CPSC 4040/6040 Computer Graphics Images

Joshua Levine
levinej@clemson.edu

Lecture 22
Frequency Domain

Nov. 12, 2015

Agenda

- Q05 Posted Tonight
- PA06/PA07 questions?

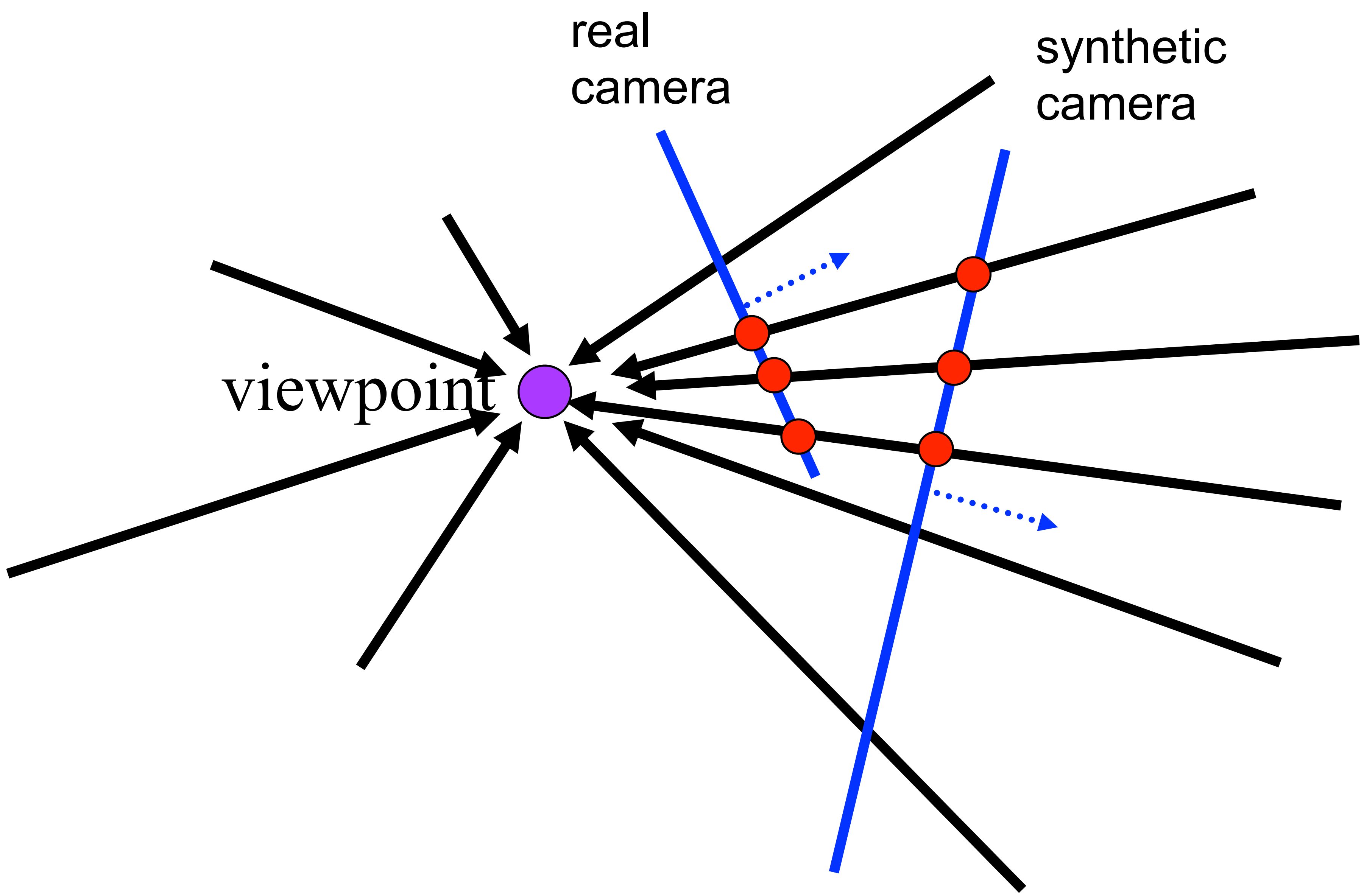
Why Panoramas?

- Are you getting the whole picture?
- Compact Camera FOV = 50 x 35°
- Human FOV = 200 x 135°
- Panoramic Mosaic = 360 x 180°

Slide from Brown & Lowe

Panoramas: Stitching Images Together

A Pencil of Rays Contains All Views



Can generate any synthetic camera view
as long as it has **the same center of projection**!

Image Stitching Pipeline

- The user gives us 4 correspondences
- We reproject one image to match the other one
- Creates a wider angle view

$$\begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} y \\ x \\ 1 \end{pmatrix} = \begin{pmatrix} y'w' \\ x'w' \\ w' \end{pmatrix}$$

$$w' = gy + hx + i$$

- **For a pair of points (x, y)->(x',y') we have**

$$ay + bx + c = y'(gy + hx + i)$$

$$dy + ex + f = x'(gy + hx + i)$$

- **Unknowns: a, b, c, d, e, f, g, h, i**
  - Linear!

Forming the matrix

$$ay + bx + c = y'(gy + hx + i)$$

$$dy + ex + f = x'(gy + hx + i)$$

$$\begin{array}{cccccccc} a & b & c & d & f & g & h & i \end{array}$$

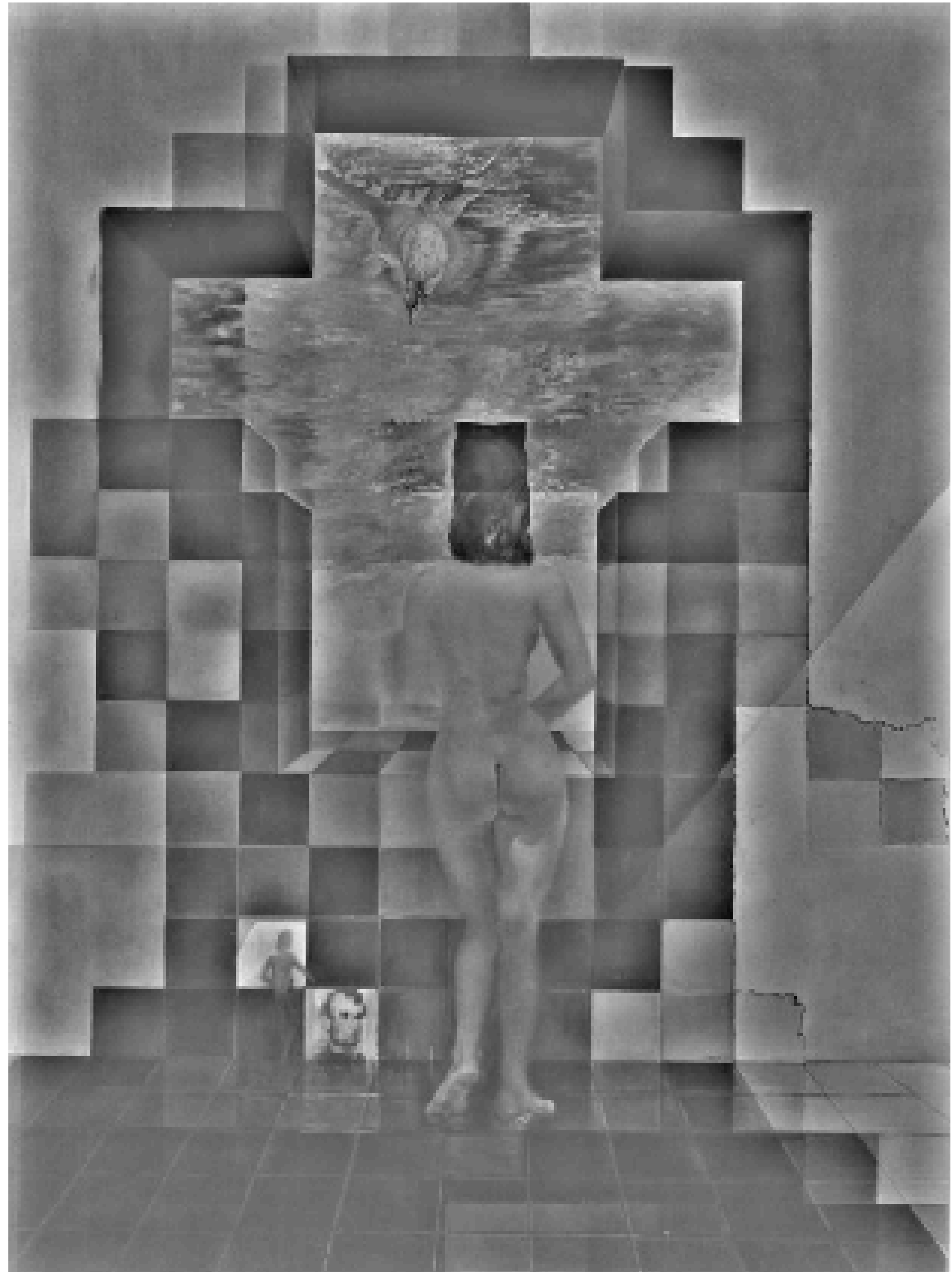
$$\begin{pmatrix} & & & & \cdots\cdots & & & & \\ y & x & 1 & 0 & 0 & 0 & -yy' & -xy' & -y' \\ & & & & \cdots\cdots & & & & \\ & & & & \cdots\cdots & & & & \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ f \\ g \\ h \\ i \end{pmatrix} = \begin{matrix} . \\ . \\ . \\ 0 \\ . \\ . \\ . \end{matrix}$$

Thinking Frequency

Salvador Dali
“Gala Contemplating the Mediterranean Sea, which at 20 meters becomes the portrait of Abraham Lincoln”, 1976

- What filter?

- What filter?



