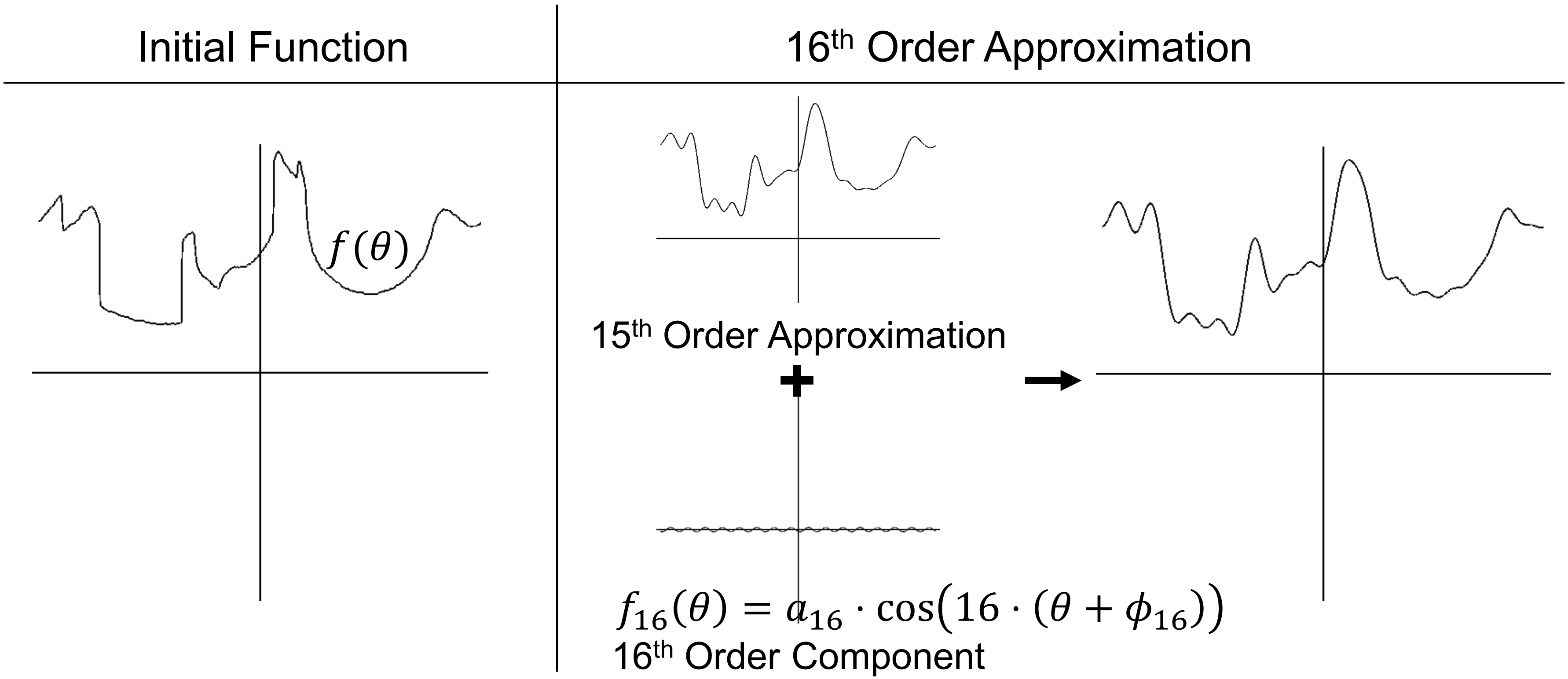
- As higher frequency components are added, finer details are captured.



Fourier Analysis

Uniquely describes a signal as a sum of <u>scaled</u> and <u>shifted</u> cosine functions.

- As higher frequency components are added, finer details are captured.



Fourier Analysis

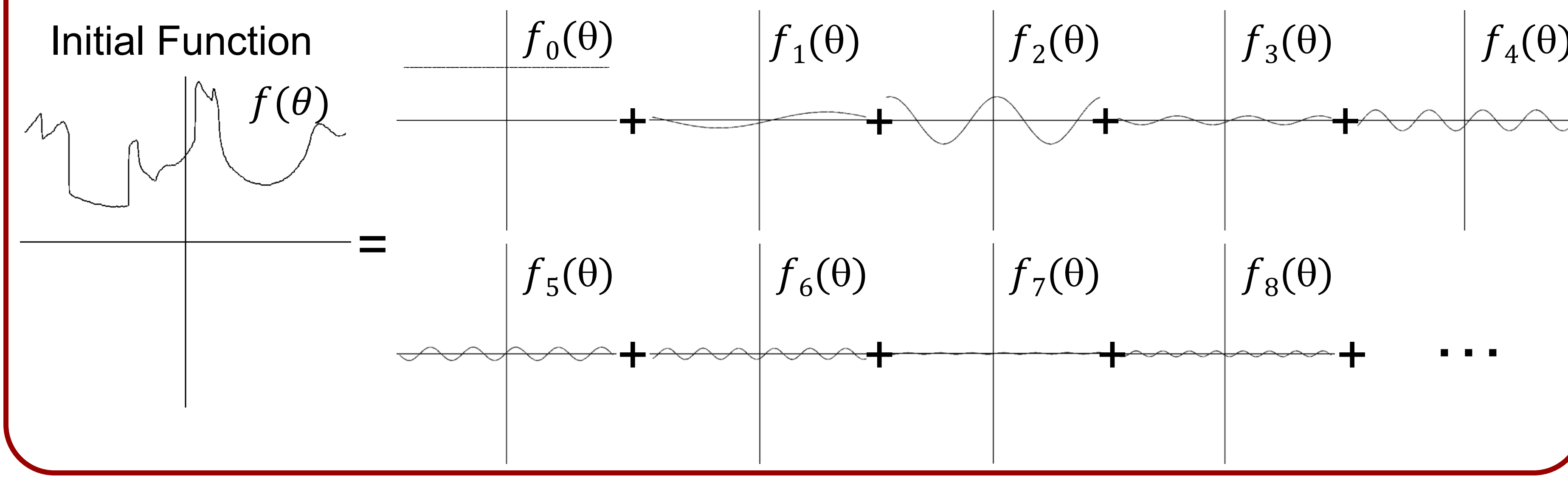
Uniquely describes a signal as a sum of <u>scaled</u> and <u>shifted</u> cosine functions.

In the limit, we "reproduce" the initial function:

$$f(\theta) = \sum_{k=0}^{\infty} a_k \cdot \cos\big(k(\theta + \phi_k)\big)$$

a_k: *amplitude* of the k^{th} frequency component

ϕ_k: *phase shift* of the k^{th} frequency component



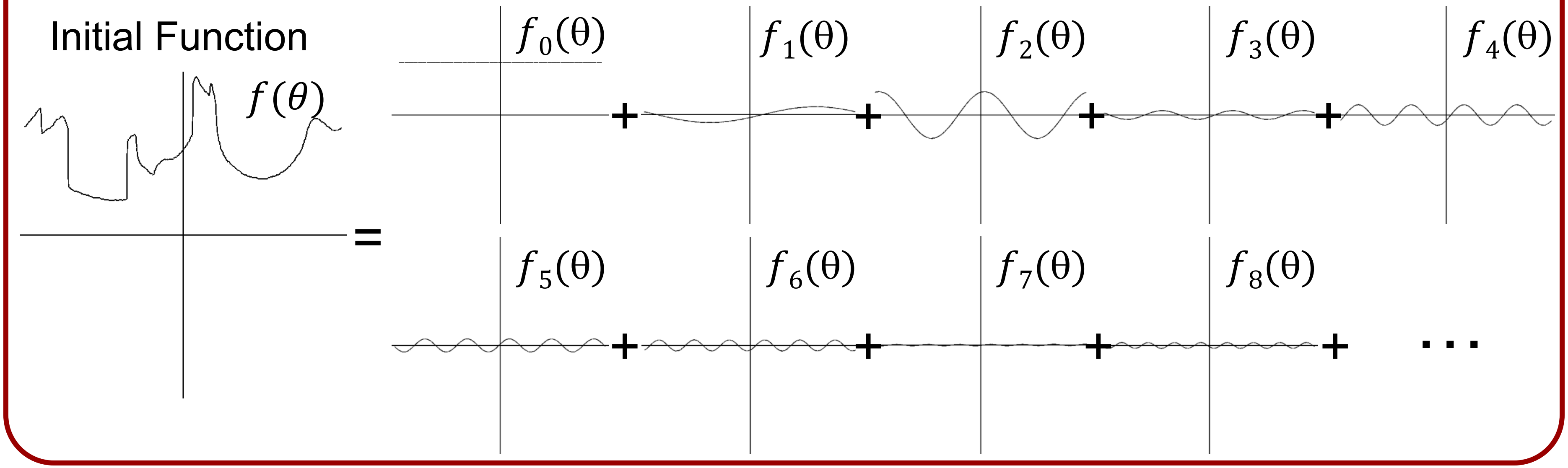
Question

<u>Goal</u>:

Fit a low-frequency signal to the m samples.

<u>Question</u>:

To fit to the m samples, what is the smallest number of (low) frequencies we need to use?



Question

Goal:

Fit a low-frequency signal to the m samples.

Question:

To fit to the m samples, what is the smallest number of (low) frequencies we need to use?

Answer:

(Except for the first) each frequency component has two degrees of freedom – amplitude and phase shift.

⇒ Each additional frequency component allows us to satisfy two more (sample) constraints

⇒ With $m/2$ (lowest) frequencies, we can fit m samples.

Sampling Theorem

Terminology:

- A signal is **band-limited** if its highest non-zero frequency is bounded (i.e. less than infinity).
- That frequency is called the **bandwidth**.

Having m samples

$$\Updownarrow$$

Having a signal width bandwidth $m/2$

Image Sampling

To reconstruct the continuous function from m_{in} inputes samples, we can find the unique function of frequency $m_{in}/2$ that interpolates the values.

Q: Why don't we just evaluate this function at the m_{out} output sample positions?

A: If $m_{out} < m_{in}$ we don't have sufficient samples!

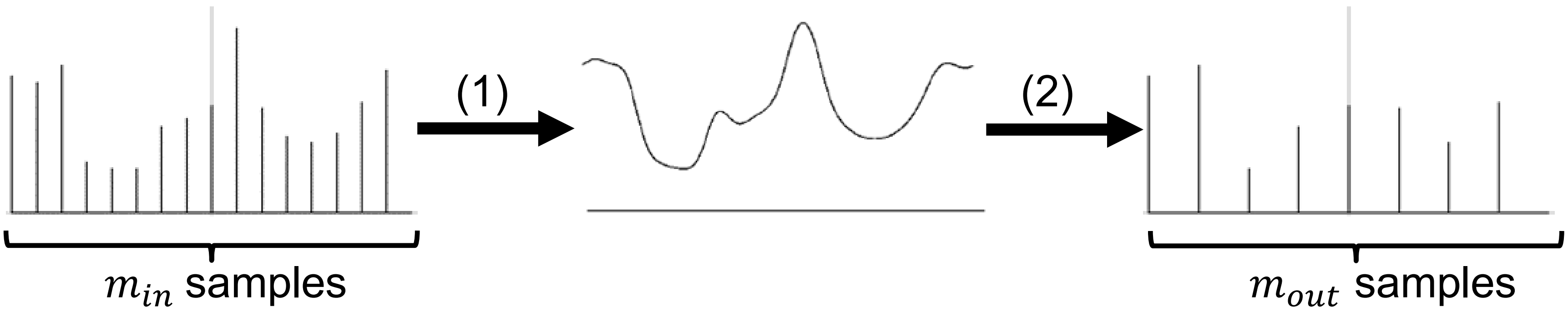


Image Sampling

Aliasing

When a high-frequency signal is sampled with too few samples, it masks/aliases as a lower-frequency signal.

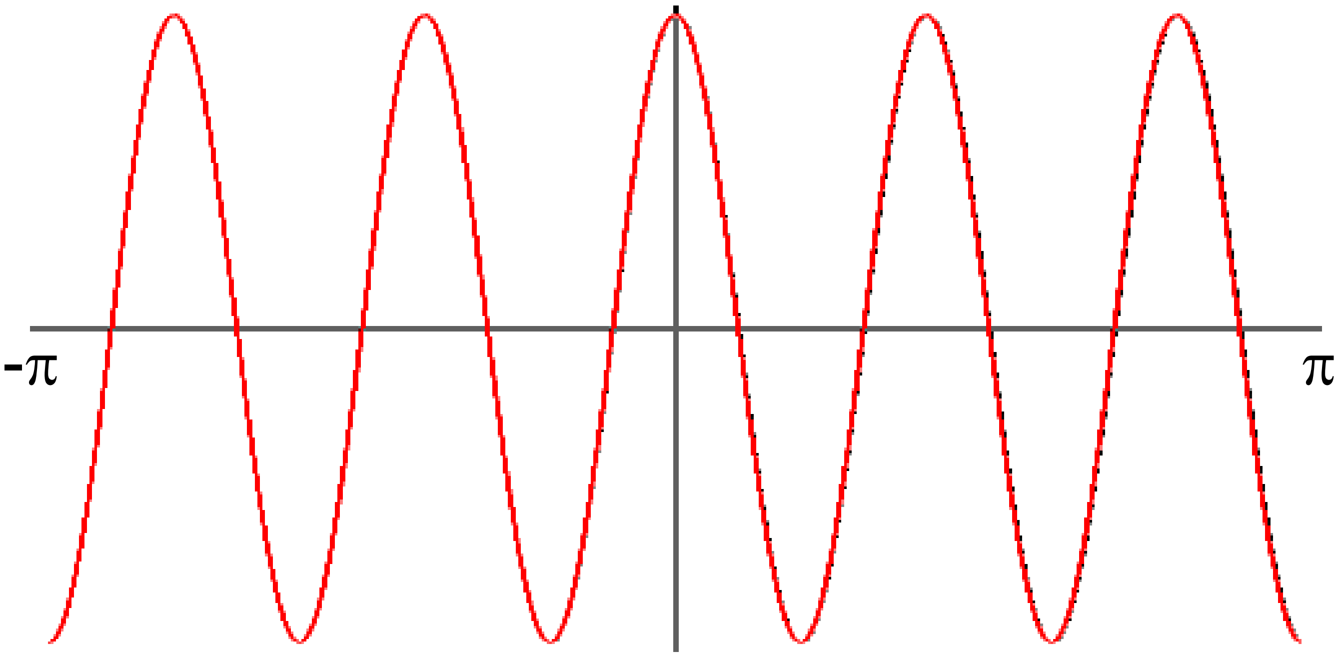
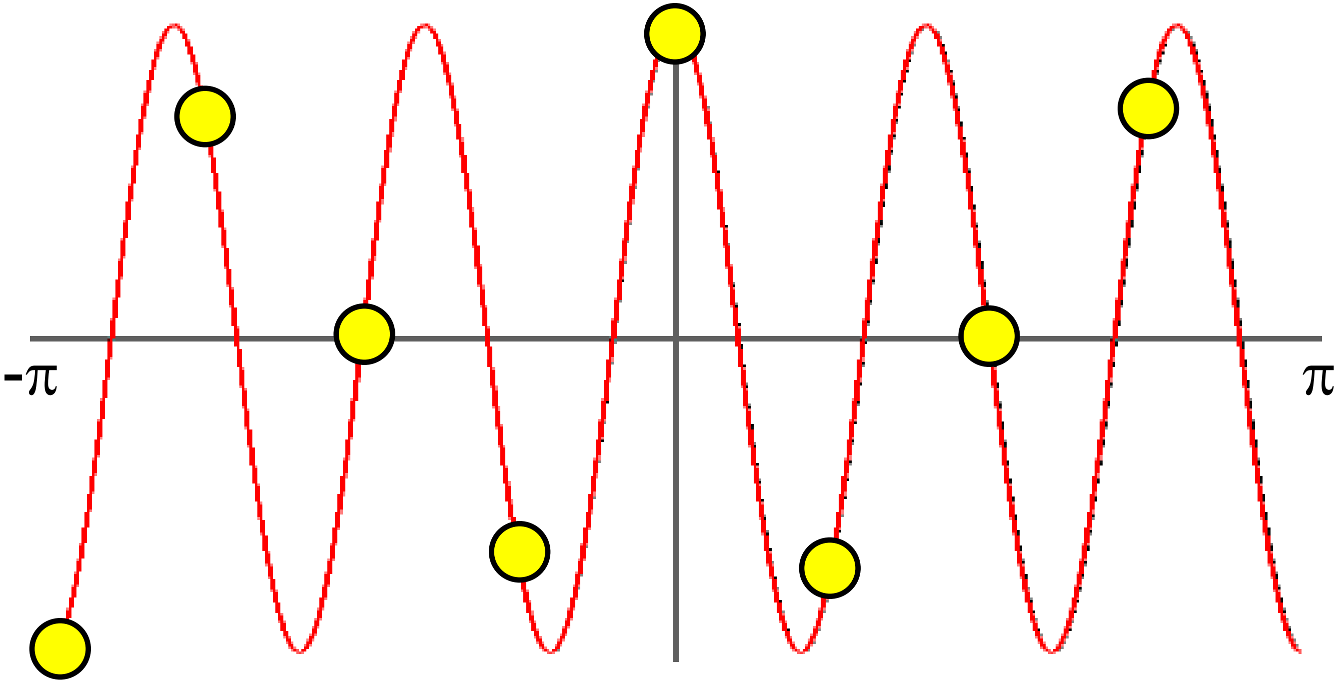


Image Sampling

Aliasing

When a high-frequency signal is sampled with too few samples, it masks/aliases as a lower-frequency signal.