Domains

- Images can be represented in different domains
  - **Spatial domain** – the strength of light at points in space
  - **Frequency domain** – the strength of patterns within an image
- Frequency domain is useful for
  - Image analysis
  - Image compression
  - Efficient processing

Jean Baptiste Joseph Fourier (1768-1830)

Fourier's Idea (1807):

"**Any** univariate function can be rewritten as a weighted sum of sines and cosines of different **frequencies**."

Jean Baptiste Joseph Fourier (1768-1830)

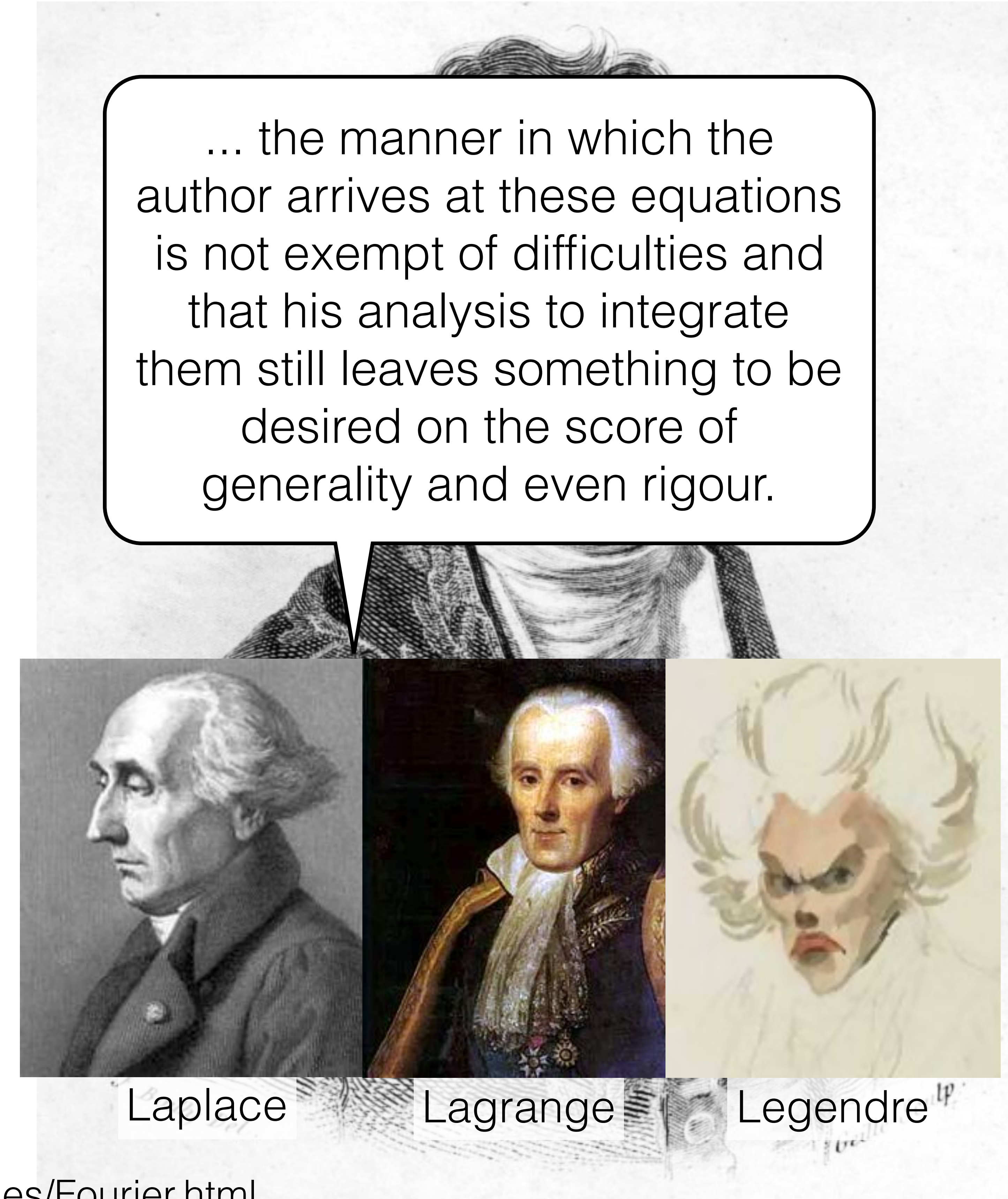
- Challenged by a number of mathematicians:
  - Including: Laplace, Lagrange, Legendre
- Fourier's idea was not even translated until 1878
- BUT, it's mostly true
  - It's called the Fourier Series
  - There are some subtle restrictions

Laplace Lagrange Legendre

http://www-history.mcs.st-and.ac.uk/Biographies/Fourier.html

Jean Baptiste Joseph Fourier (1768-1830)

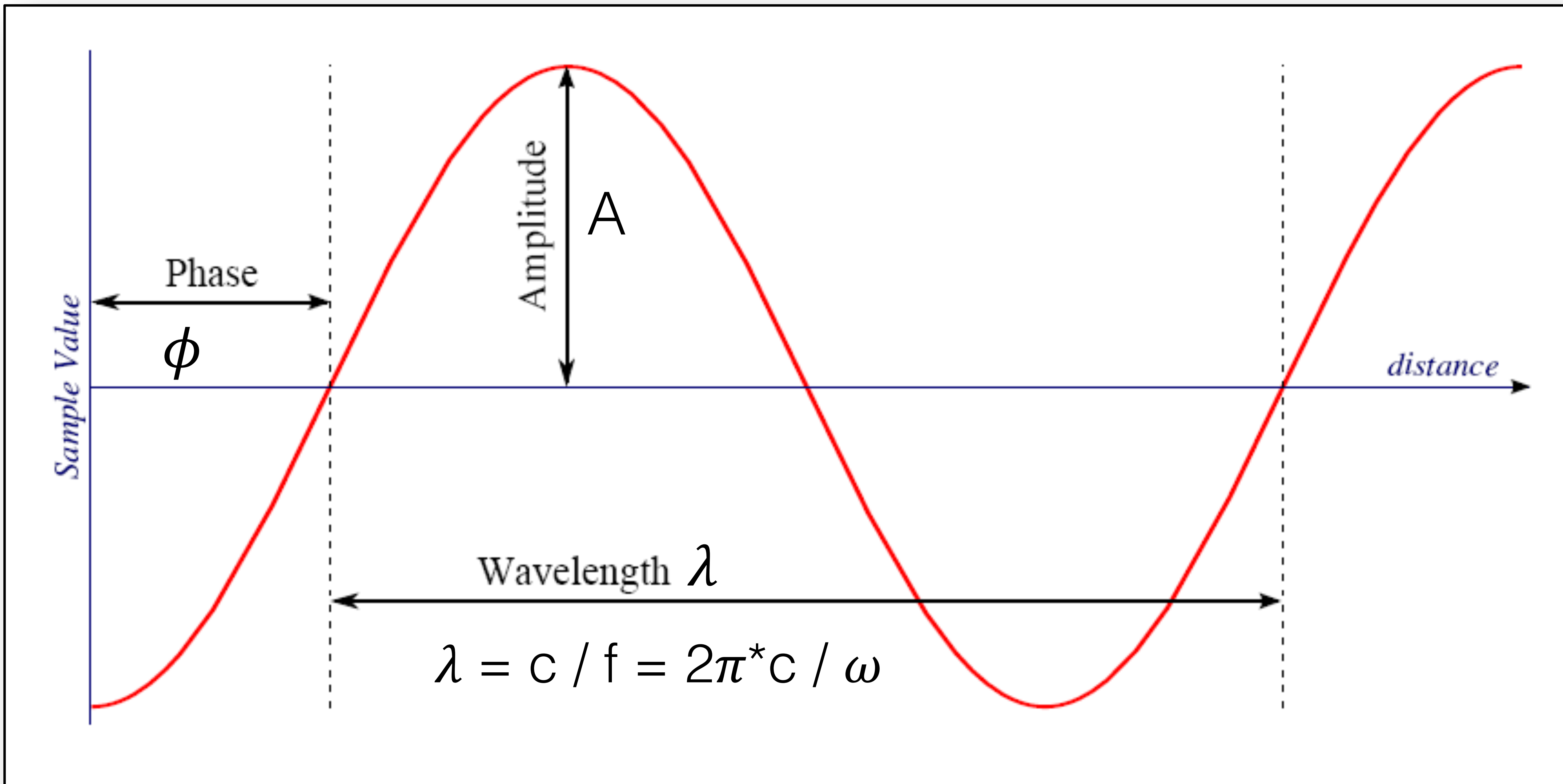
- Challenged by a number of mathematicians:
  - Including: Laplace, Lagrange, Legendre
- Fourier's idea was not even translated until 1878
- BUT, it's mostly true
  - It's called the Fourier Series
  - There are some subtle restrictions