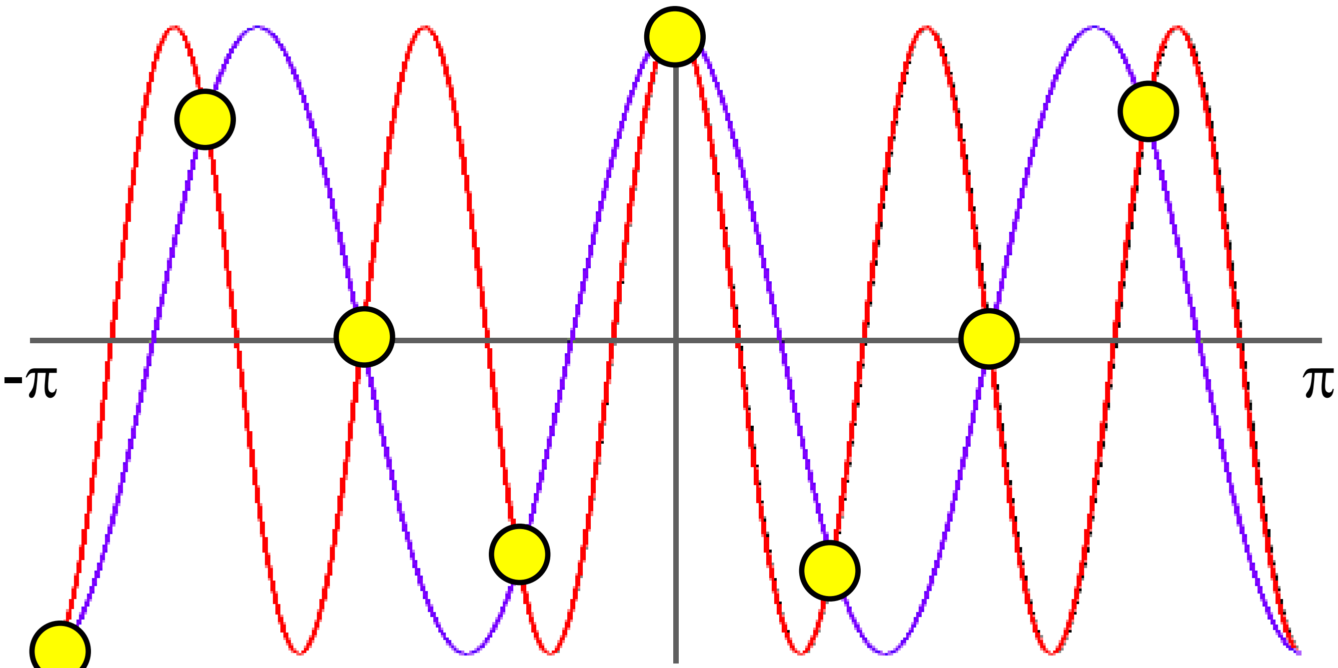


Eight samples of a frequency-5 function (need at least 10).

Image Sampling

Aliasing

When a high-frequency signal is sampled with too few samples, it masks/aliases as a lower-frequency signal.

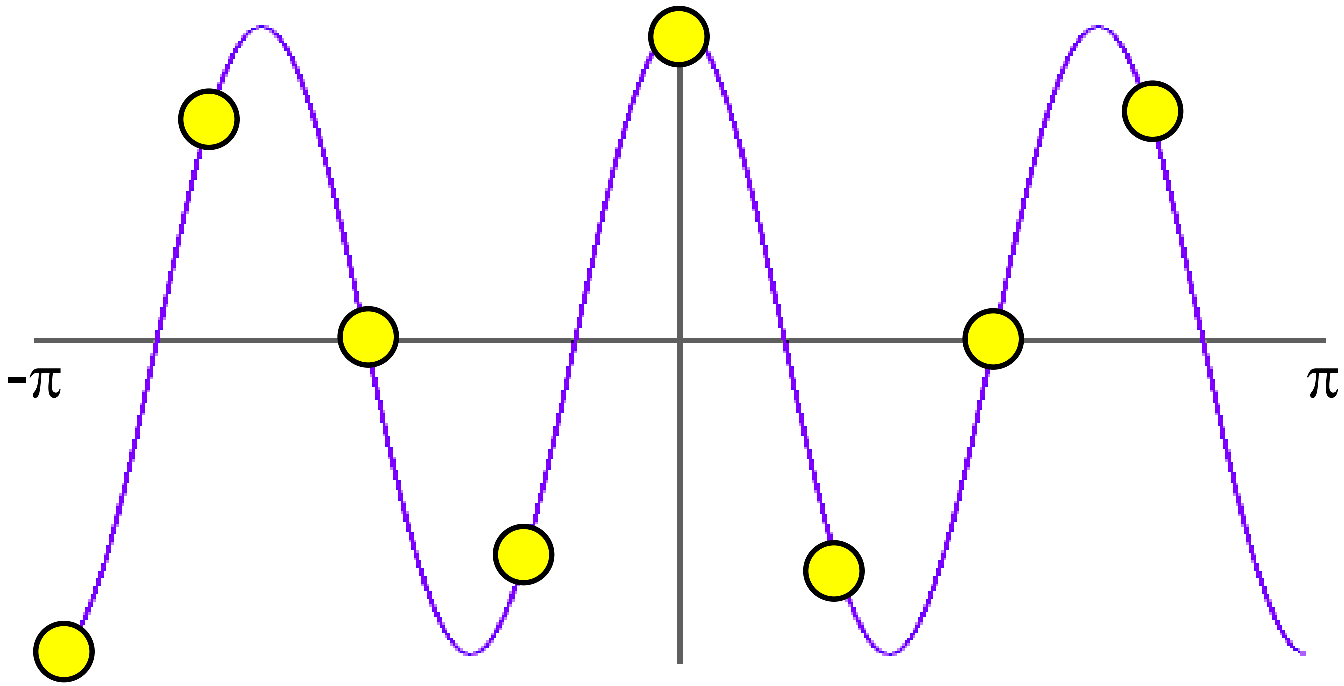


Eight samples of a frequency-5/3 function (need at least 10).

Image Sampling

Aliasing

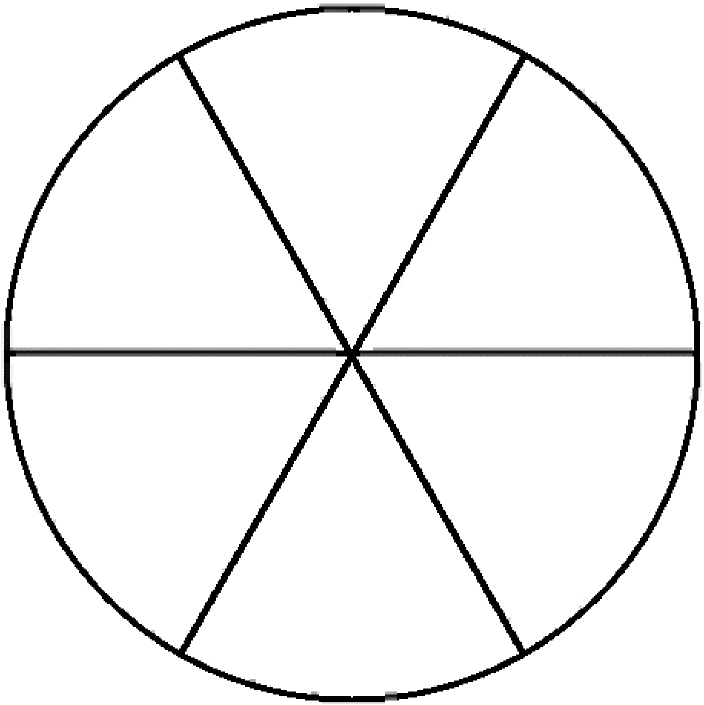
When a high-frequency signal is sampled with too few samples, it <u>masks</u>/<u>aliases</u> as a lower-frequency signal.



Eight samples of a frequency-3 function (need at least 10).

Temporal Aliasing

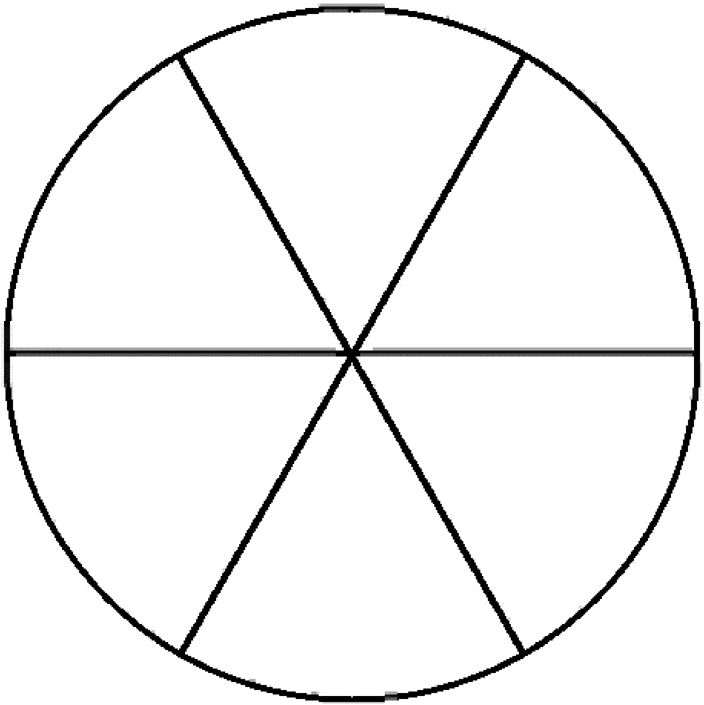
- Artifacts due to limited temporal resolution



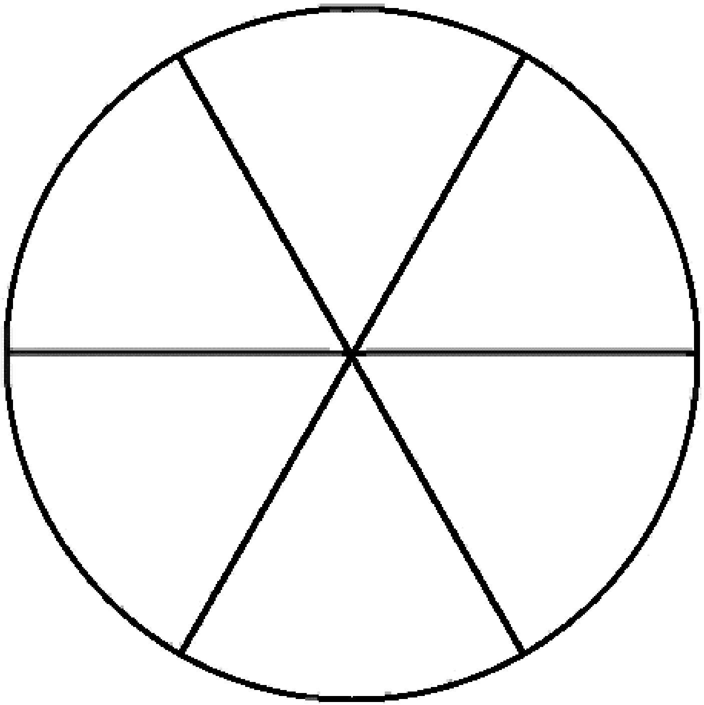
10 fps

Temporal Aliasing

- Artifacts due to limited temporal resolution



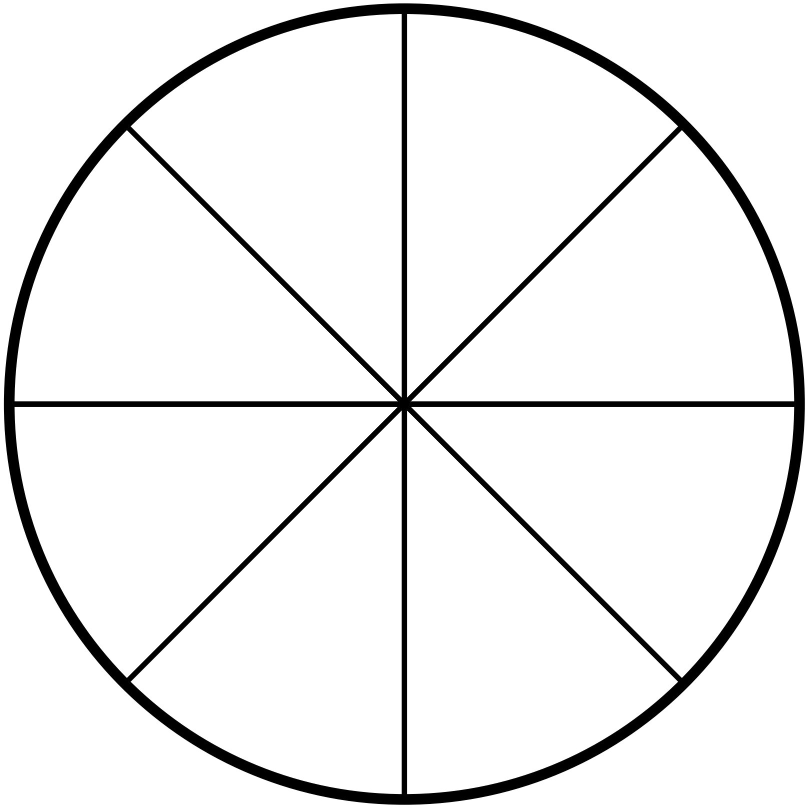
10 fps



25 fps

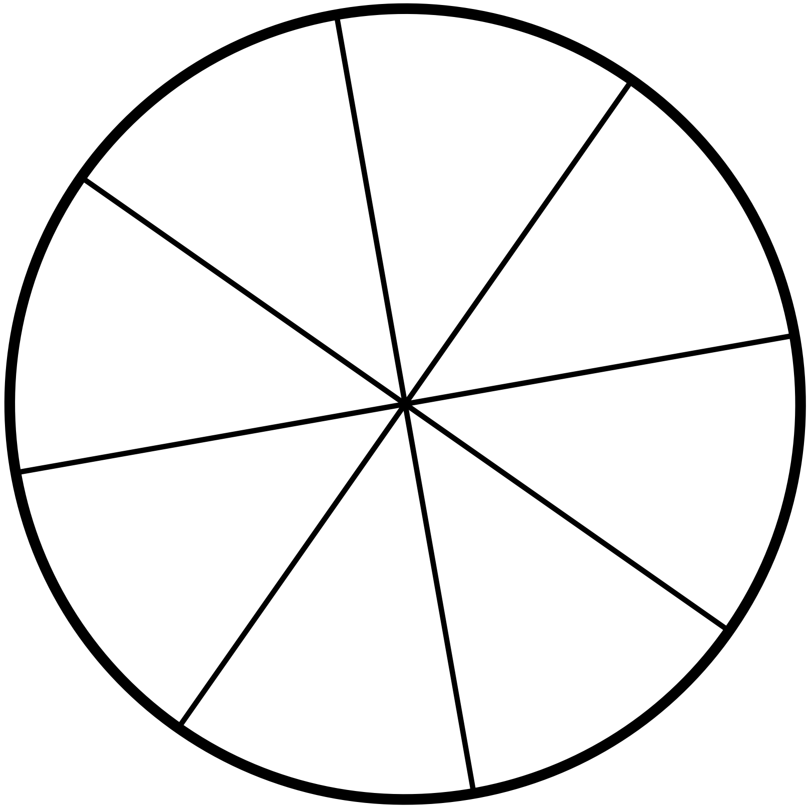
Temporal Aliasing

- Artifacts due to limited temporal resolution



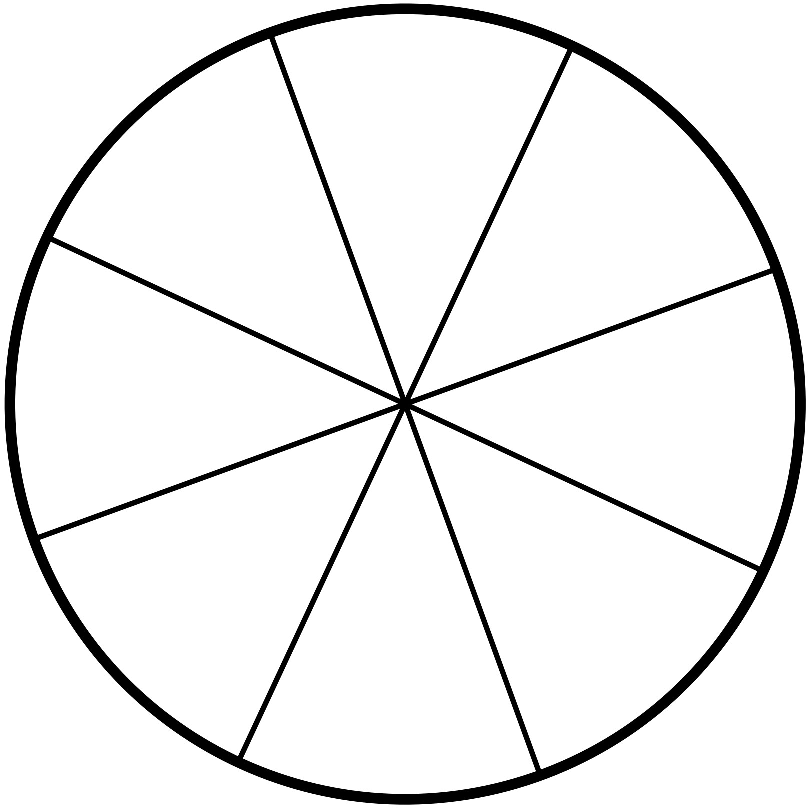
Temporal Aliasing

- Artifacts due to limited temporal resolution



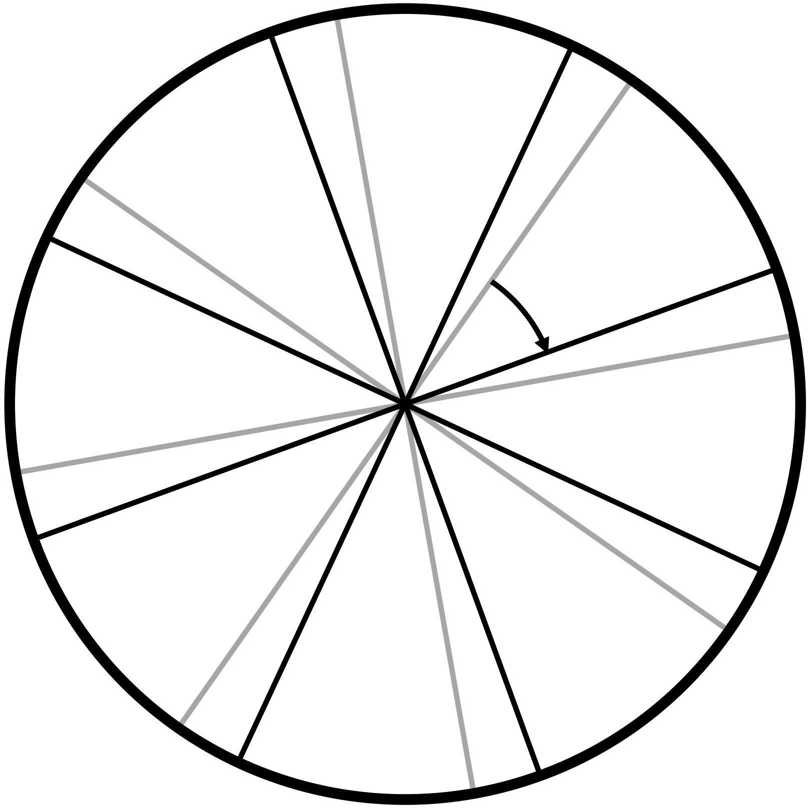
Temporal Aliasing

- Artifacts due to limited temporal resolution



Temporal Aliasing

- Artifacts due to limited temporal resolution



Sampling

There are two problems:

1. Don’t have enough samples to correctly reconstruct/represent the higher-frequency information
2. **Corrupt** the low-frequency information because the higher-frequencies mask themselves as lower ones.