http://www-history.mcs.st-and.ac.uk/Biographies/Fourier.html

Sinusoids $f(x) = A*\sin(\omega x + \phi)$

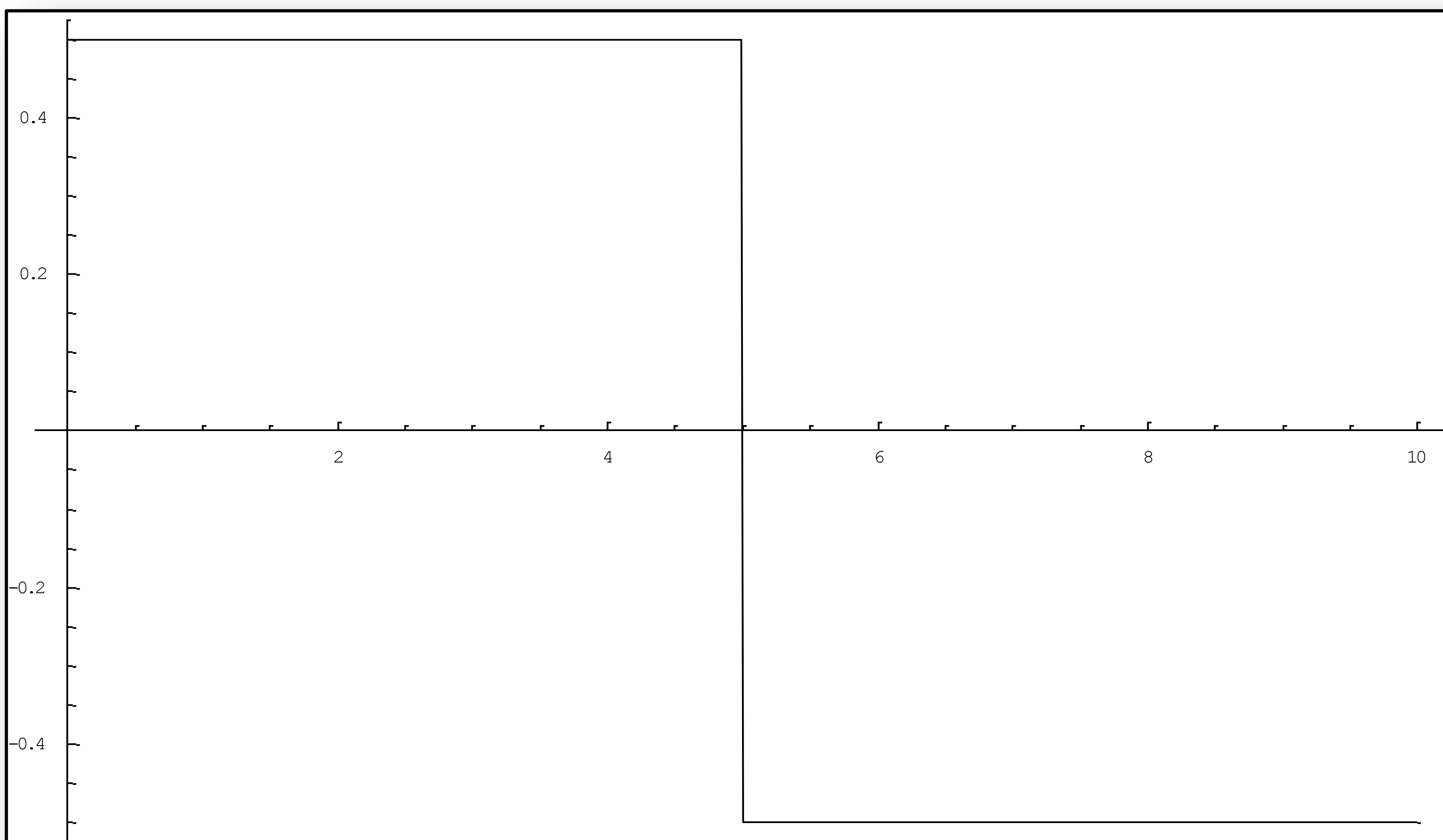
- A sinusoidal is characterized by the amplitude (A), frequency ($\omega = 2\pi f$), and phase (ϕ)



Sums of Sinusoids

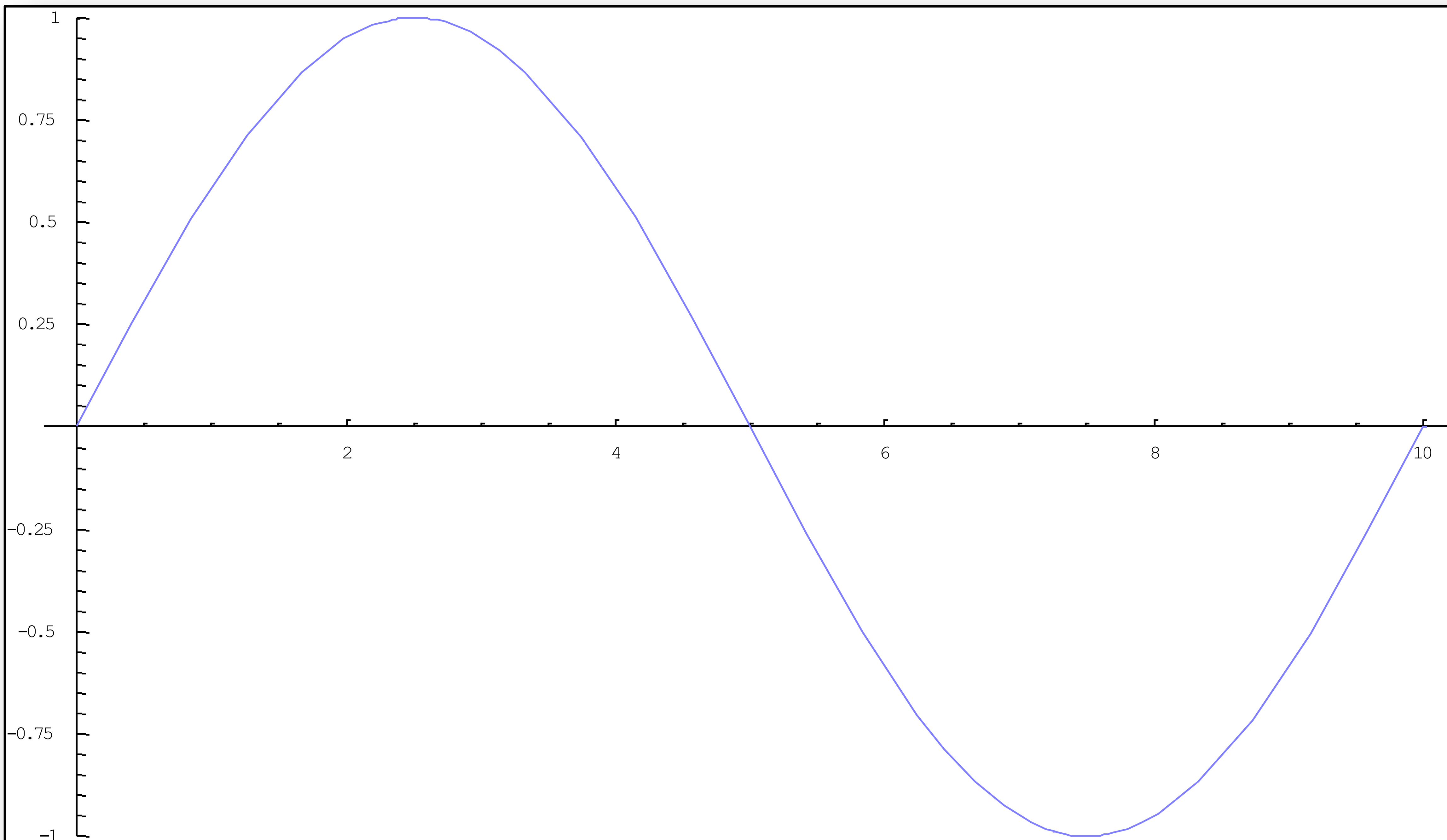
Functions into sinusoids

- Any function can be represented as the sum of sinusoids
- Consider a one-dimensional square wave.
- Can it be represented as the sum of ‘non-square waves’?



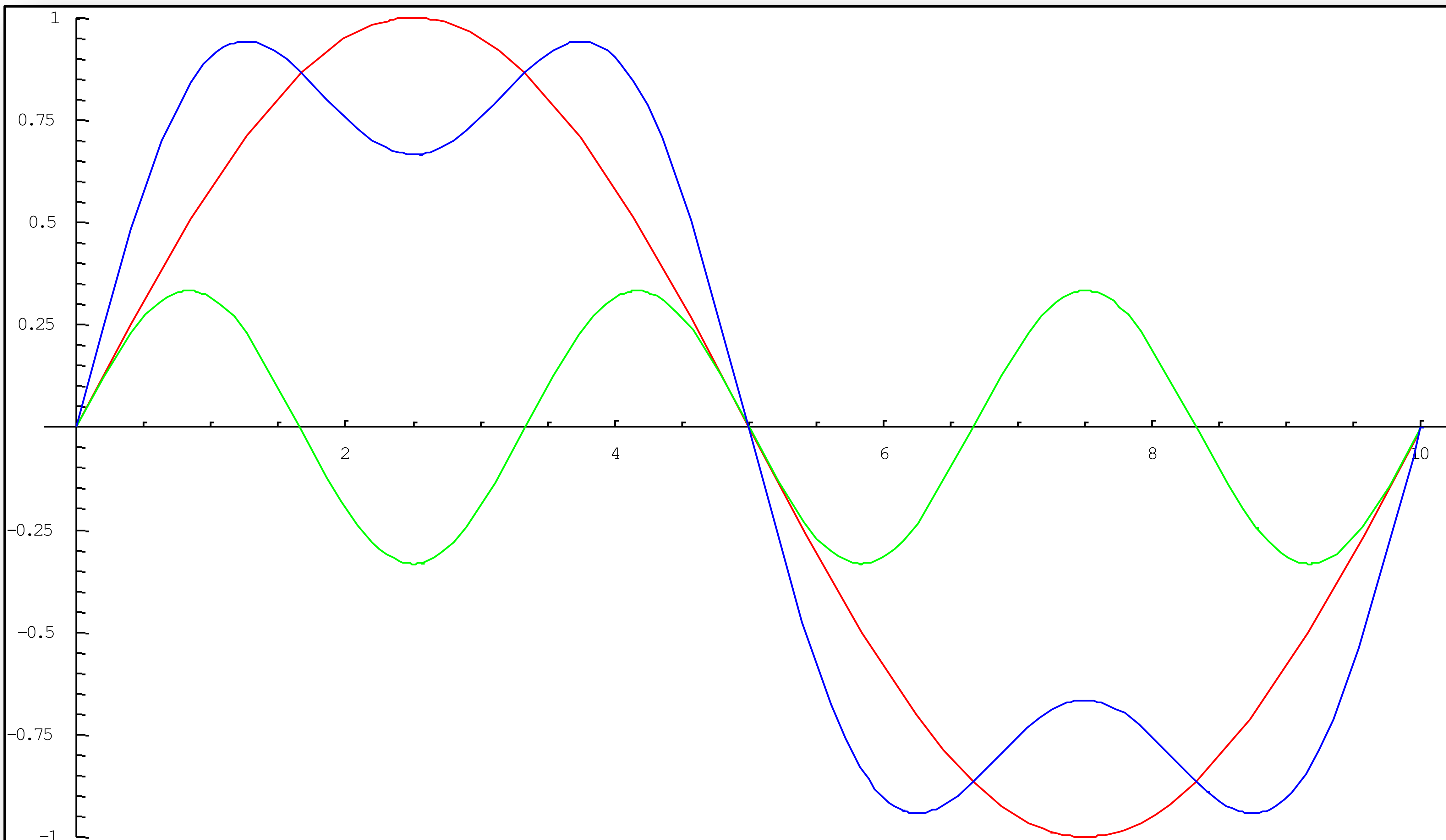
Decomposing functions into sinusoids

- Start with a sin wave of the same frequency as the square wave.
- This is the "base" or "fundamental" frequency.



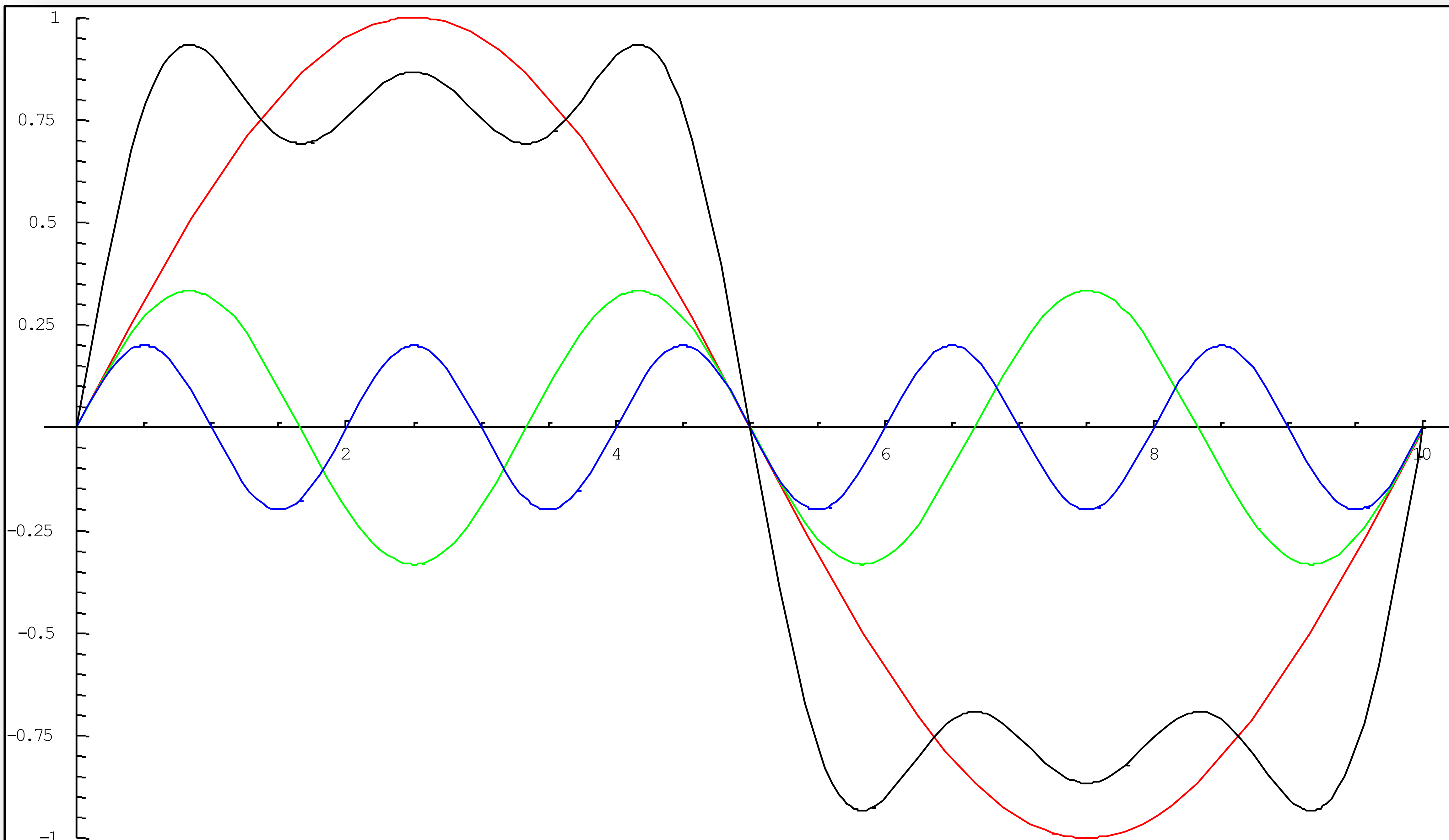
Decomposing functions into sinusoids

- Add a 3rd "harmonic" (green) to the fundamental frequency
- The amplitude is less than the base and the frequency is 3 times that of the base.



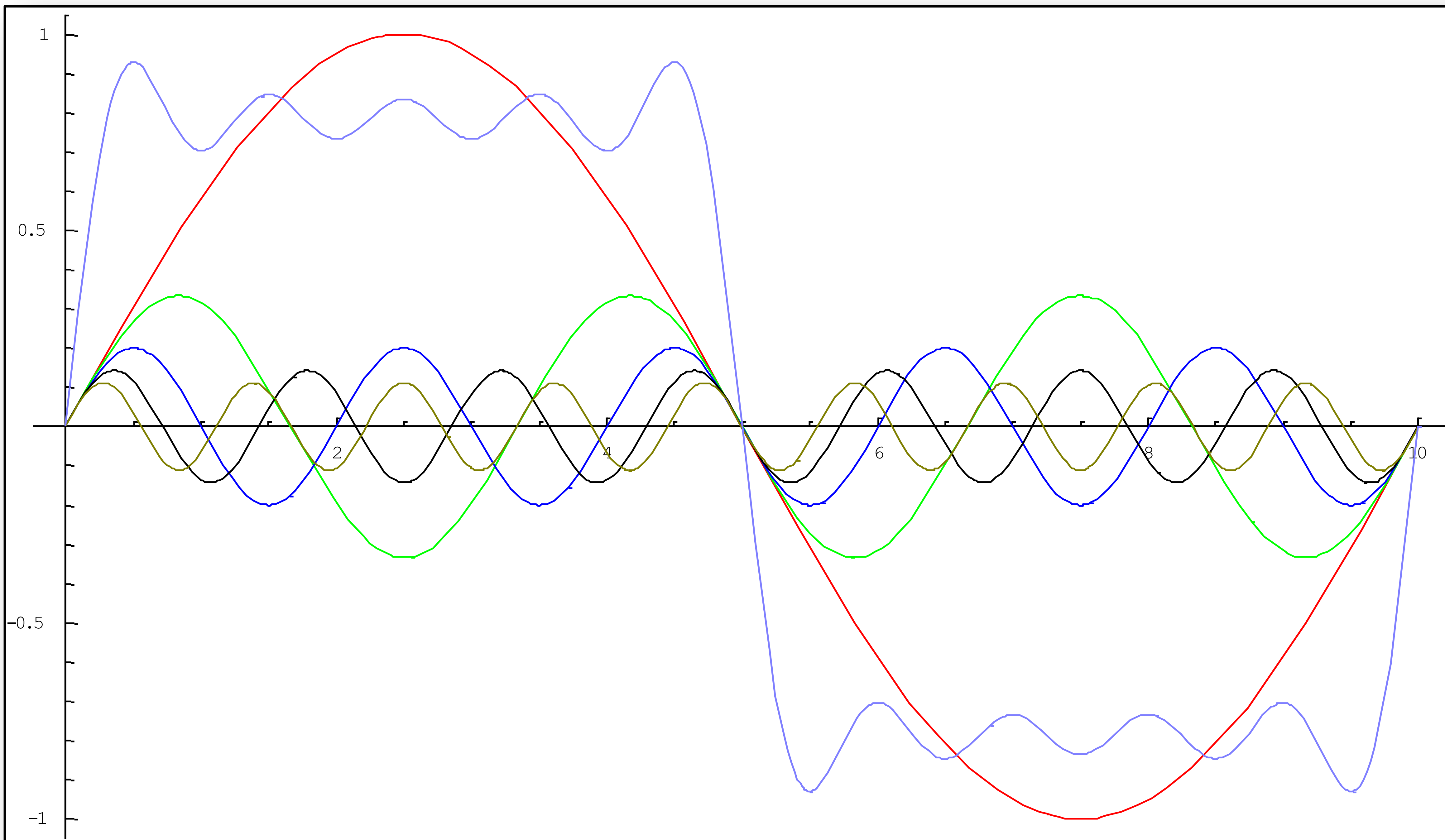
Decomposing functions into sinusoids

- Add a 5rd “harmonic” (blue) to the fundamental frequency
- The amplitude is less than the base and the frequency is 5 times that of the base.