Anti-Aliasing

Two possible ways to address aliasing:

- Sample at higher rate
- Pre-filter to form band-limited signal

Anti-Aliasing

Two possible ways to address aliasing:

- Sample at higher rate
  - Not always possible (e.g. fixed-resolution displays)
- Pre-filter to form band-limited signal

Anti-Aliasing

Two possible ways to address aliasing:

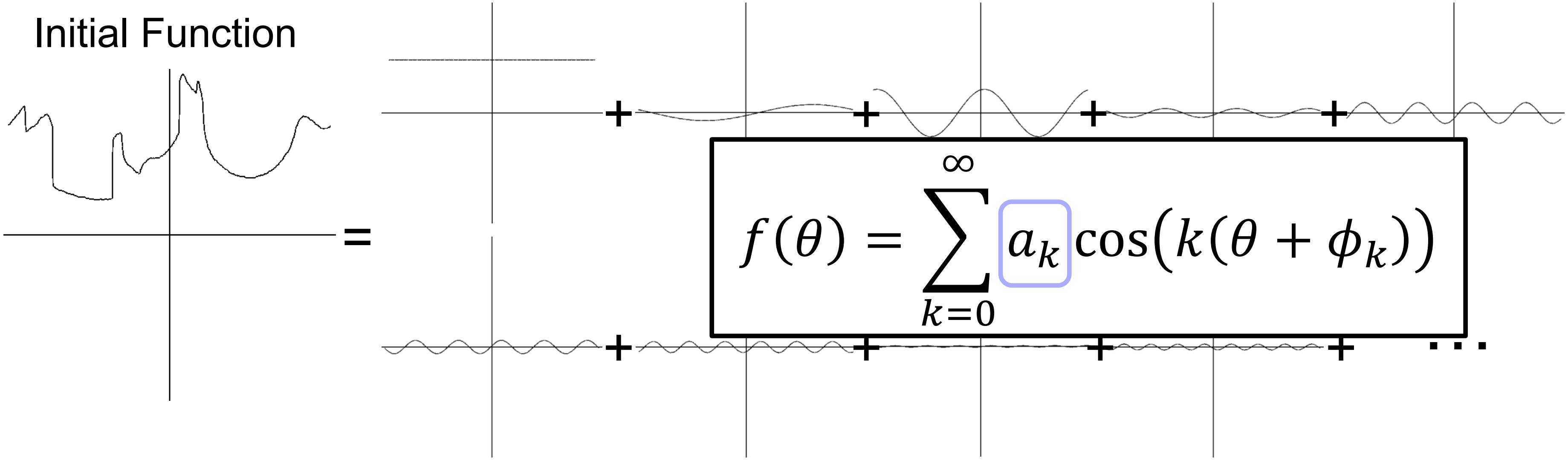
- Sample at higher rate
- Pre-filter to form an appropriately band-limited signal
  - You still don't get your high frequencies, but the low frequencies are not corrupted.

Recall:

Sampling with a wider Gaussian has the effect of smoothing out higher frequencies

Fourier Analysis

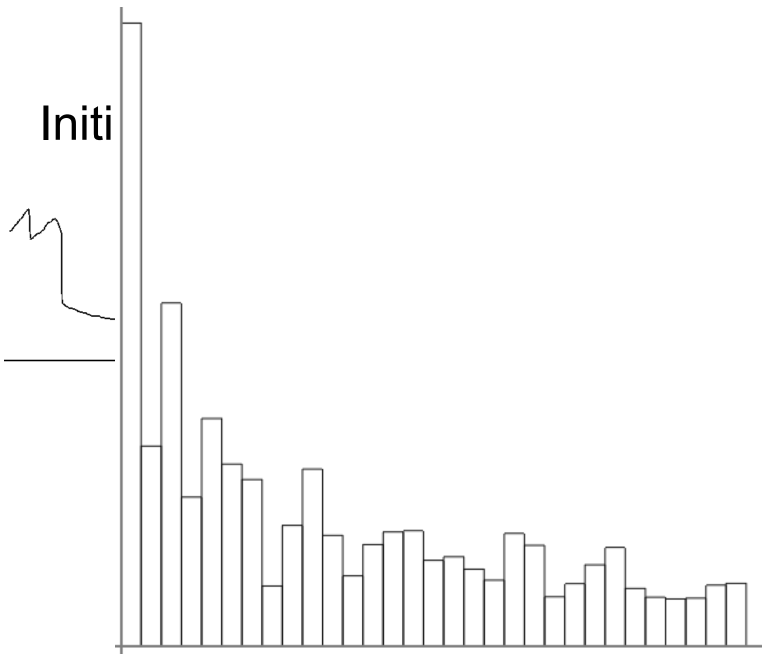
If we look at the amplitude at each frequency, we obtain the **power spectrum** of the signal:



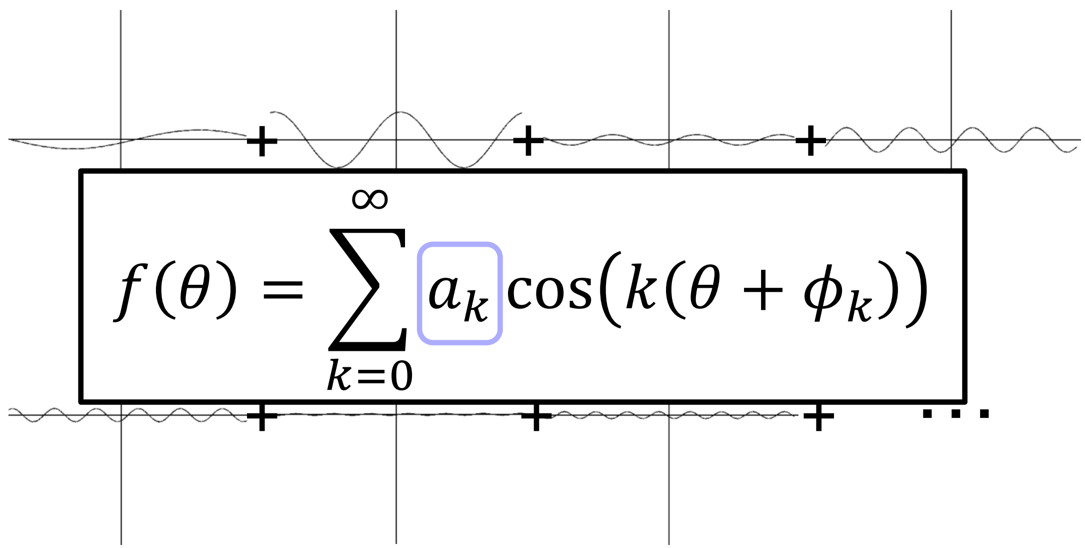
$$f(\theta) = \sum_{k=0}^{\infty} a_k \cos\left(k(\theta + \phi_k)\right)$$

Fourier Analysis

If we look at the amplitude at each frequency, we obtain the **power spectrum** of the signal:



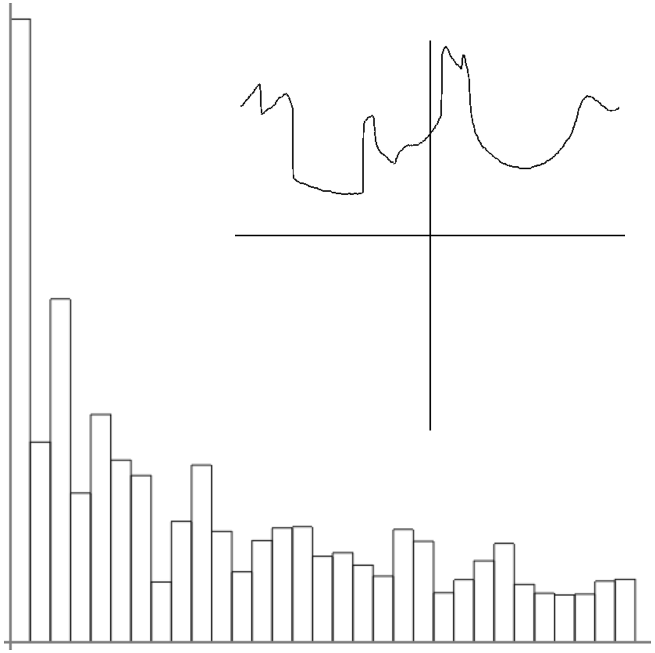
Power Spectrum



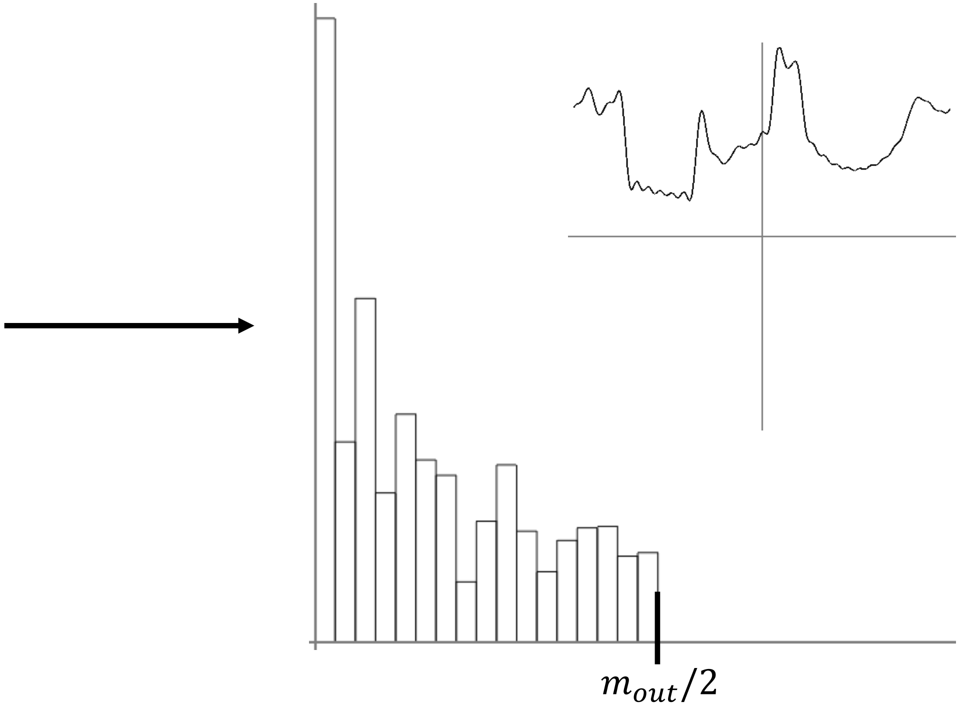
$$f(\theta) = \sum_{k=0}^{\infty} a_k \cos\big(k(\theta + \phi_k)\big)$$

Pre-Filtering

To avoid aliasing when sampling with m_{out} samples, we should first band-limit by discarding the high-frequency (greater than $m_{out}/2$) components.



Initial Spectrum

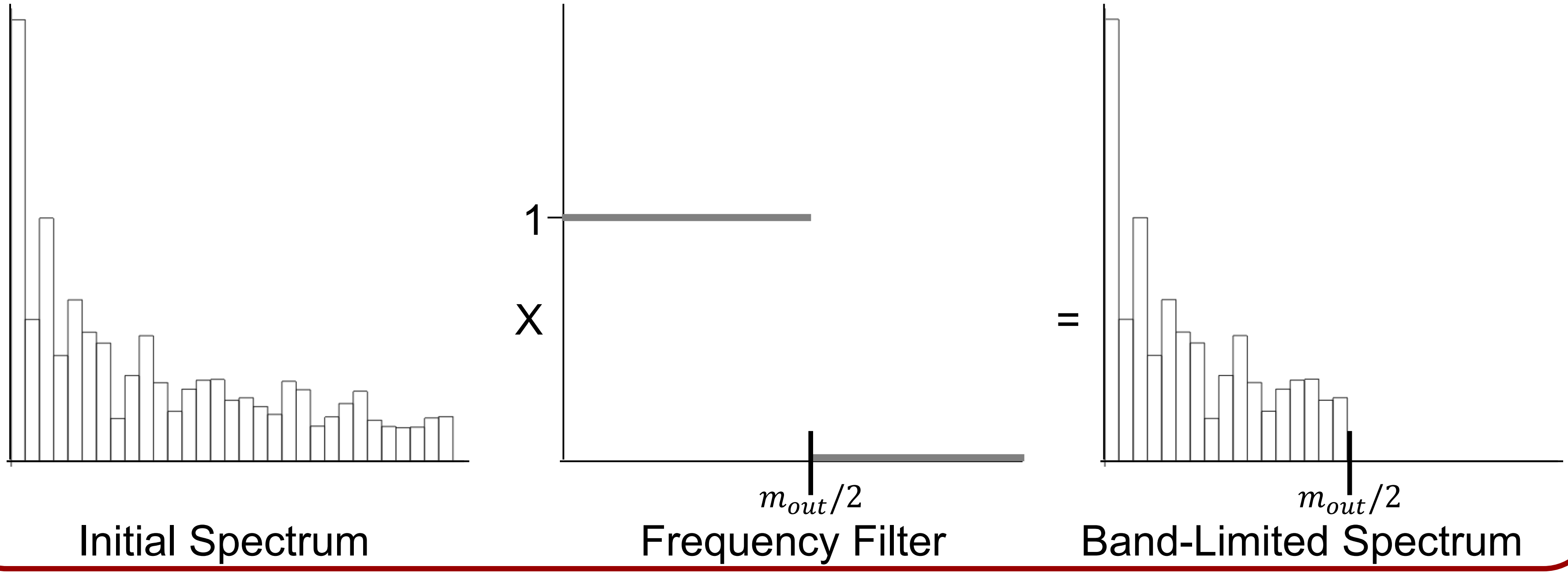


Band-Limited Spectrum

Pre-Filtering

To avoid aliasing when sampling with m_{out} samples, we should first band-limit by discarding the high-frequency (greater than $m_{out}/2$) components.

We could do this if we could **multiply** the frequency components by a 0/1 function:



Pre-Filtering

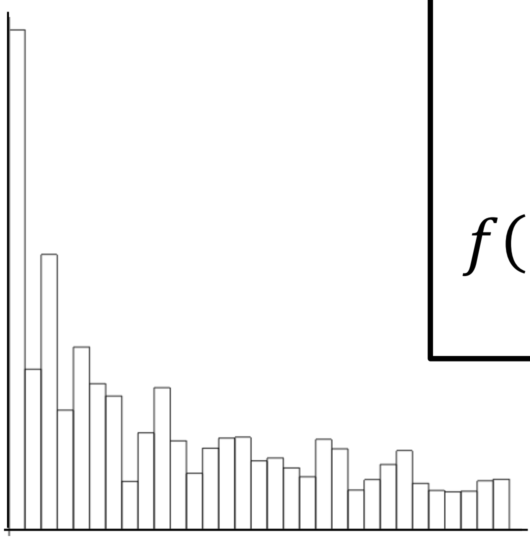
To avoid aliasing when sampling with m_{out} samples, we should first band-limit by discarding the high-frequency (greater than $m_{out}/2$) components.

We could do frequency components

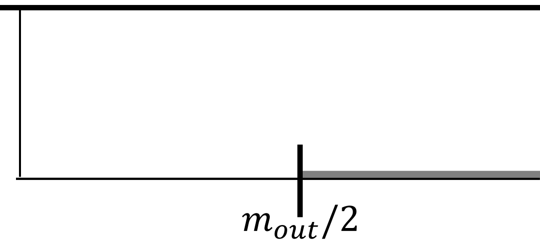
$$f(\theta) = \sum_{k=0}^{\infty} a_k \cos\big(k(\theta + \phi_k)\big)$$

$$\Downarrow$$

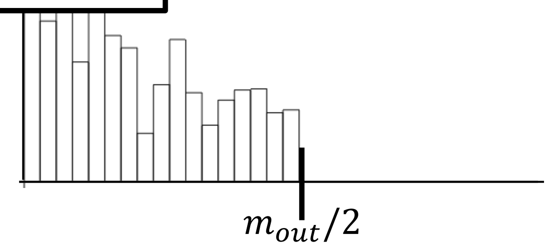
$$f(\theta) = \sum_{k=0}^{m_{out}/2} a_k \cos\big(k(\theta + \phi_k)\big)$$



x



=



Initial Spectrum

Frequency Filter

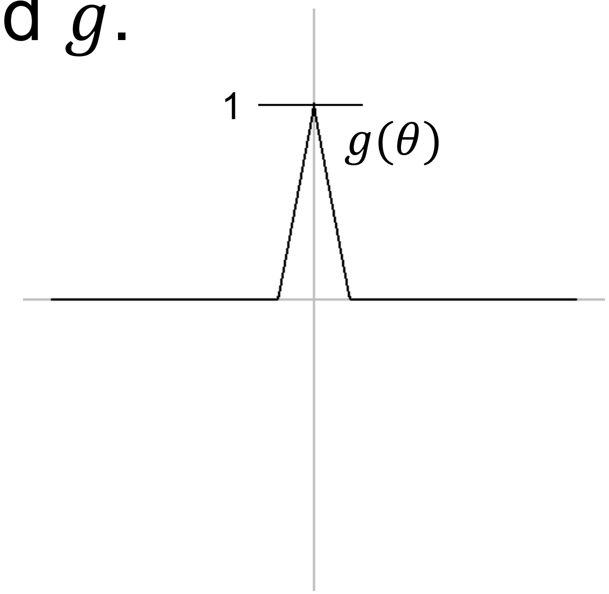
Band-Limited Spectrum

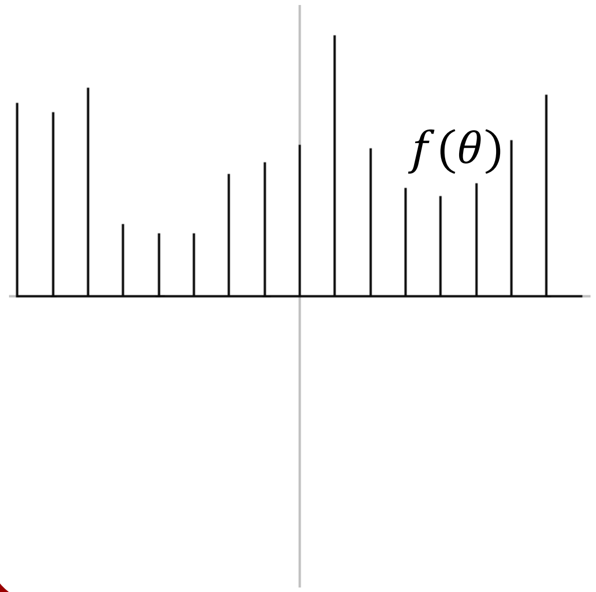
Fourier Theory

A fundamental fact from Fourier theory is that multiplication of power spectra in the frequency domain is convolution in the spatial domain.