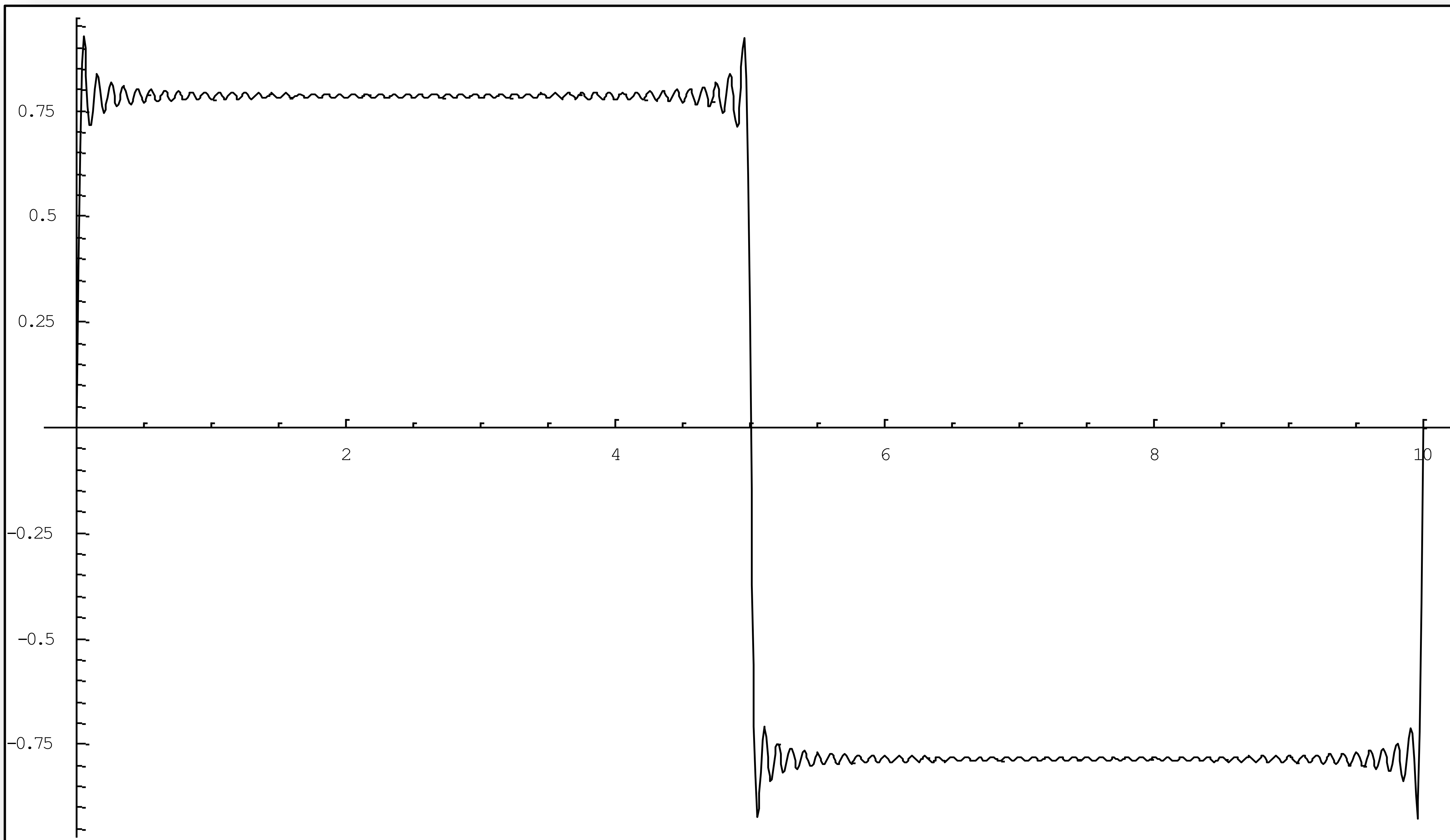
Decomposing functions into sinusoids

- Add a 7th (black) and 9th (taupe) "harmonic" to the fundamental frequency



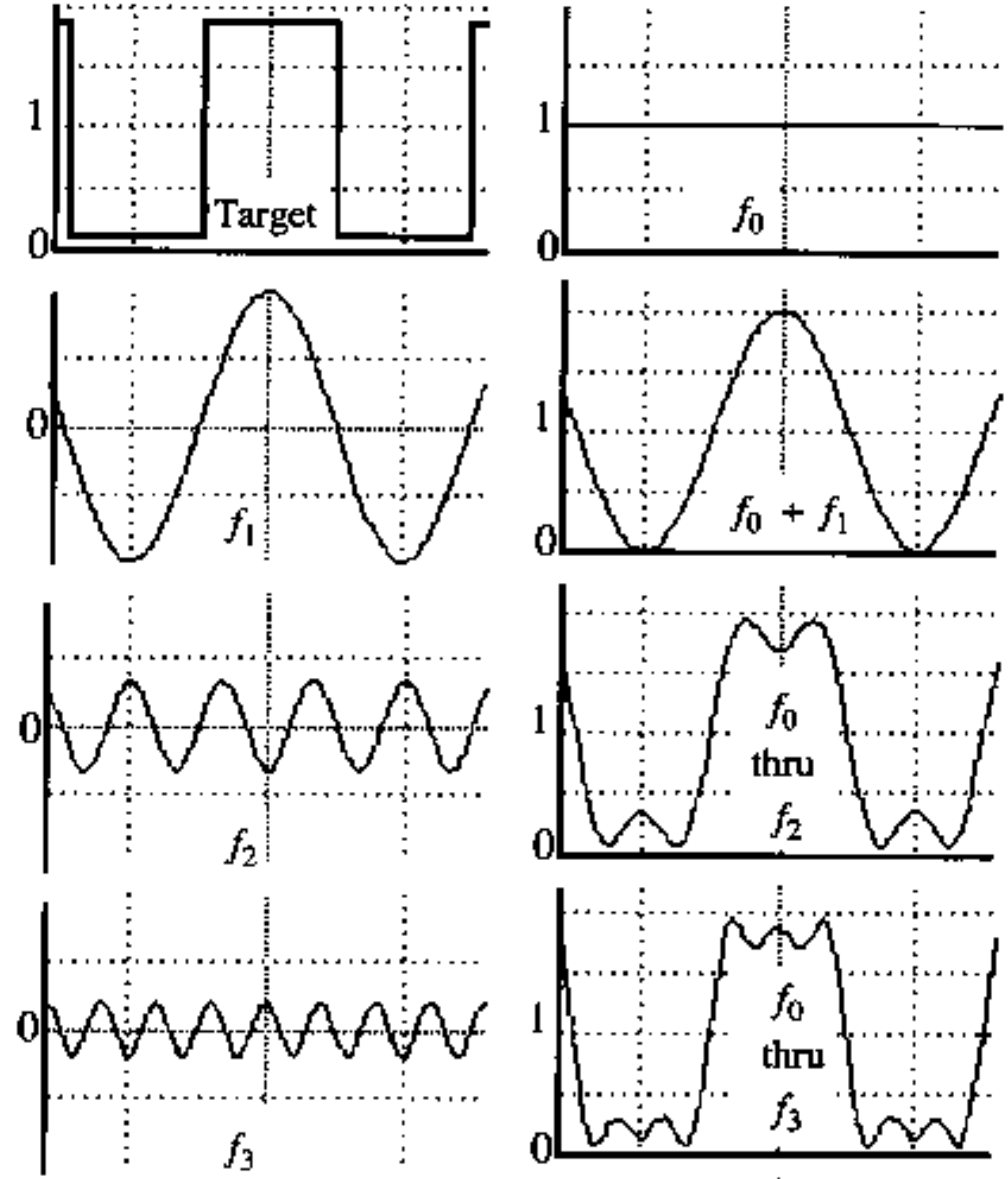
Decomposing functions into sinusoids

- Adding all harmonics up to the 100th



A Sum of Sines

- Our building block:
    - $f(x) = A*\sin(\omega x + \phi)$
- Add enough of them to get any signal you want!
- How many degrees of freedom?
- What does each control?
- Which one encodes the coarse vs. fine structure of the signal?



f(target)=
$f_1 + f_2 + f_3 ... + f_n + ...$

Sinusoids (one-D) as Images

$$f(x) = A * \sin(\frac{2\pi ux}{N} + \phi)$$

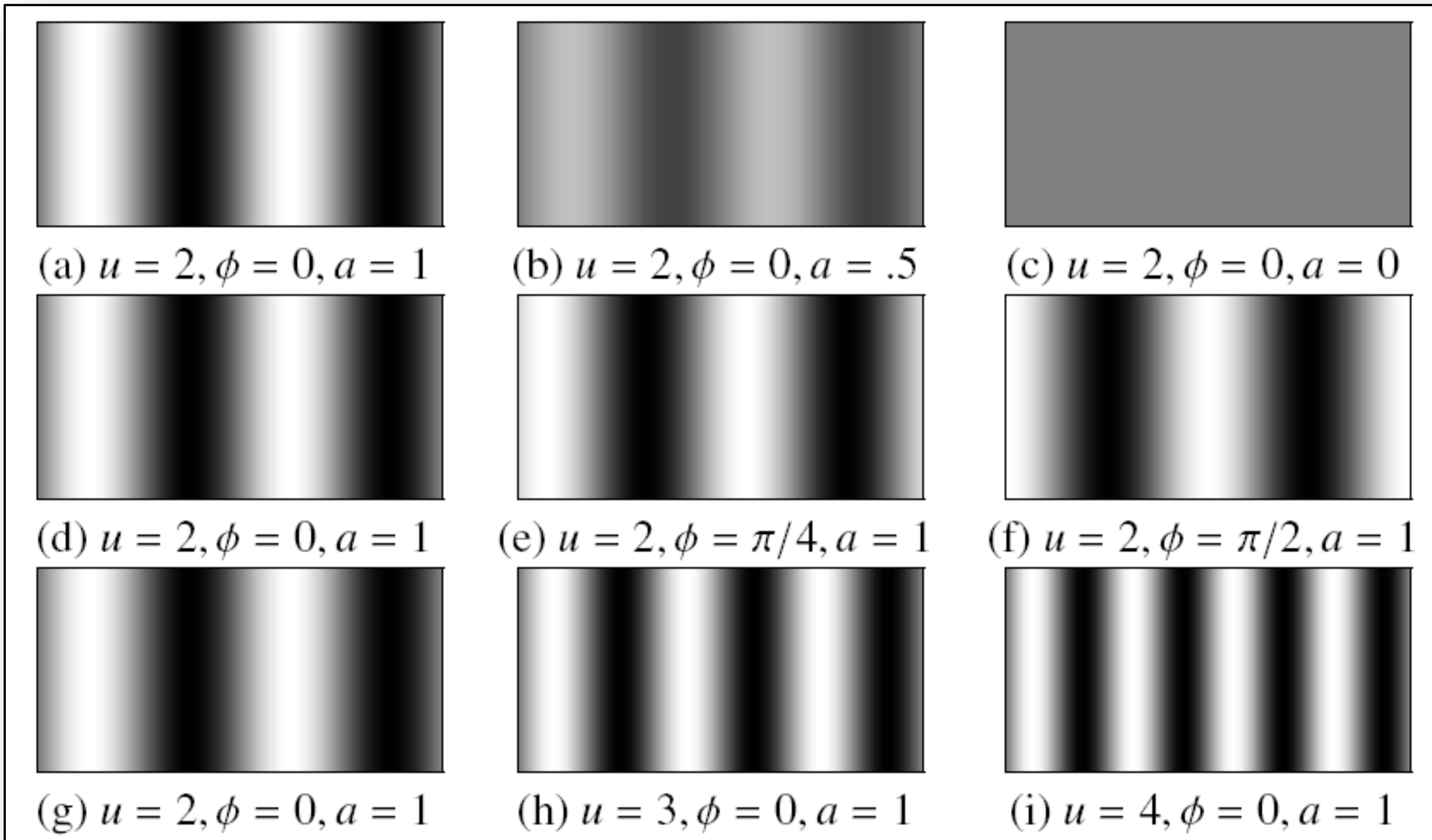
Breakdown frequency by:

- N is the width of the image
- u controls number of cycles

Varying Amplitude

Varying Phase

Varying Frequency



(a) $u = 2, \phi = 0, a = 1$ (b) $u = 2, \phi = 0, a = .5$ (c) $u = 2, \phi = 0, a = 0$

(d) $u = 2, \phi = 0, a = 1$ (e) $u = 2, \phi = \pi/4, a = 1$ (f) $u = 2, \phi = \pi/2, a = 1$

(g) $u = 2, \phi = 0, a = 1$ (h) $u = 3, \phi = 0, a = 1$ (i) $u = 4, \phi = 0, a = 1$

Shannon-Nyquist Theorem

Discrete Sinusoid

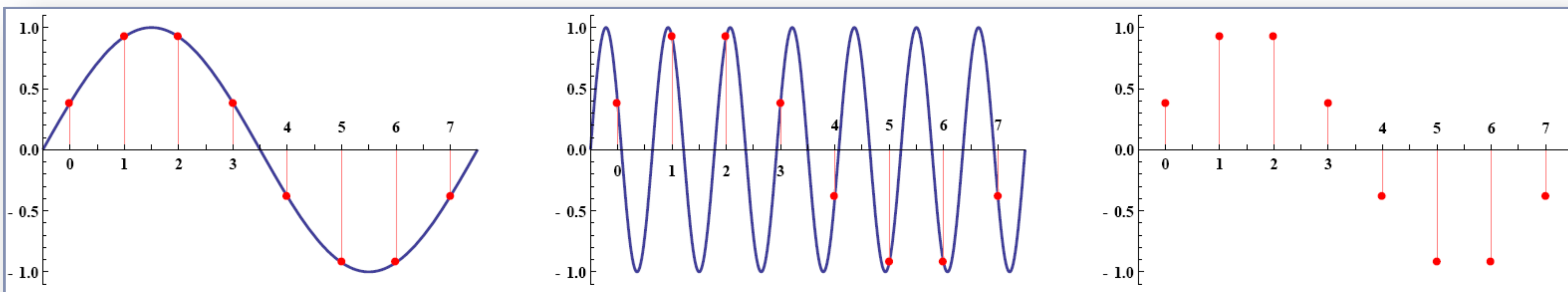
- Consider an “image” of samples of some sinusoid.
- Take the continuous sinusoidal function into the discrete domain via sampling at ½ unit intervals (“pixels”).

$$f(x) = A * \sin\left(\frac{2\pi u}{N}(x + 1/2) + \phi\right) \quad x \in [0, N-1]$$

- One effect is to place an upper limit on the frequencies that can be captured via sampling.
- u must be less than N/2 in order to be recoverable by the discrete samples. This is the Shannon-Nyquist limit.
- Higher frequencies generate aliases

Aliasing

- (image a) Consider a sinusoid such that u=1 over a span of N = 8 units.
- (image b) Consider a sinusoid such that u=7 over a span of N = 8 units.
- The resulting samples are identical. The two different signals are indistinguishable after sampling and hence are aliases for each other.

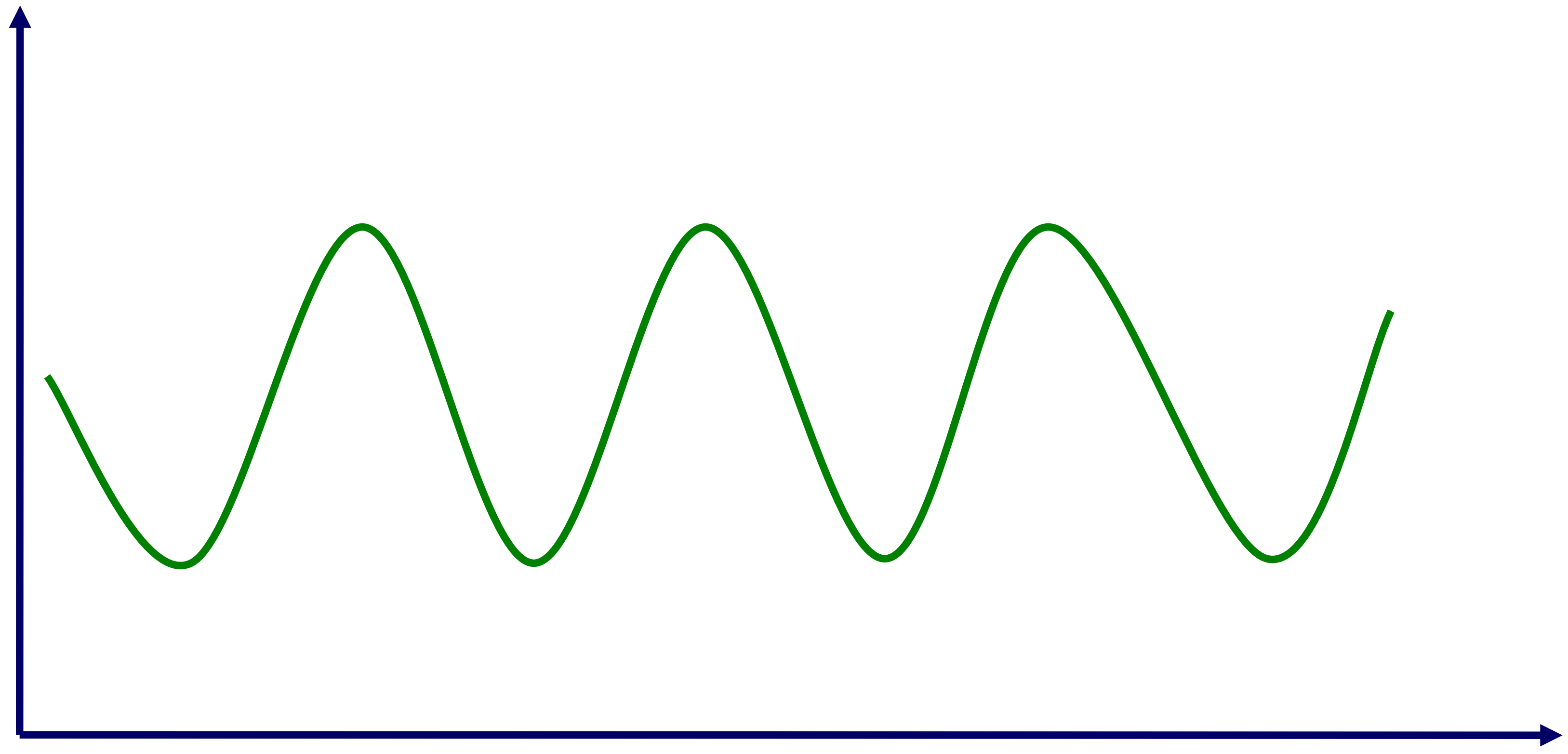


(a) $N > 2u$ (b) $N < 2u$ (c) Sampled signal

Recall: Sampling Rates

How many samples are enough to avoid aliasing?