Convolution

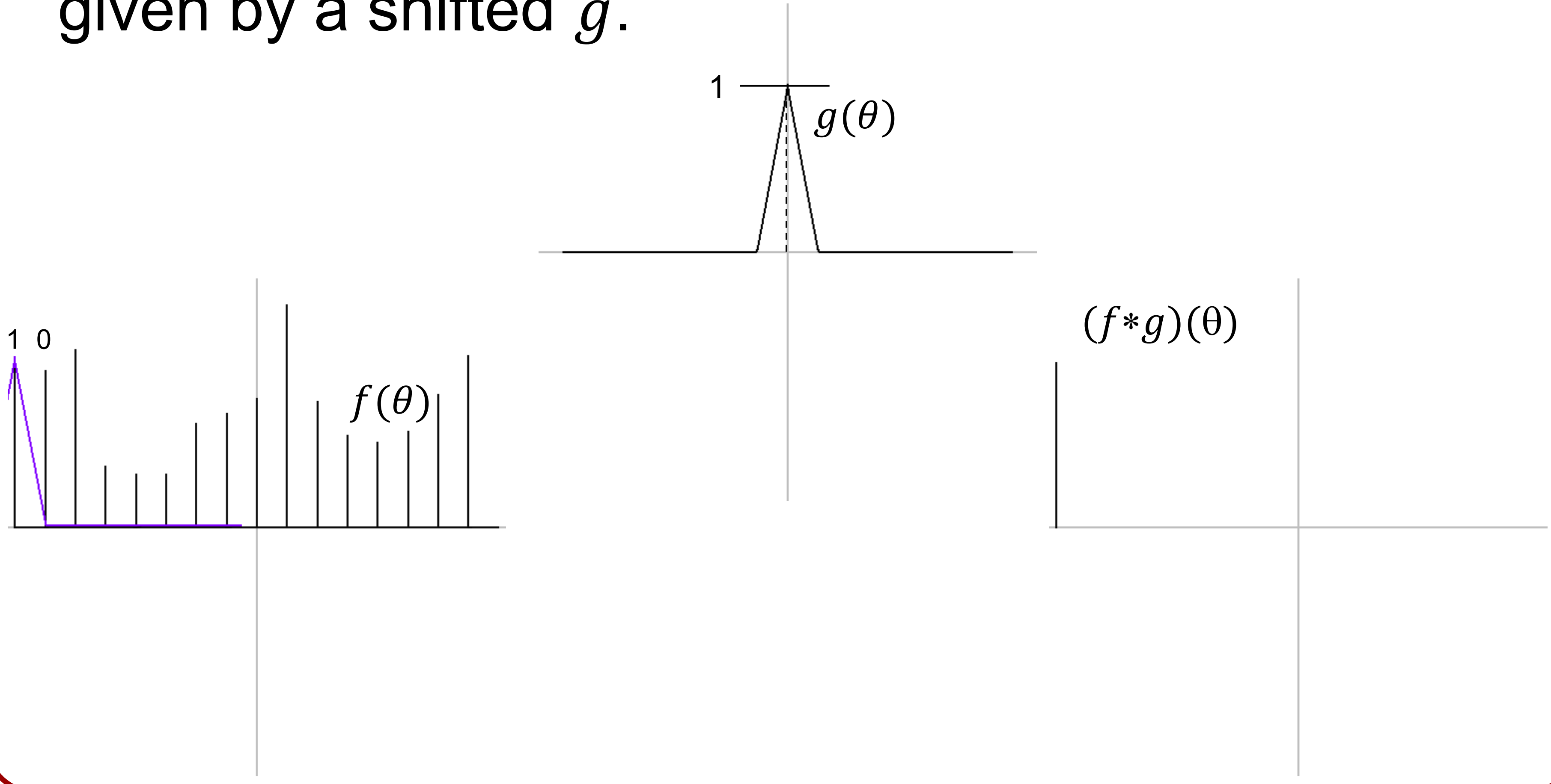
The <u>convolution</u> of functions f and g, denoted $f * g$, is obtained by sampling the values of f with weights given by a shifted g.





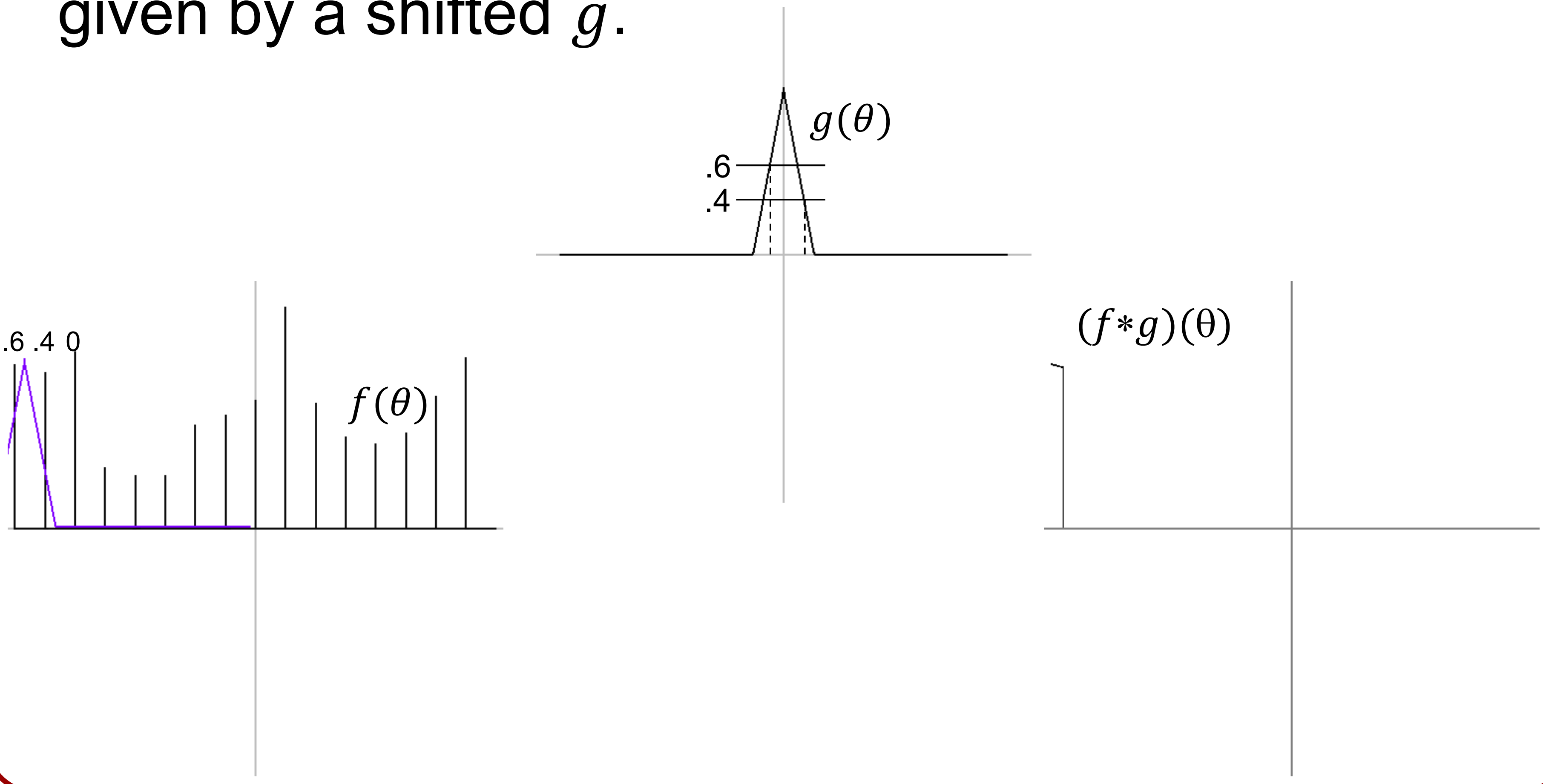
Convolution

The <u>convolution</u> of functions f and g, denoted $f * g$, is obtained by sampling the values of f with weights given by a shifted g.



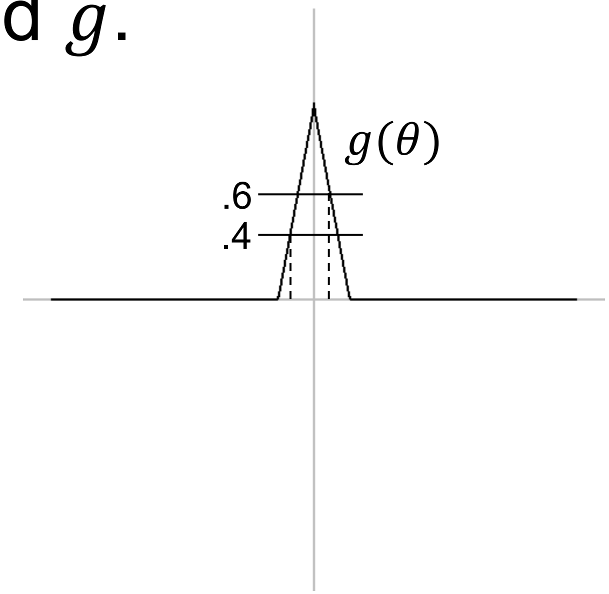
Convolution

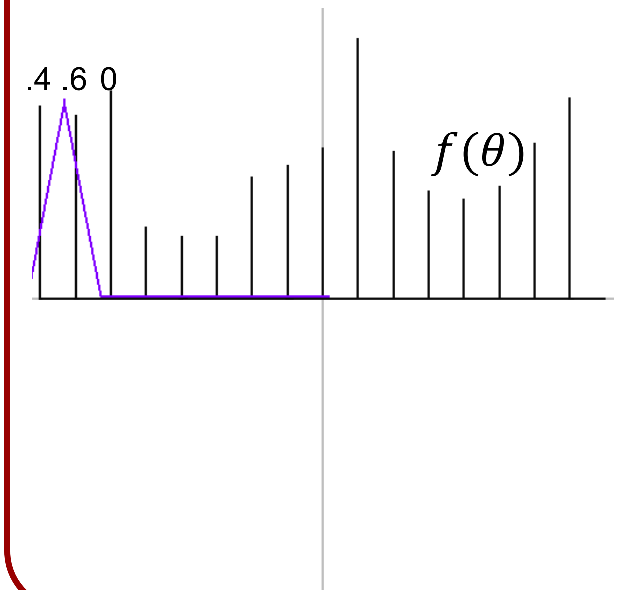
The <u>convolution</u> of functions f and g, denoted $f * g$, is obtained by sampling the values of f with weights given by a shifted g.

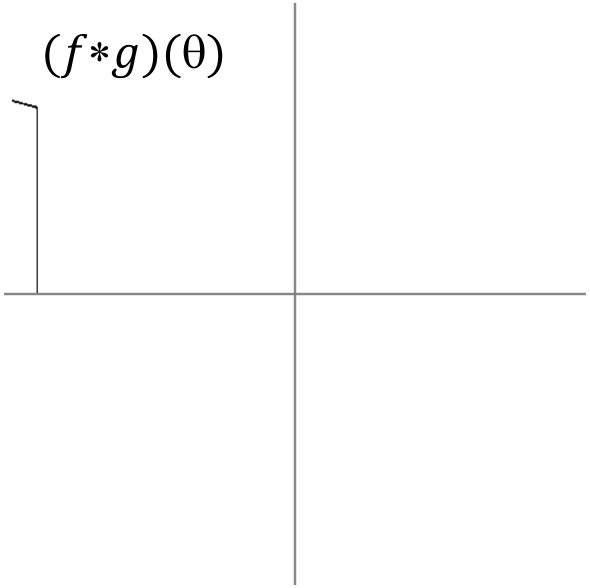


Convolution

The <u>convolution</u> of functions f and g, denoted $f * g$, is obtained by sampling the values of f with weights given by a shifted g.

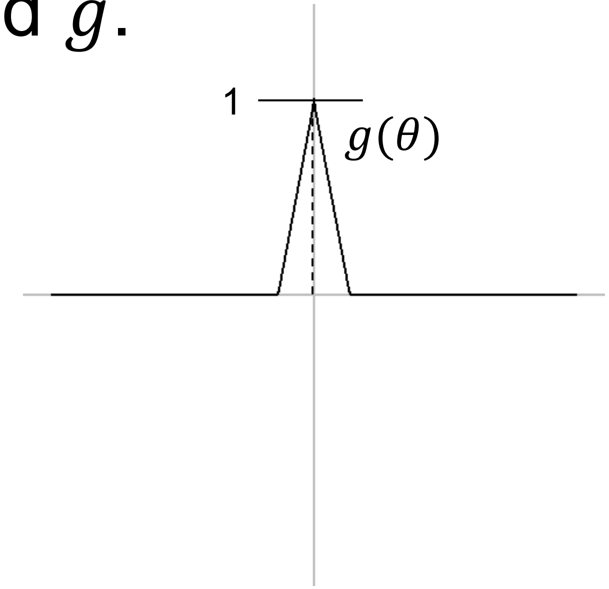


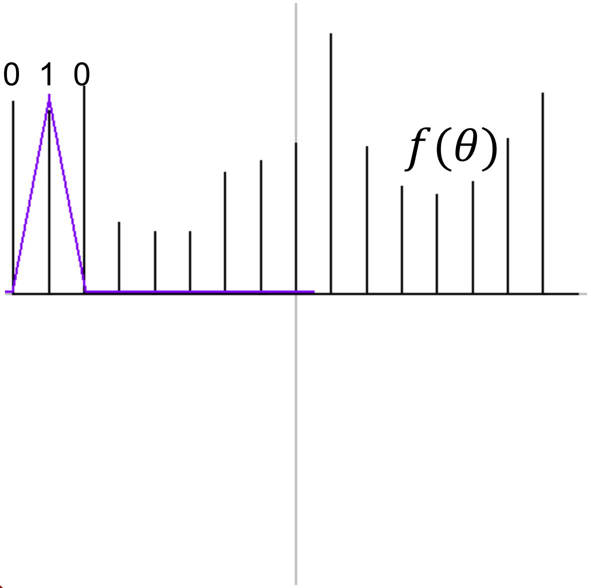


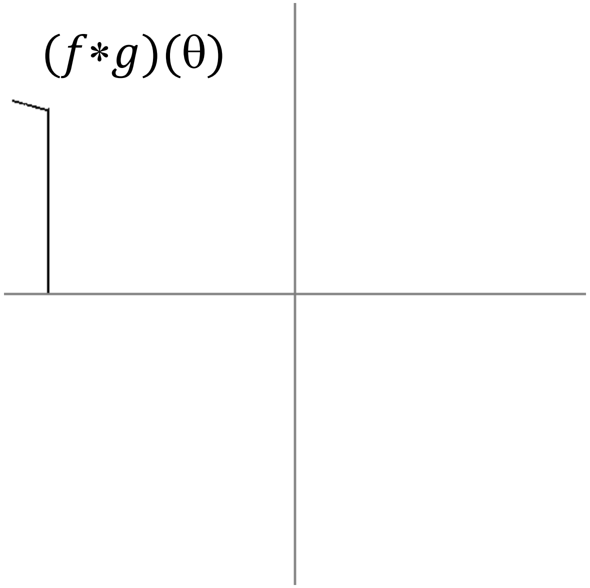


Convolution

The <u>convolution</u> of functions f and g, denoted $f * g$, is obtained by sampling the values of f with weights given by a shifted g.

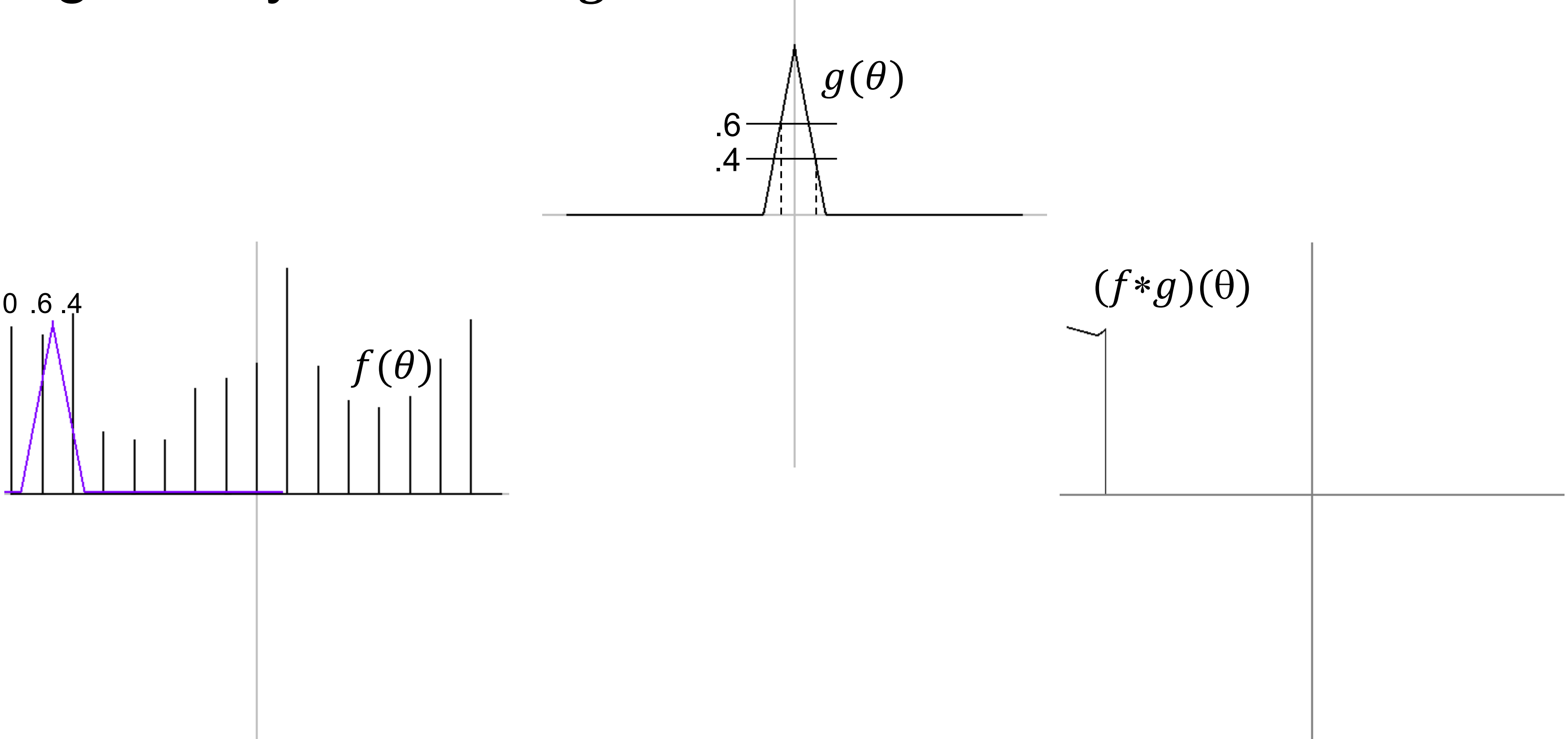






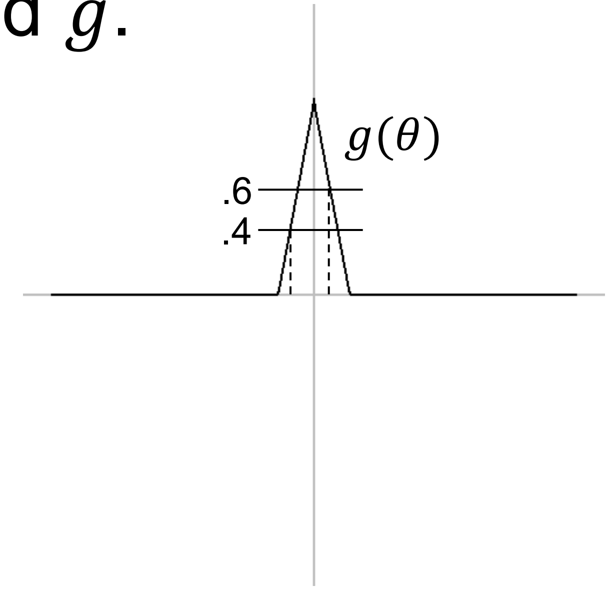
Convolution

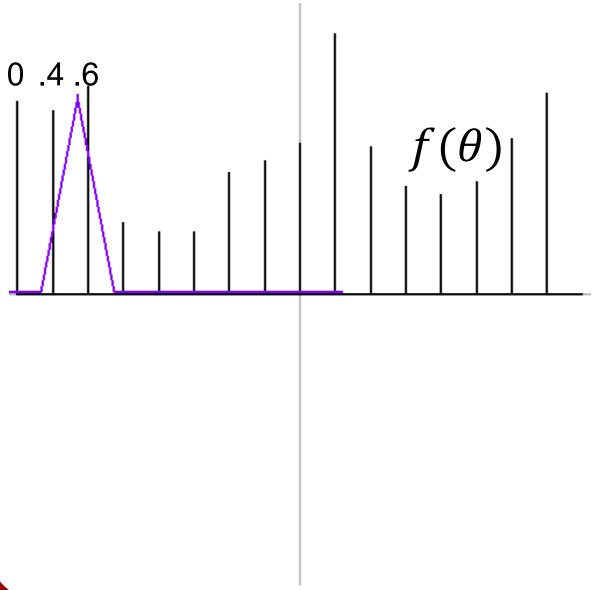
The convolution of functions f and g, denoted $f * g$, is obtained by sampling the values of f with weights given by a shifted g.

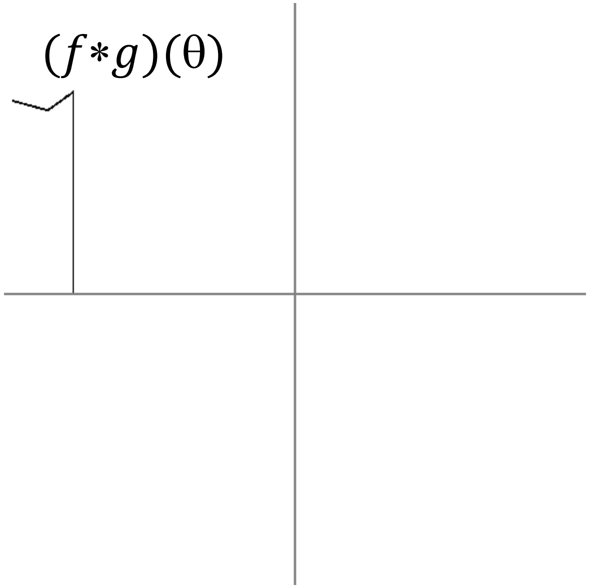


Convolution

The <u>convolution</u> of functions f and g, denoted $f * g$, is obtained by sampling the values of f with weights given by a shifted g.

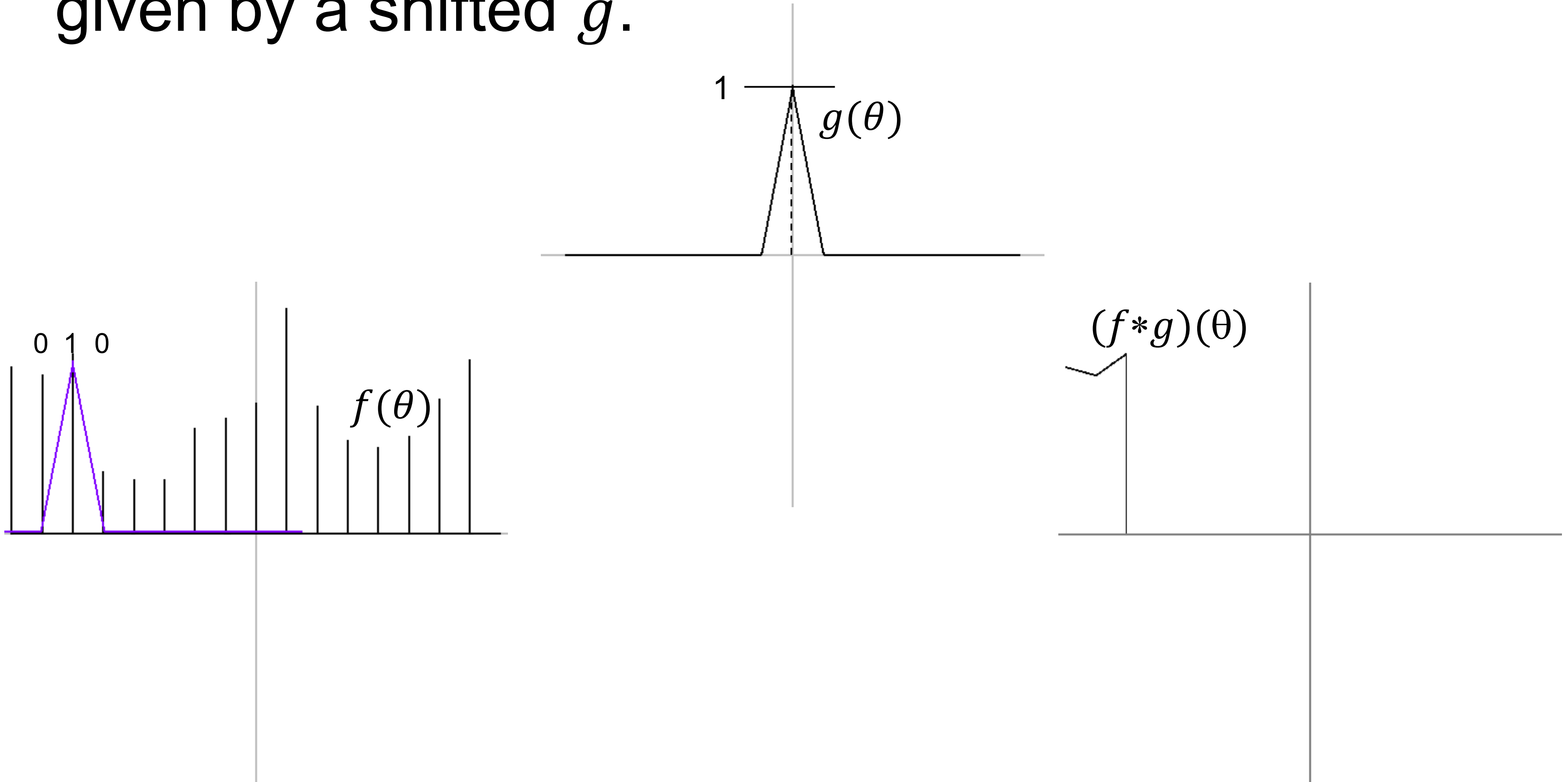






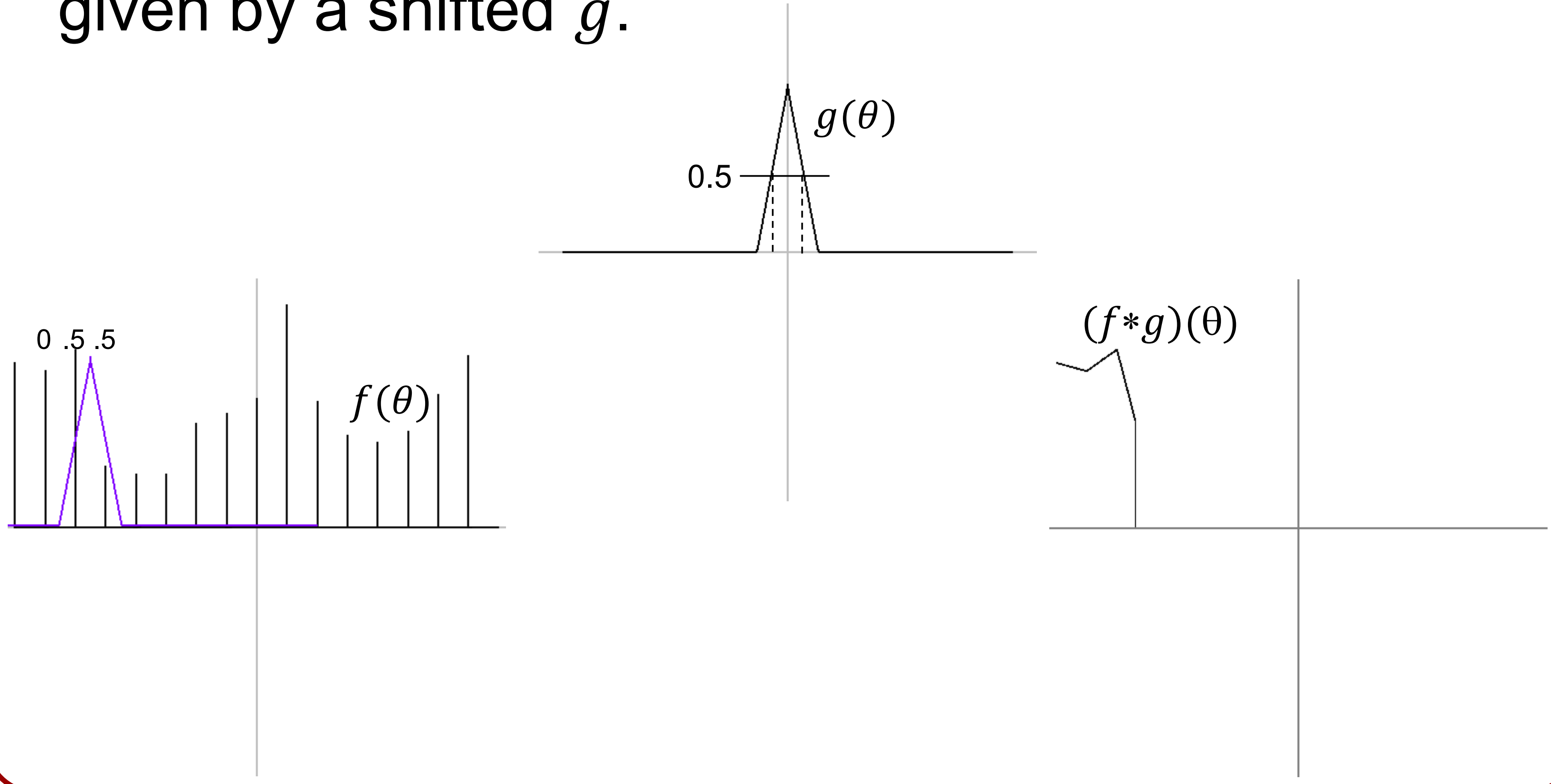
Convolution

The convolution of functions f and g, denoted $f * g$, is obtained by sampling the values of f with weights given by a shifted g.



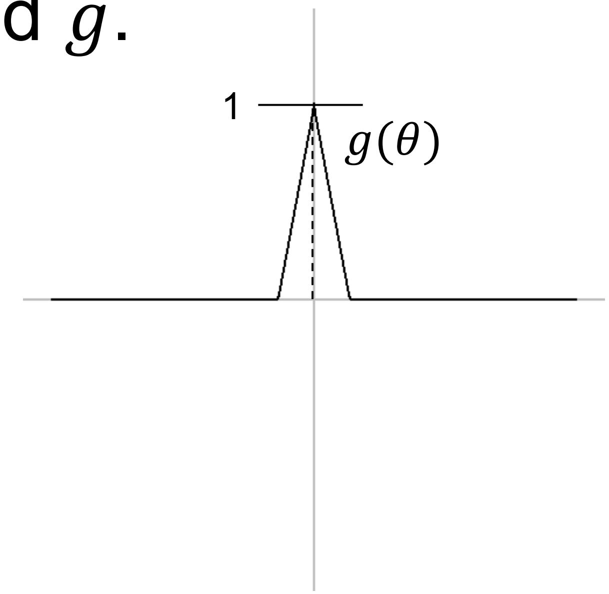
Convolution

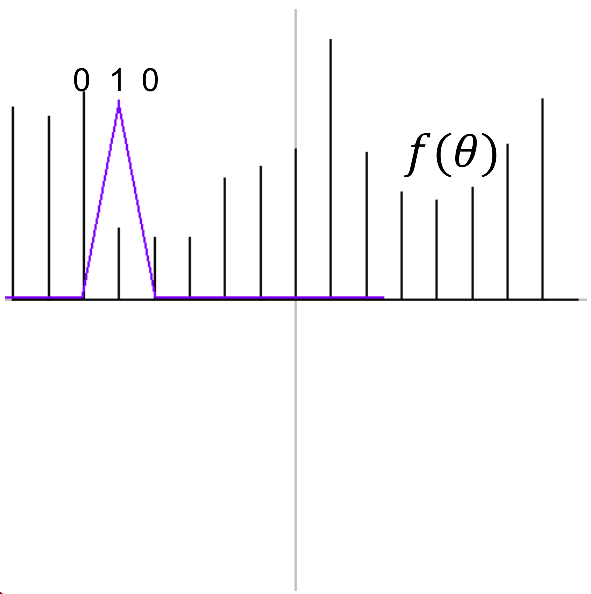
The <u>convolution</u> of functions f and g, denoted $f * g$, is obtained by sampling the values of f with weights given by a shifted g.

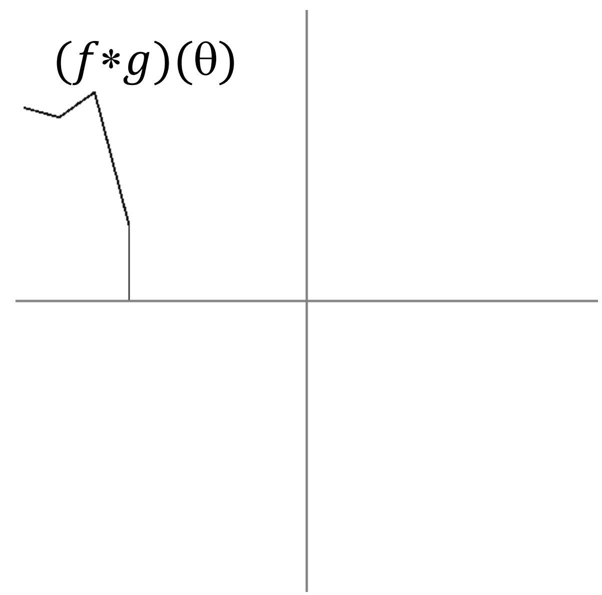


Convolution

The convolution of functions f and g, denoted $f * g$, is obtained by sampling the values of f with weights given by a shifted g.