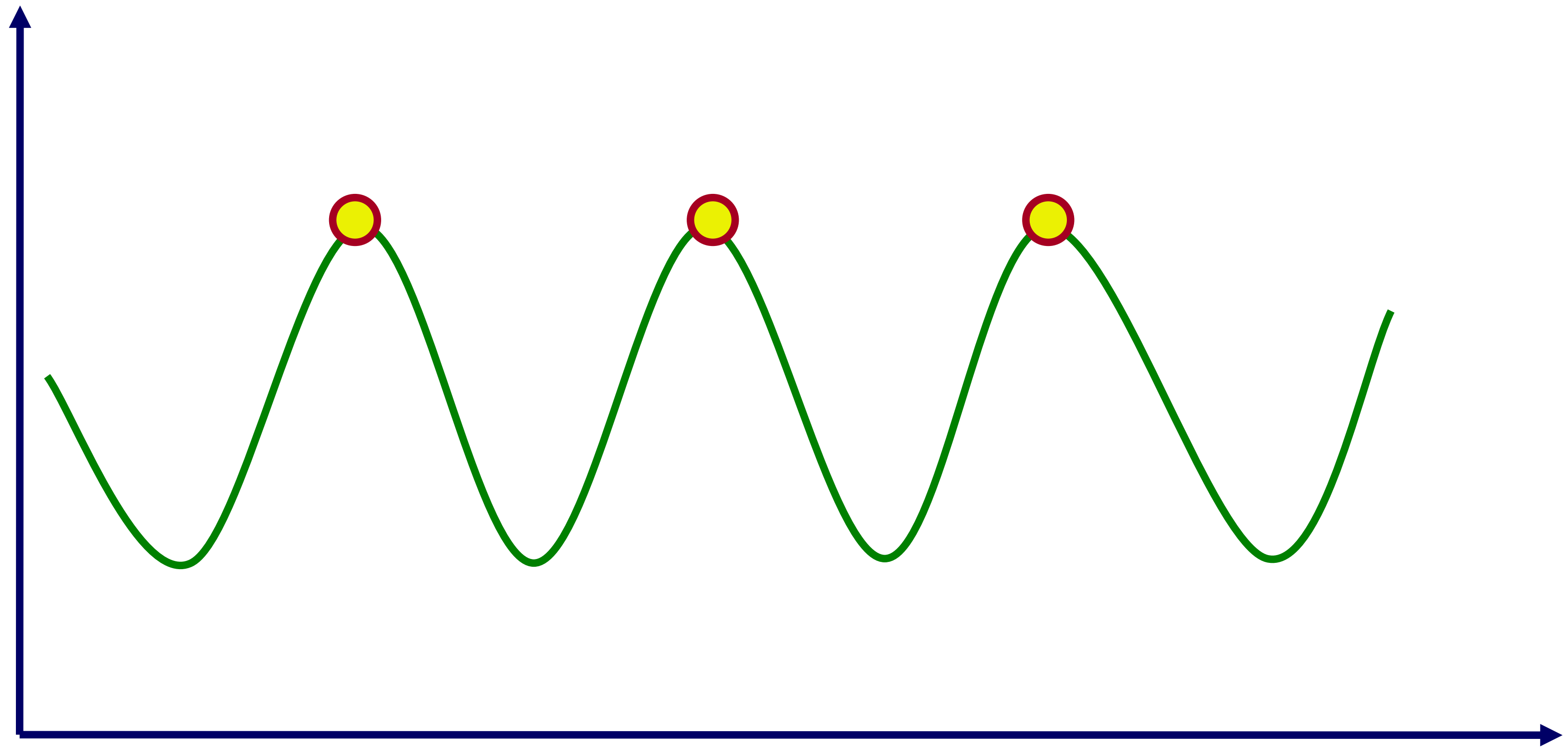
- How many samples are required to represent a given signal without loss of information?
- What signals can be reconstructed without loss for a given sampling rate?



Recall: Sampling Rates

How many samples are enough to avoid aliasing?

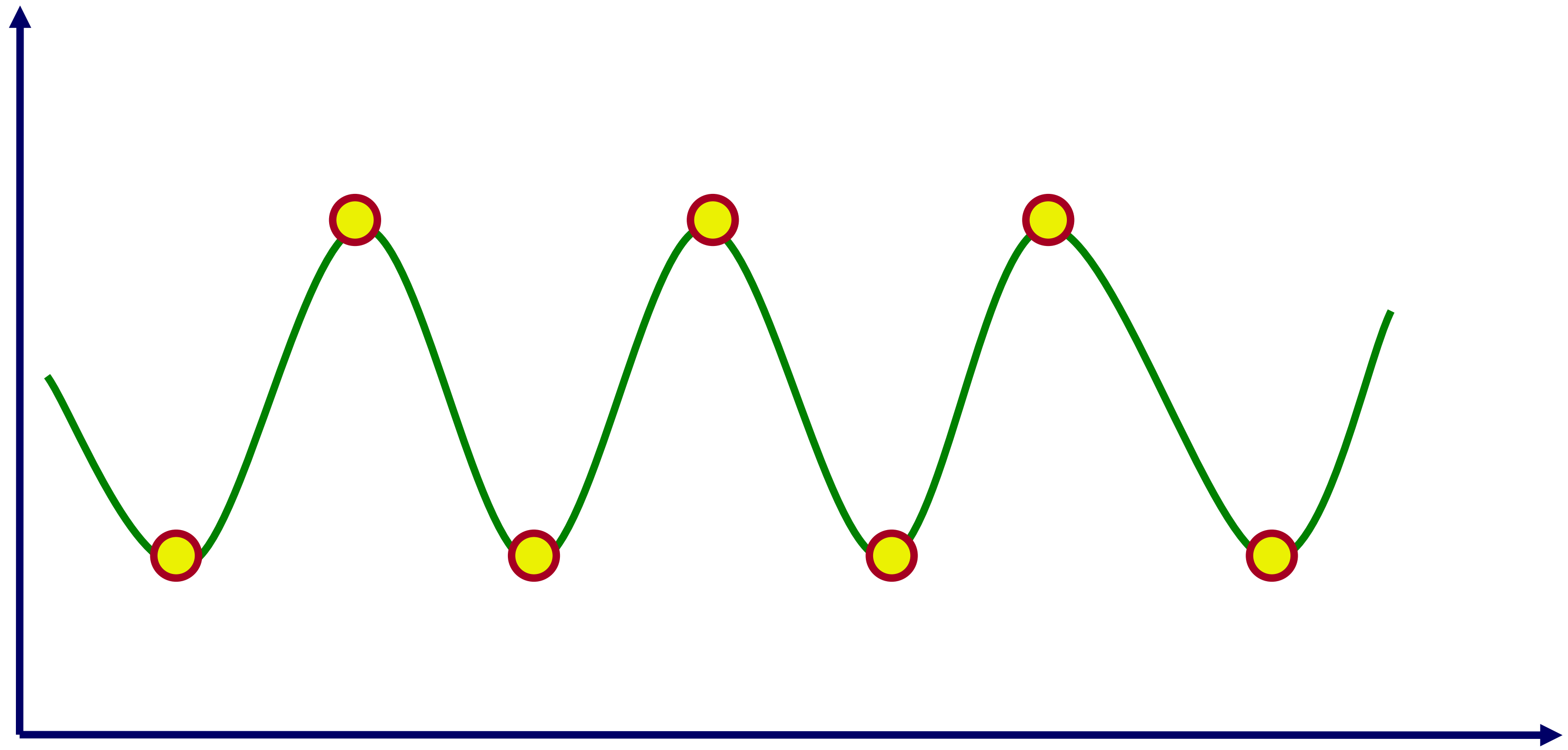
- How many samples are required to represent a given signal without loss of information?
- What signals can be reconstructed without loss for a given sampling rate?



Recall: Sampling Rates

How many samples are enough to avoid aliasing?

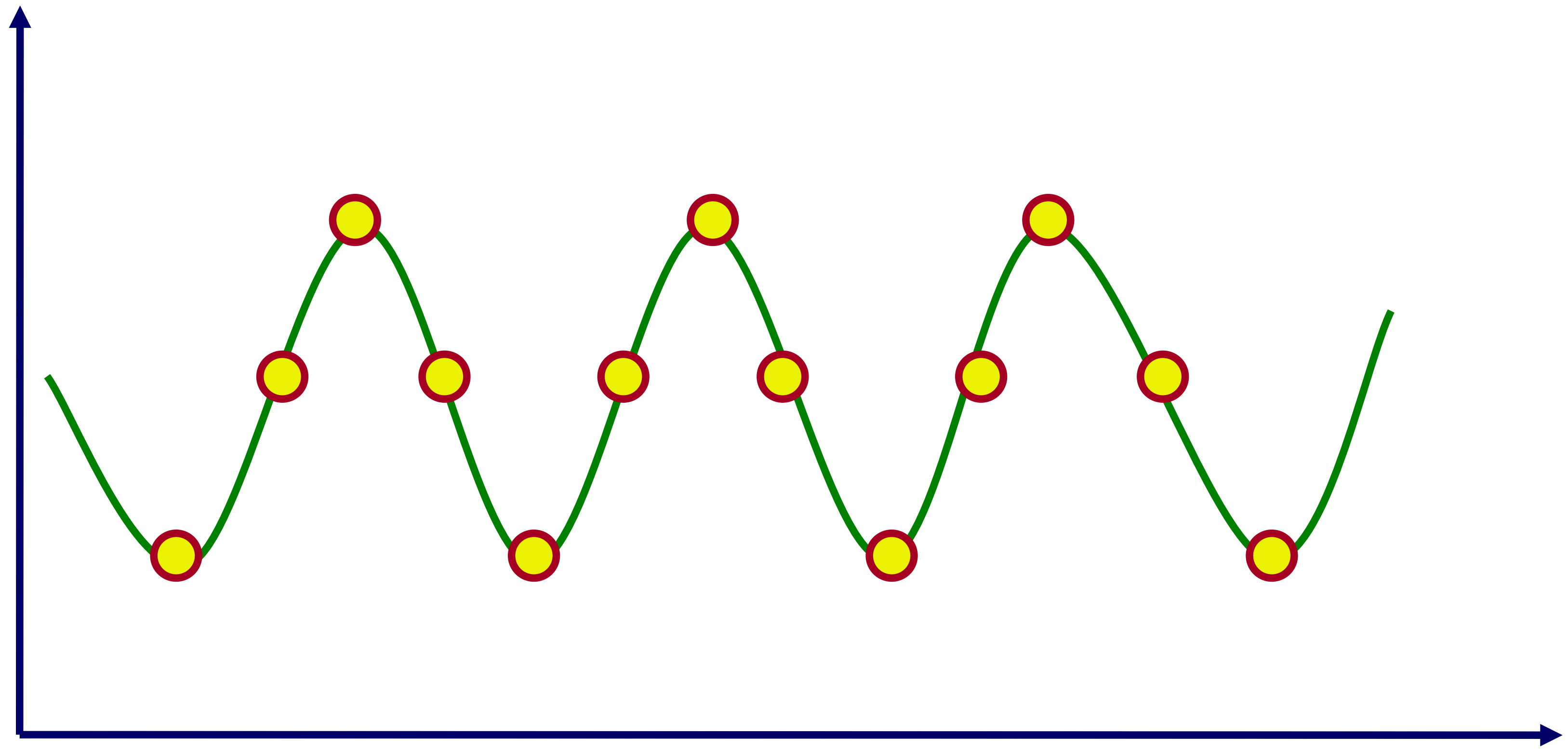
- How many samples are required to represent a given signal without loss of information?
- What signals can be reconstructed without loss for a given sampling rate?



Recall: Sampling Rates

How many samples are enough to avoid aliasing?