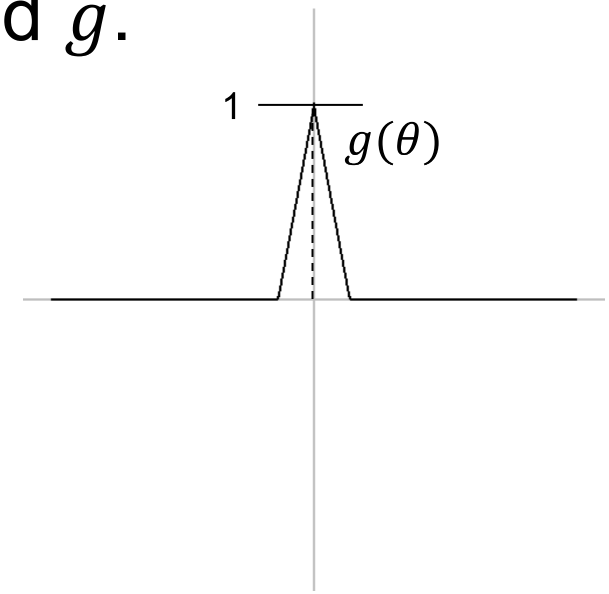


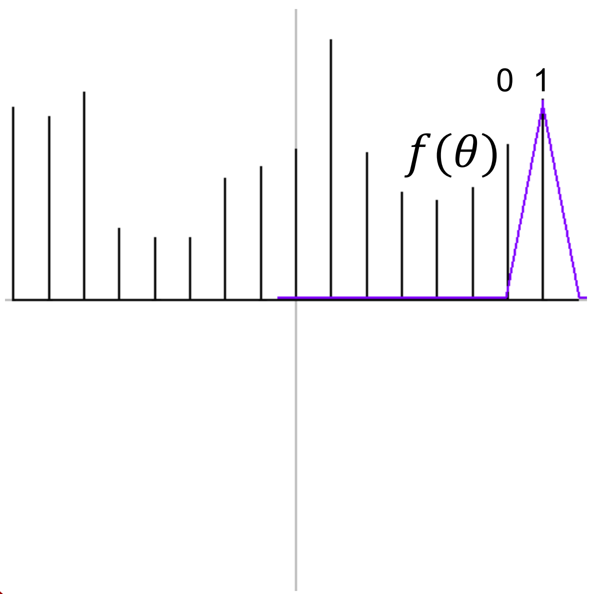


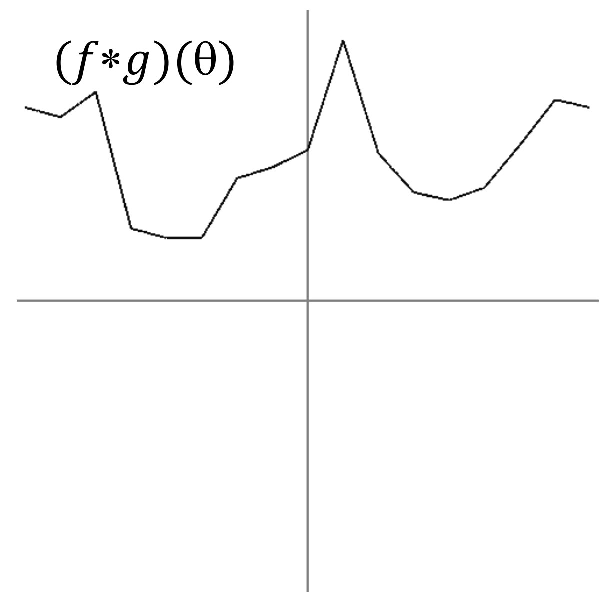


Convolution

The convolution of functions f and g, denoted $f * g$, is obtained by sampling the values of f with weights given by a shifted g.

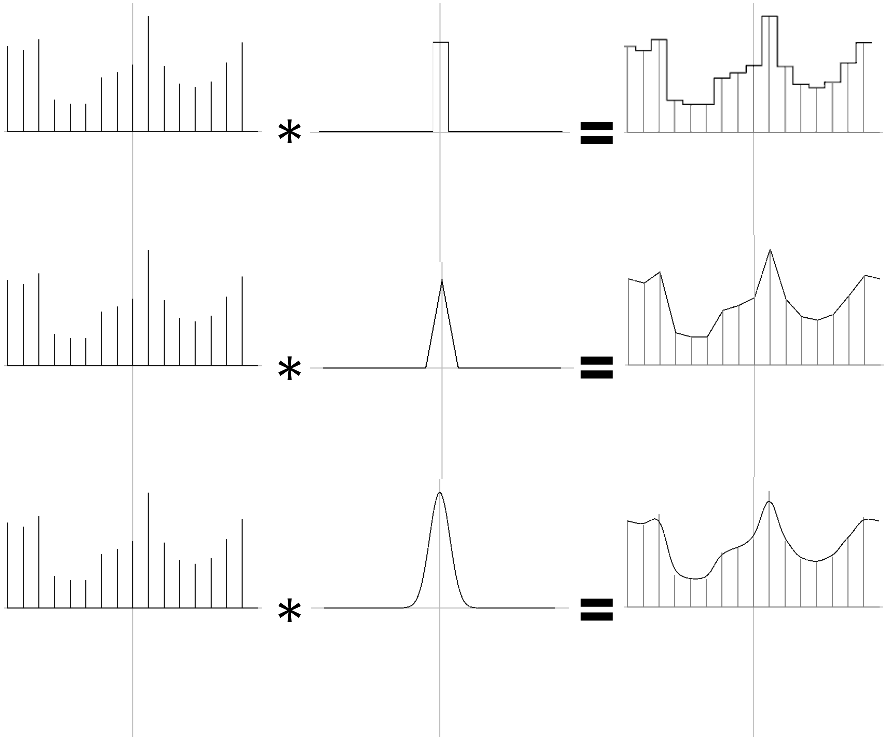






Convolution

- To convolve functions f and g, we resample the function f using the weights given by g.
- Nearest, (bi)linear, and Gaussian interpolation are convolutions with different filters.

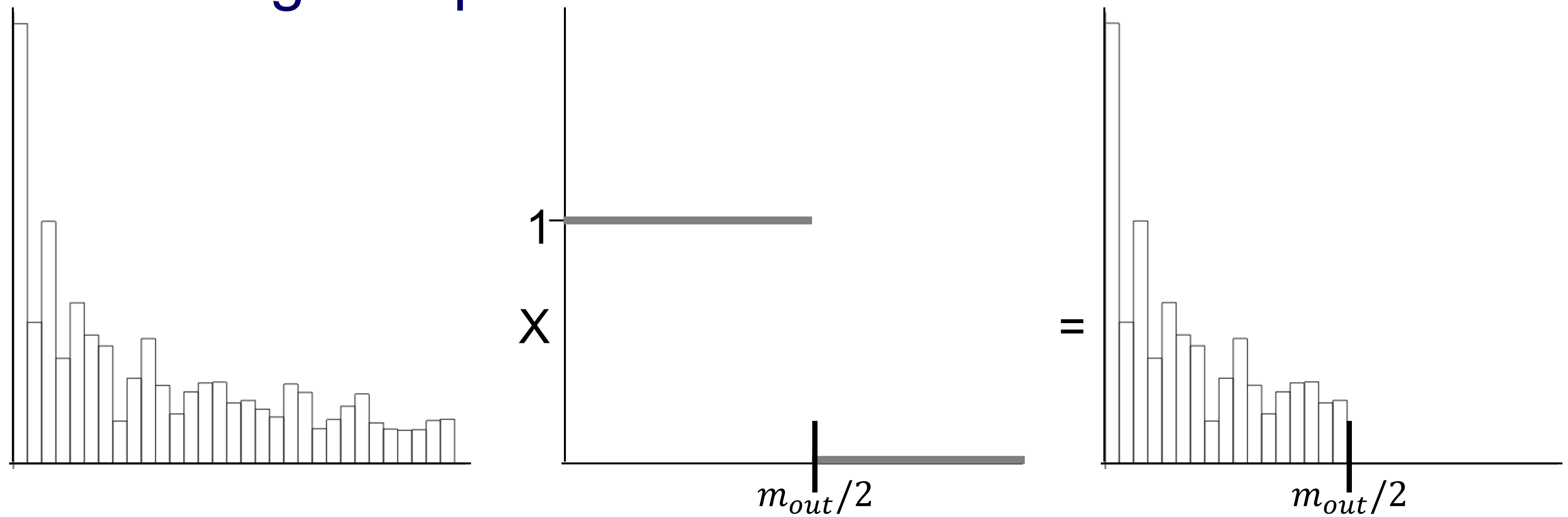


Convolution

Since convolution in the spatial domain is equal to multiplication in the frequency domain…

⇒ To avoid aliasing, we should convolve with a filter whose power spectrum has value:

- 1 at low frequencies
- 0 at high frequencies

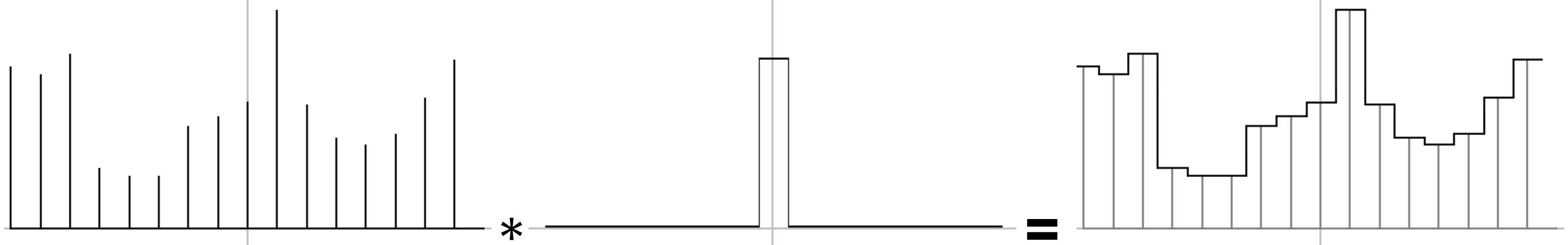


Initial Spectrum

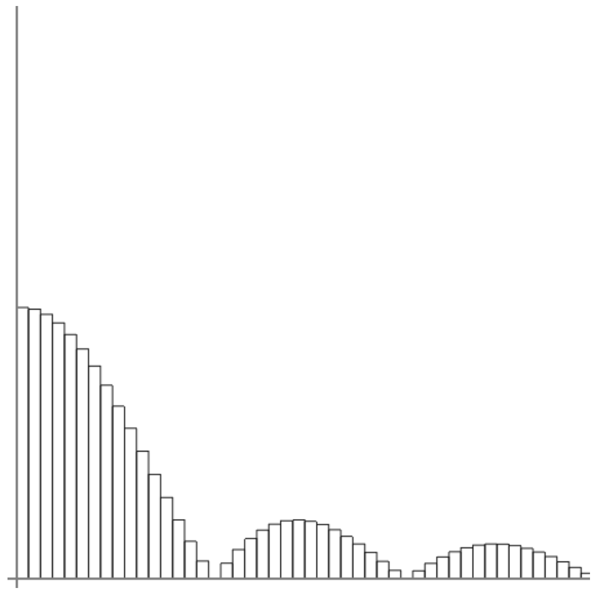
Frequency Filter

Band-Limited Spectrum

Nearest Point Convolution



Discrete Samples * Reconstruction Filter = Reconstructed Function

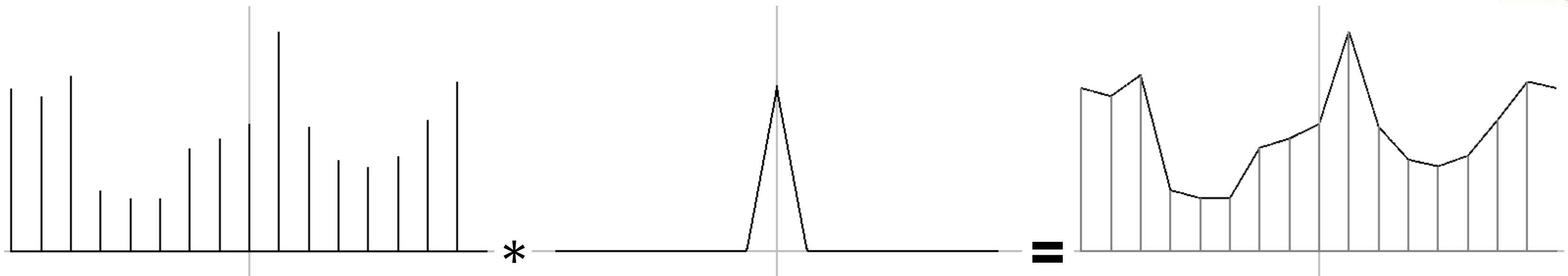


Filter’s Power Spectrum

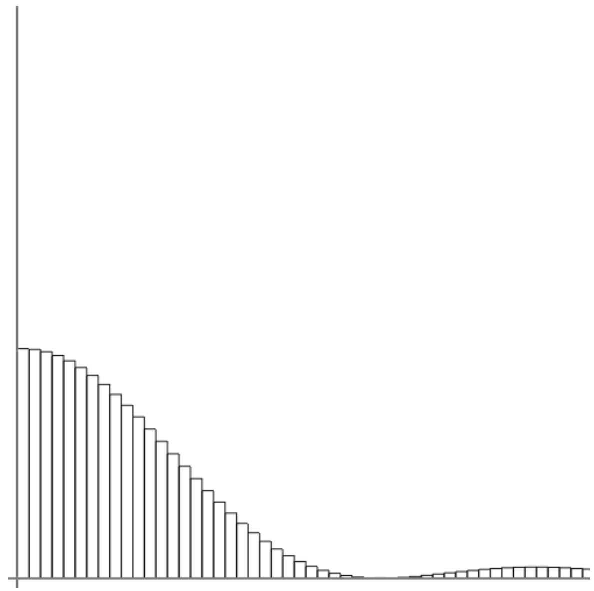
Note:
The spectrum does not really fall off at high frequencies.

Also:
The nearest-point filter does not provide a way for controlling the cut-off frequency.

(Bi)Linear Convolution



Discrete Samples * Reconstruction Filter = Reconstructed Function

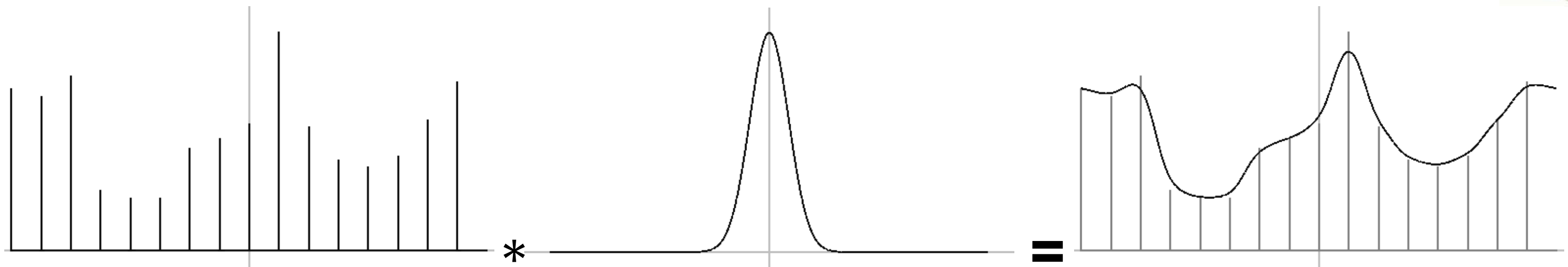


Filter's Power Spectrum

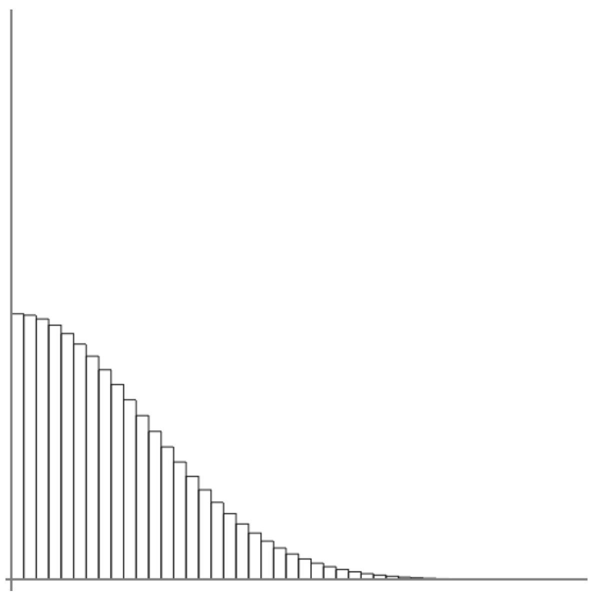
Note:
The spectrum does a better job of falling off at high frequencies, but still doesn't go to zero.

Also:
The (bi)linear filter does not provide a way for controlling the cut-off frequency.

Gaussian Convolution



Discrete Samples | Reconstruction Filter | Reconstructed Function



Filter's Power Spectrum

<u>Note</u>:
The spectrum quickly decays to zero at high frequencies, (falling off like a Gaussian).

<u>Also</u>:
The variance of the Gaussian filter provides a way for controlling the cut-off frequency.

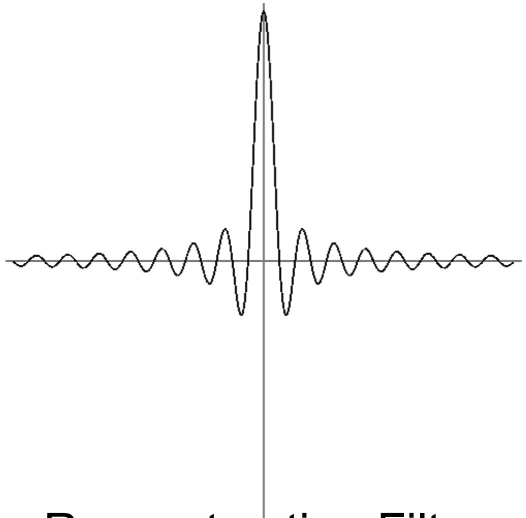
Convolution

- The ideal filter for avoiding aliasing should have a power spectrum with values:
  - 1 at low frequencies
  - 0 at high frequencies
- The **sinc** function has such a power spectrum and is referred to as the *ideal reconstruction filter*:

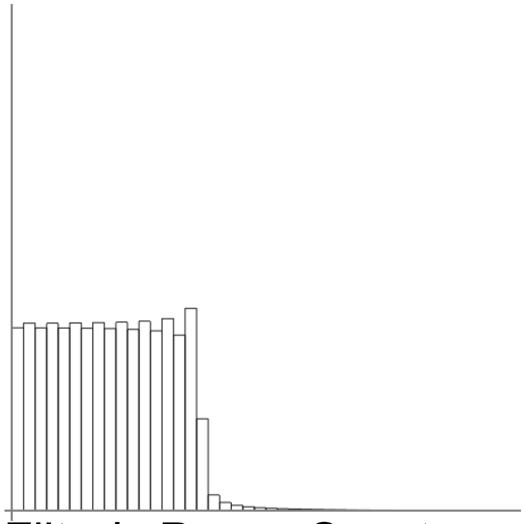
$$\operatorname{sinc}(\theta) = \begin{cases} \dfrac{\sin(\theta)}{\theta} & \text{if } \theta \neq 0 \\ 1 & \text{if } \theta = 0 \end{cases}$$

The Sinc Filter

- The ideal filter for avoiding aliasing should have a power spectrum with values:
    - 1 at low frequencies
    - 0 at high frequencies
- The **sinc** function has such a power spectrum and is referred to as the *ideal reconstruction filter*:



Reconstruction Filter

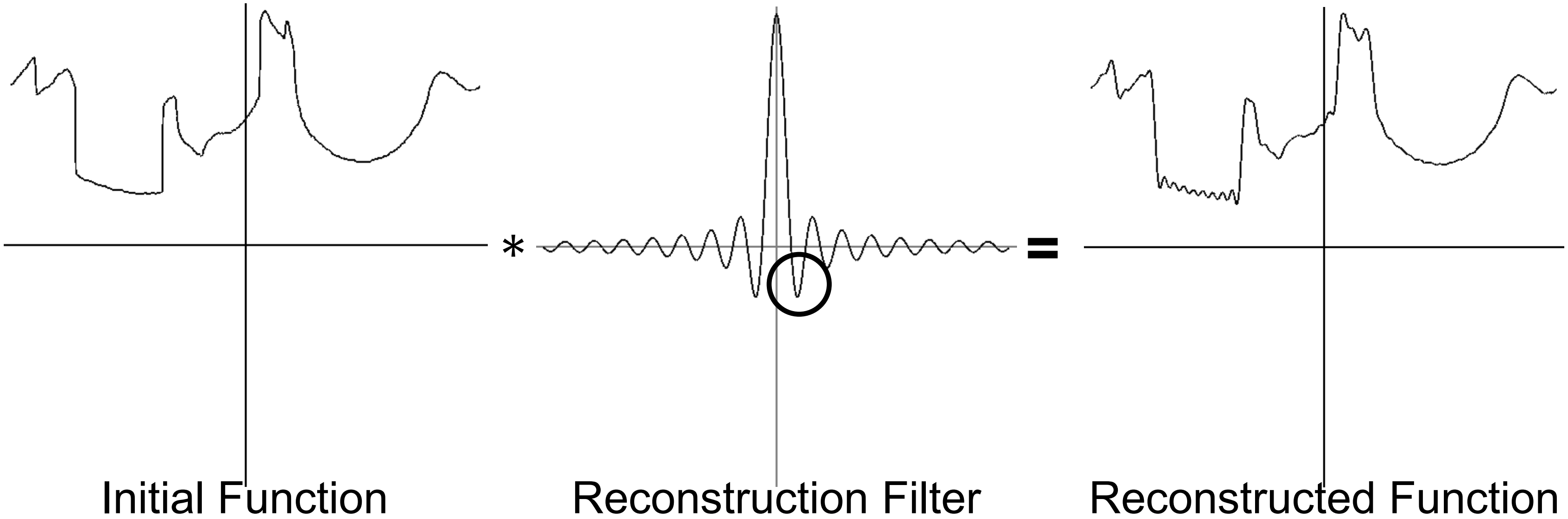


Filter's Power Spectrum

The Sinc Filter

Limitations:

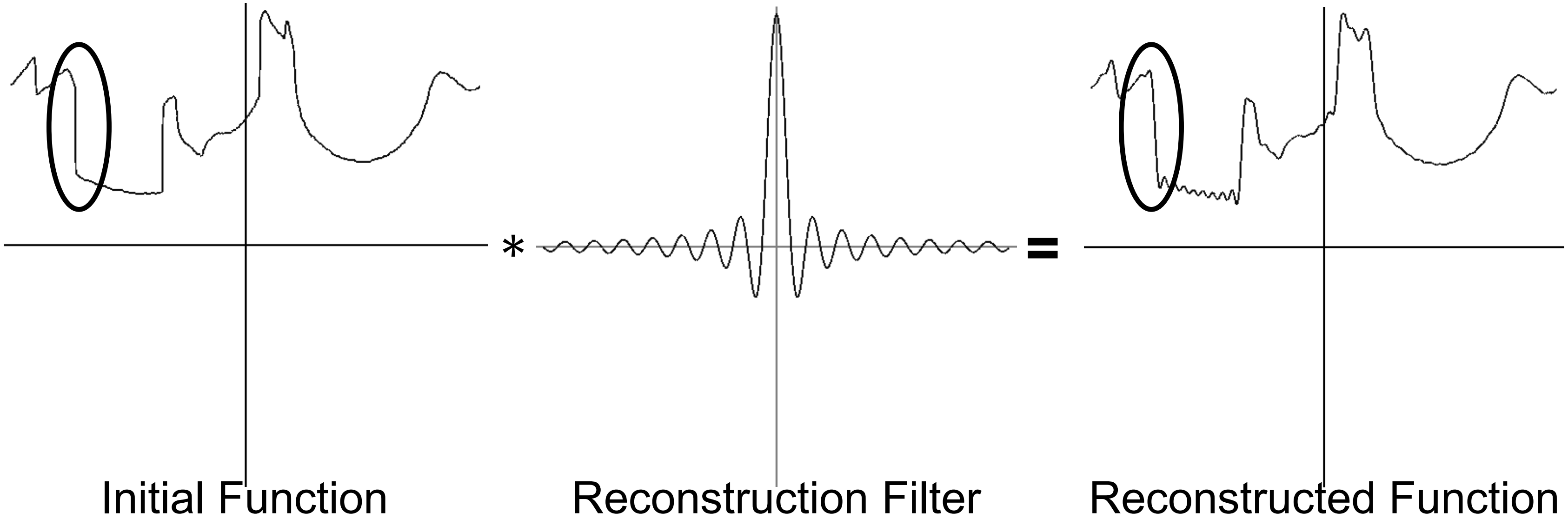
- Has negative values, giving rise to negative weights in the interpolation → can extrapolate values.



The Sinc Filter

Limitations:

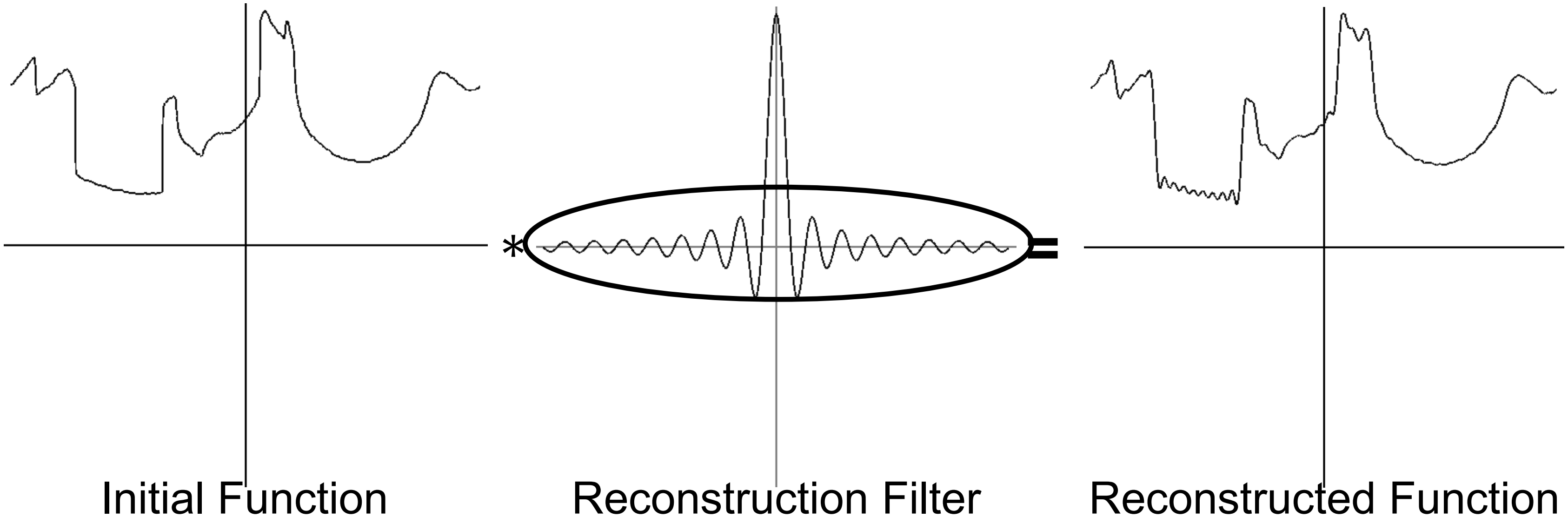
- Has negative values, giving rise to negative weights in the interpolation → can extrapolate values.
- Discontinuity in the frequency domain causes **ringing** near spatial discontinuities (Gibbs Phenomenon).



The Sinc Filter

Limitations:

- Has negative values, giving rise to negative weights in the interpolation → can extrapolate values.
- Discontinuity in the frequency domain causes **ringing** near spatial discontinuities (Gibbs Phenomenon).
- The filter has large support so evaluation is slow.



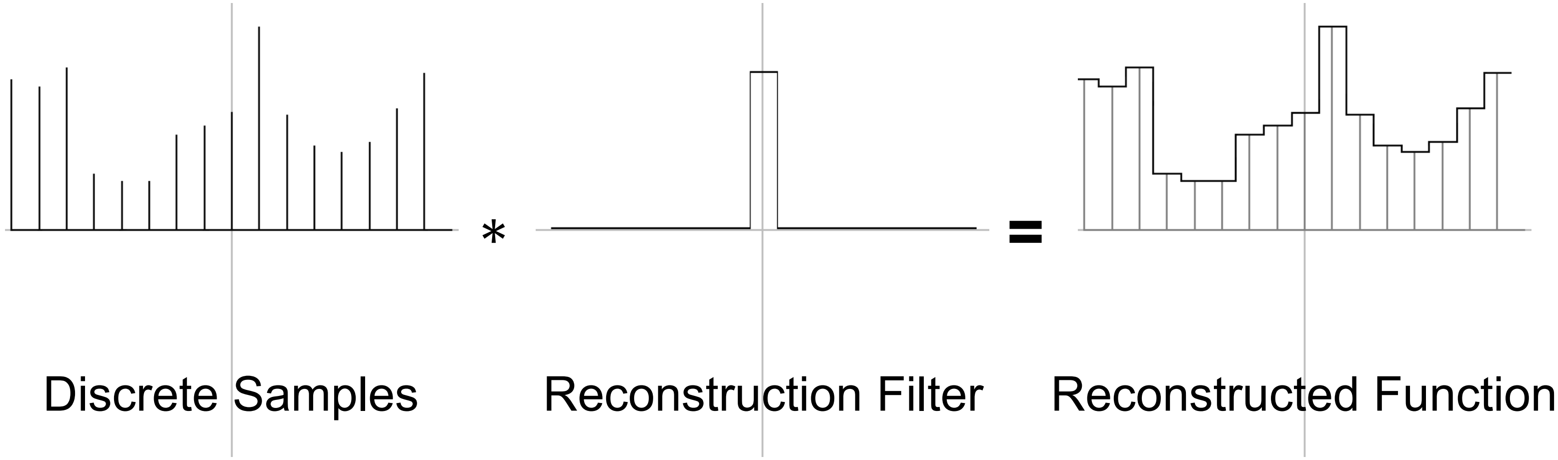
Summary

There are different ways to sample an image:

- Nearest Point Sampling
- Linear Sampling
- Gaussian Sampling
- Sinc Sampling

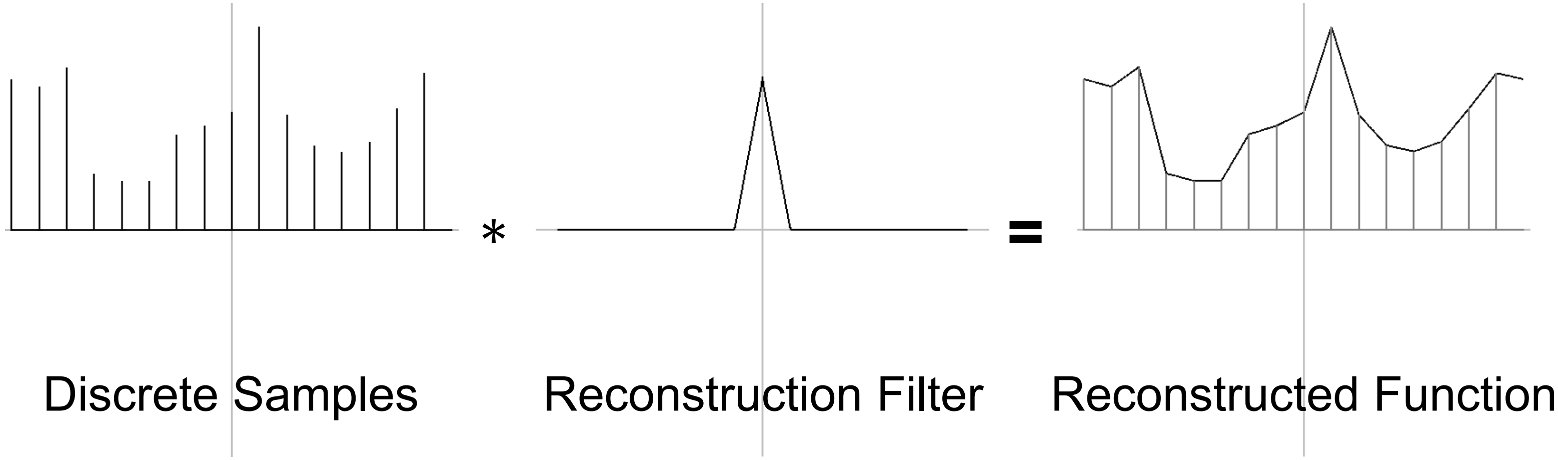
Summary – Nearest

- ✓ Can be implemented efficiently because the filter is non-zero in a very small region.
- ? Interpolates the samples.
- ✗ Is discontinuous.
- ✗ Does not address aliasing, giving bad results when a high-frequency signal is under-sampled.
- ✗ Particularly bad when the output resolution is much lower than the input resolution.



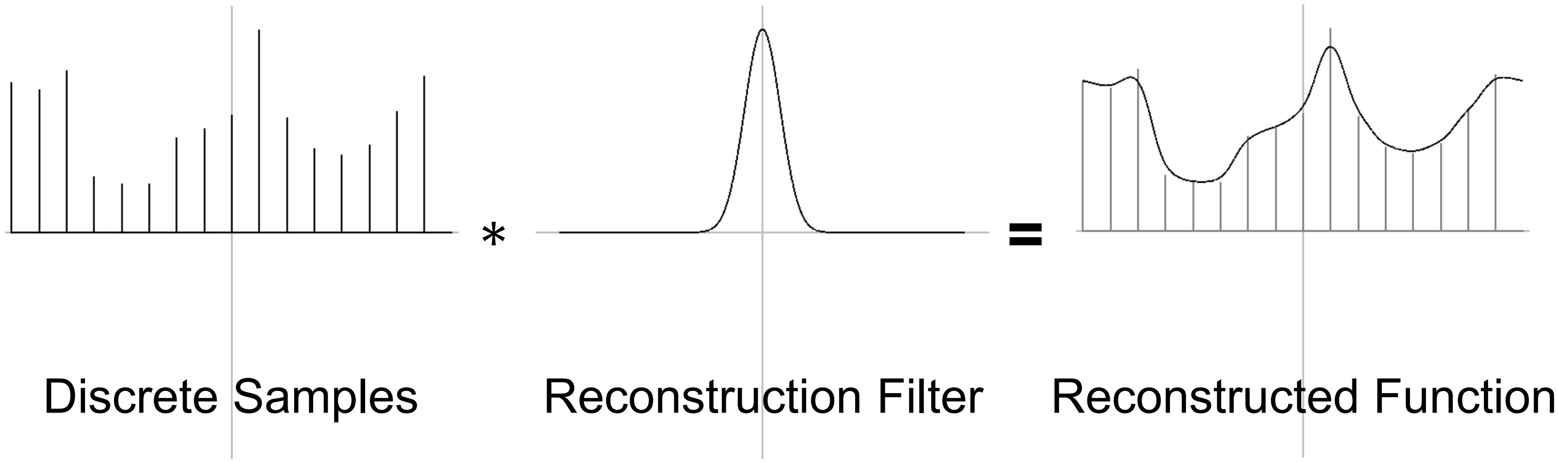
Summary – (Bi)linear

- ✓ Can be implemented efficiently because the filter is non-zero in a very small region.
- ? Interpolates the samples.
- ✗ Is not smooth.
- ✗ Partially addresses aliasing, but still gives bad results when a high-frequency signal is under-sampled.
- ✗ Still bad when the output resolution is much lower than the input resolution.



Summary – Gaussian

- ✗ Is slow to implement because the filter is non-zero in a large region.
- ? Does not interpolate the samples.
- ✓ Is smooth.
- ✓ Addresses aliasing by killing off high frequencies.
- ✓ Works well when the output resolution is much lower than the input resolution (by adapting the variance).



Summary – Sinc

- ✗ Is really slow to implement because the filter is non-zero, and large, in a large region.
- ? Interpolates the samples.
- ✗ Assigns negative weights.
- ✗ Ringing at discontinuities.
- ✓ Addresses aliasing by killing off high frequencies.
- ✓ Works well when the output resolution is much lower than the input resolution (by adapting the cut-off frequency).

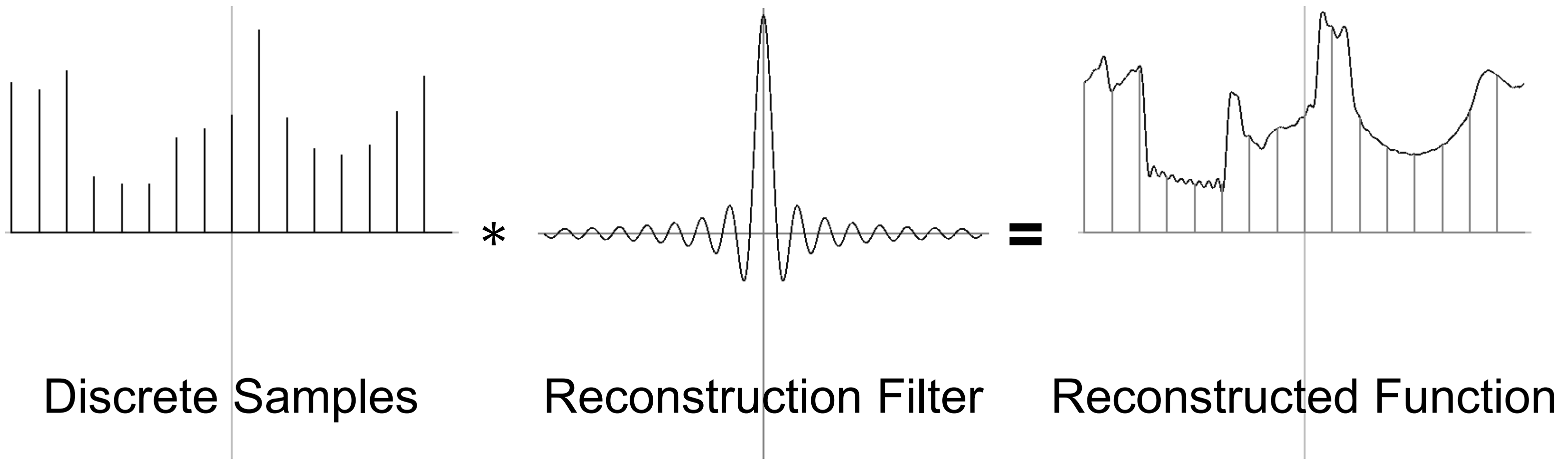


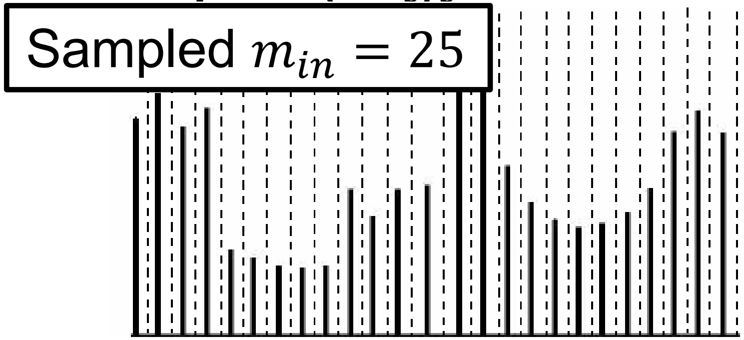
Image Sampling (Conceptually)

Given a source signal sampled at m_{in} positions, to get a destination image sampled at m_{out} positions:

1. Reconstruct:
   a) Generate a function with bandwidth $m_{in}/2$.
   b) Further filter the function to have frequency no larger than $m_{out}/2$.
2. Sample:
   Evaluate the filtered function at the m_{out} positions.

Image Sampling (Conceptually)

Example ($m_{in} = 25 \to m_{out} = 25/10$):



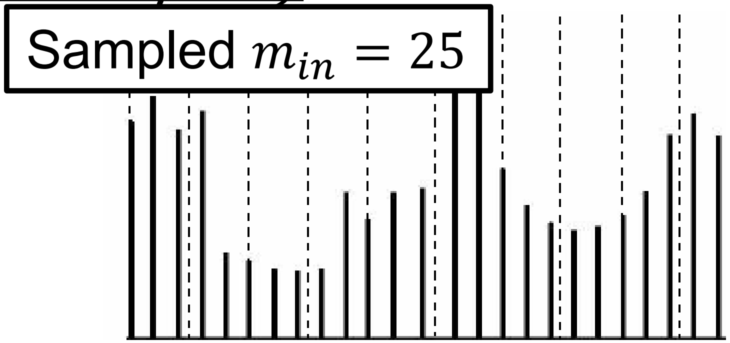
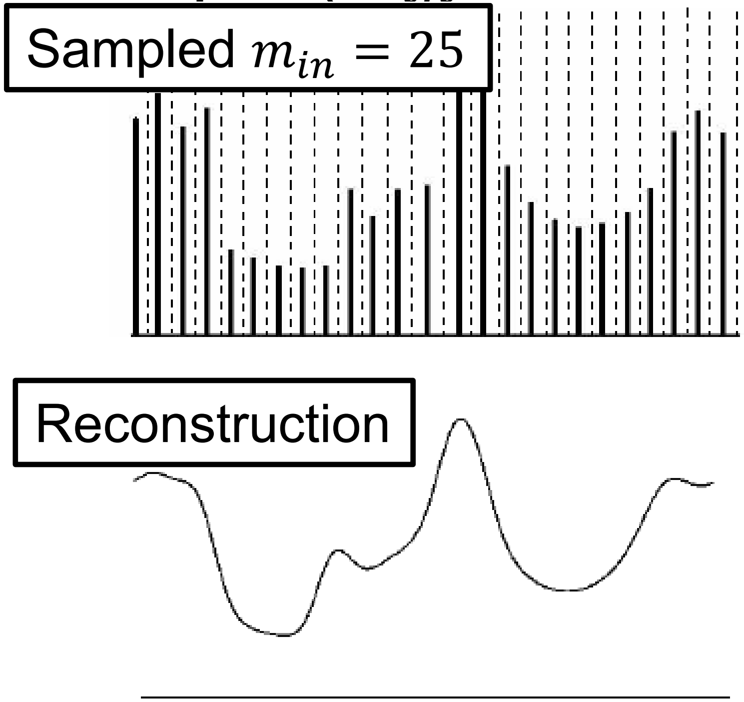


Image Sampling (Conceptually)

Example ($m_{in} = 25 \rightarrow m_{out} = 25/10$):



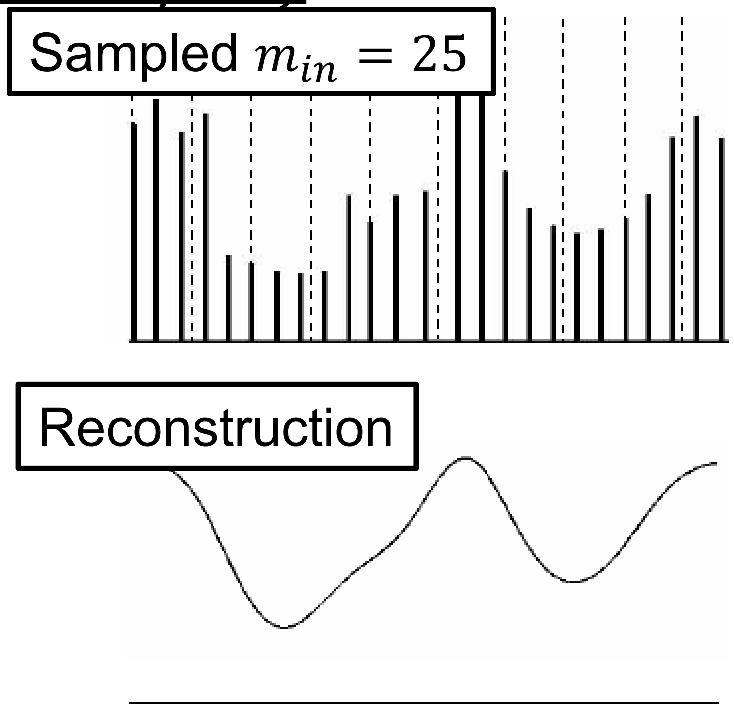


Image Sampling (Conceptually)

Example ($m_{in} = 25 \rightarrow m_{out} = 25/10$):

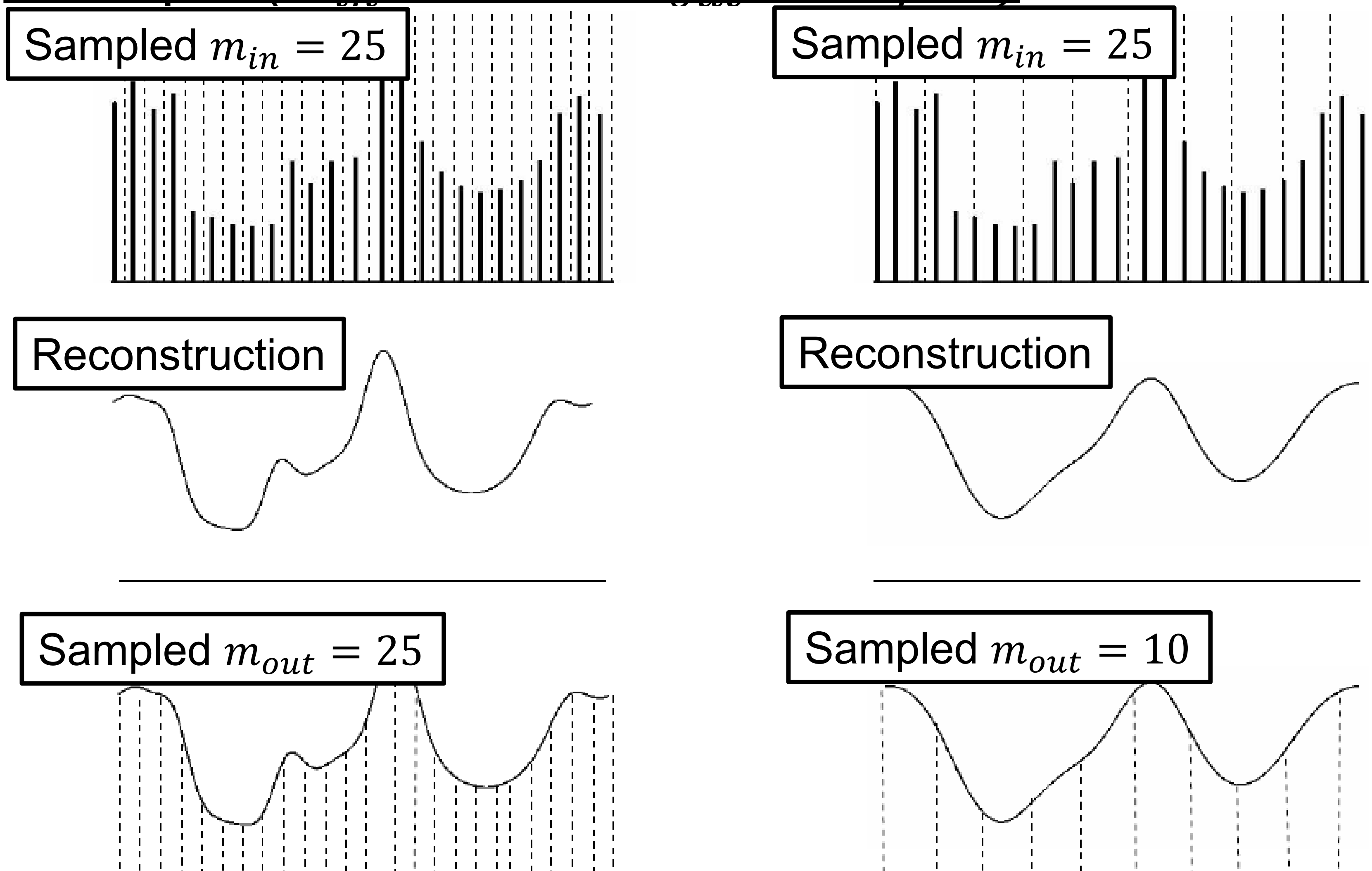


Image Sampling (in Practice)

Given a source signal sampled at m_{in} positions, to get a destination image sampled at m_{out} positions:

- Resample the source image using a (Gaussian) filter whose width is determined by the number of input and output samples.
- This simultaneously:
  1. *Reconstructs* a band-limited function from the input samples
  2. *Samples* the band-limited function at the output positions

Gaussian Sampling

Recall:

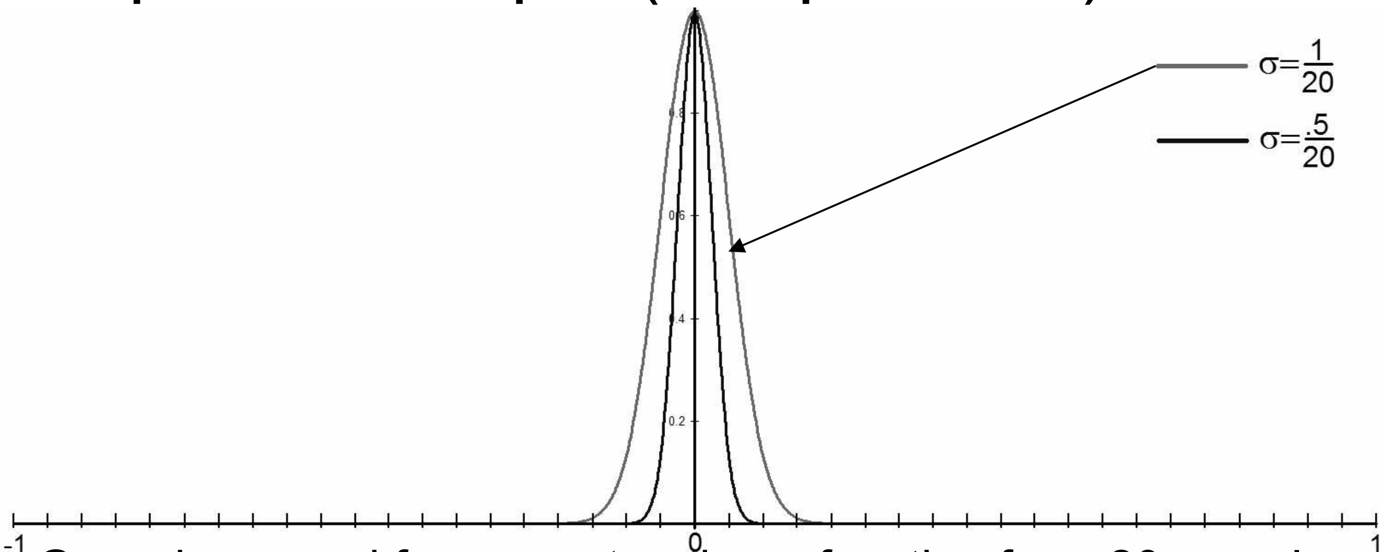
To avoid aliasing, we kill off the high-frequency components by convolving with a Gaussian because its power spectrum is:

- (approximately) one at low frequencies
- (approximately) zero at high frequencies

Gaussian Sampling (Rule of Thumb)

Q: What standard deviation should we use to sample the input?

A: The standard deviation should be between 0.5 and 1.0 times the maximum sample spacing in the input **and** output (in input units).

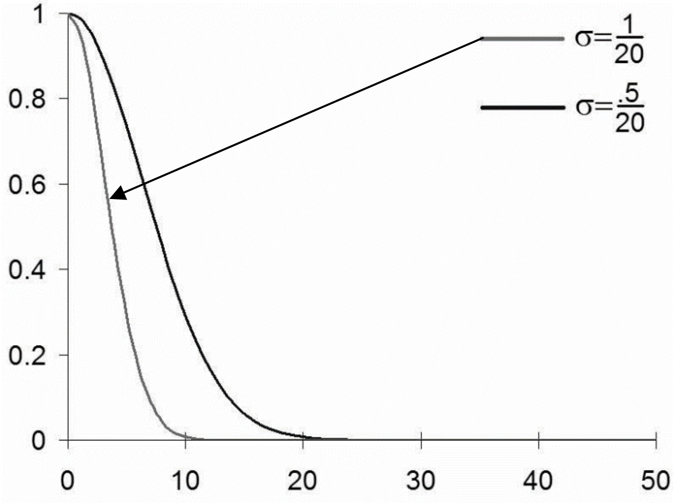


Gaussians used for reconstructing a function from 20 samples
(samples are 1/20 units apart)

Gaussian Sampling (Rule of Thumb)

Q: What standard deviation should we use to sample the input?

A: The standard deviation should be between 0.5 and 1.0 times the maximum sample spacing in the input **and** output (in input units).



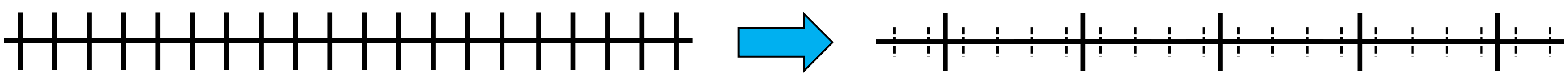
Power spectra of the Gaussians used for reconstructing and sampling a function with 20 samples

Gaussian Sampling

<u>Scaling Example</u>:

Q: If we have data represented by $m_{in} = 20$ samples that we want to down-sample to $m_{out} = 5$ samples.
What standard deviation should we use?

A: Distance between input samples (in input units): 1
Distance between output samples (in input units): 4
⇒ The standard deviation of the Gaussian used to sample the input should be between 2.0 and 4.0 (input units).



Gaussian Sampling

Scaling Example:

Q: If we have data represented by $m_{in} = 5$ samples that we want to up-sample to $m_{out} = 20$ samples.
What standard deviation should we use?

A: Distance between input samples (in input units): 1
Distance between output samples (in input units): 0.25
⇒ The standard deviation of the Gaussian used to sample the input should be between 0.5 and 1.0 (input units).

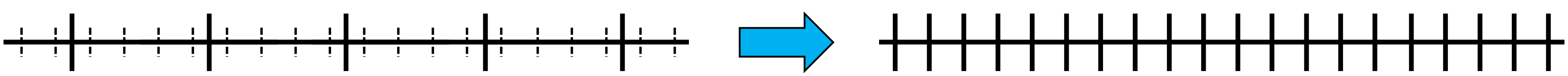


Image Processing

- Quantization
  - Uniform quantization
  - Ordered dither
  - Random dither
  - Floyd-Steinberg dither
- Pixel operations
  - Compute luminance
  - Change contrast
  - Change saturation
- Filtering
  - Blurring
  - Edge-detection
- Morphing
  - Blending
  - Warp
- Sampling
  - Aliasing
  - Ideal filter