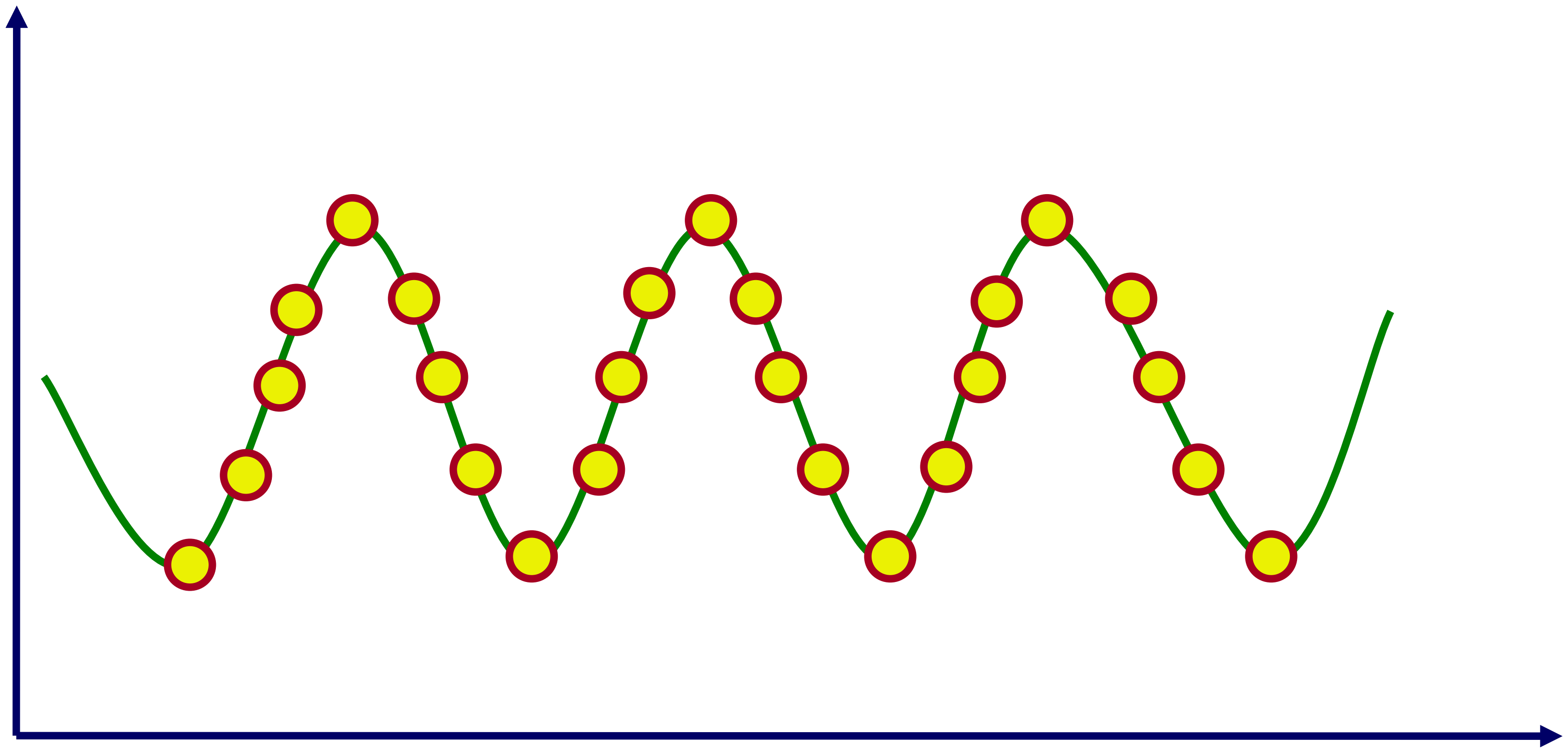
- How many samples are required to represent a given signal without loss of information?
- What signals can be reconstructed without loss for a given sampling rate?



Recall: Sampling Rates

How many samples are enough to avoid aliasing?

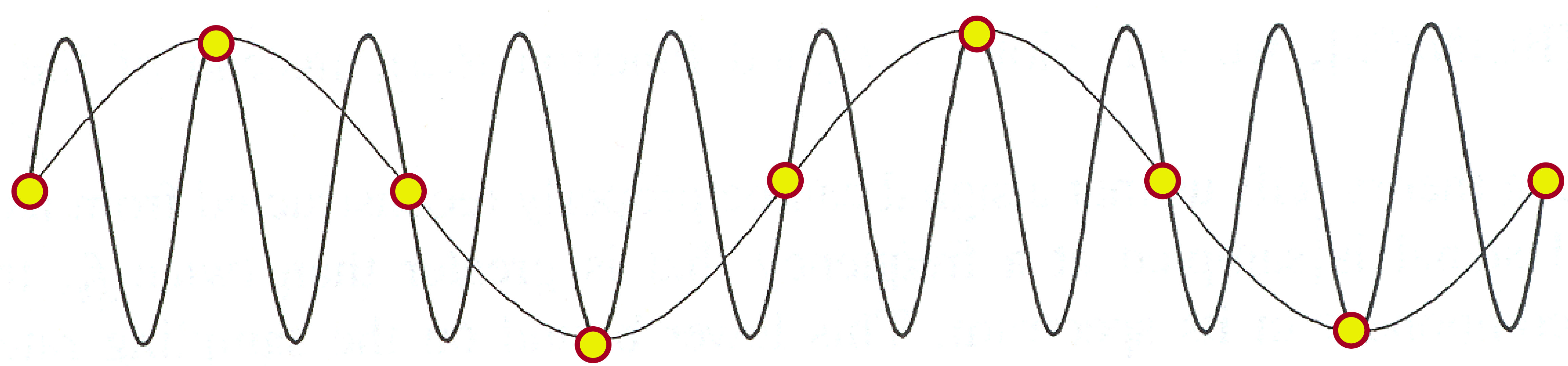
- How many samples are required to represent a given signal without loss of information?
- What signals can be reconstructed without loss for a given sampling rate?



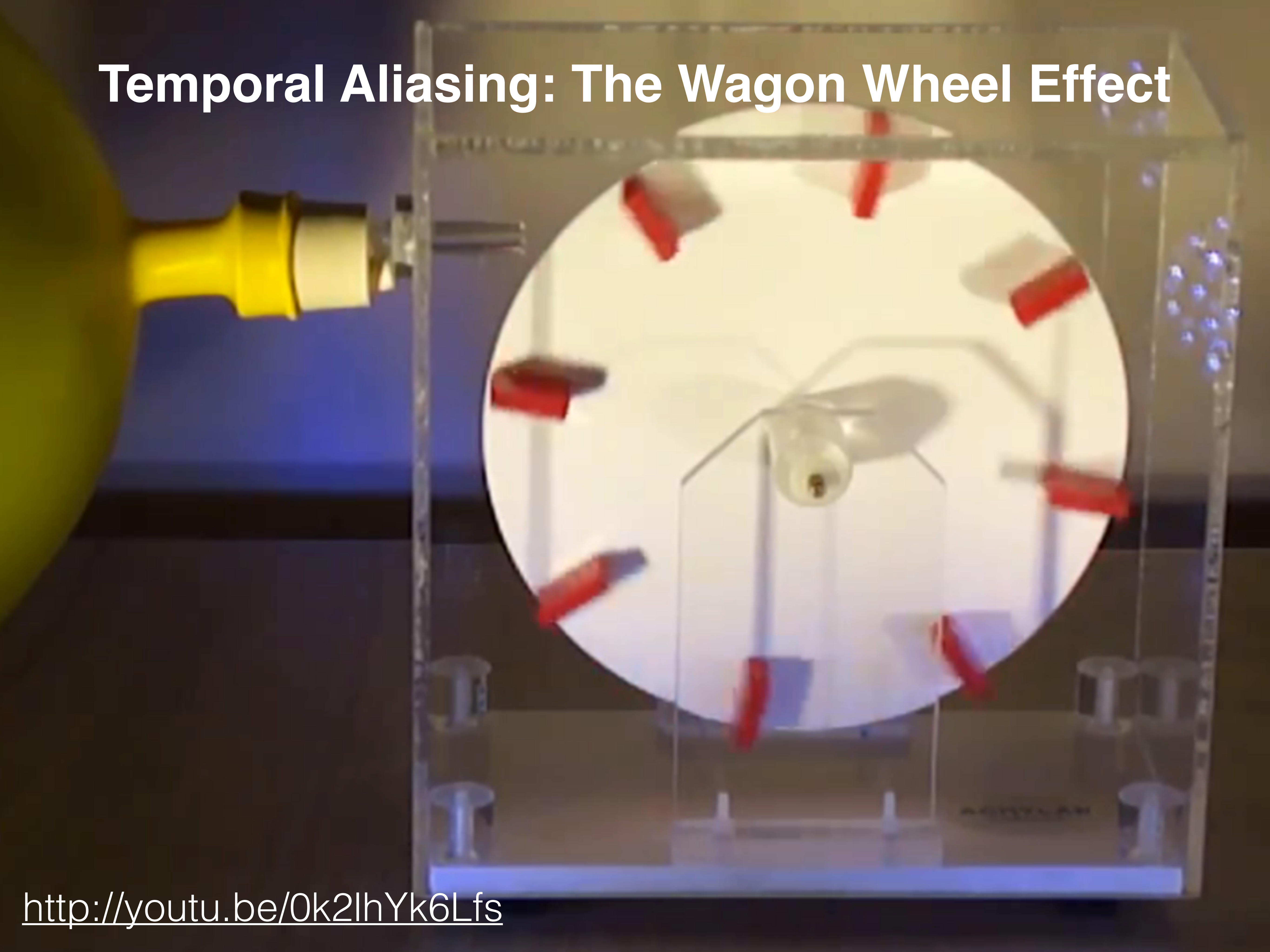
Sampling Theorem

A signal can be reconstructed from its samples, iff the original signal has no content >= 1/2 the sampling frequency - Shannon

Aliasing will occur if the signal is under-sampled



Temporal Aliasing: The Wagon Wheel Effect

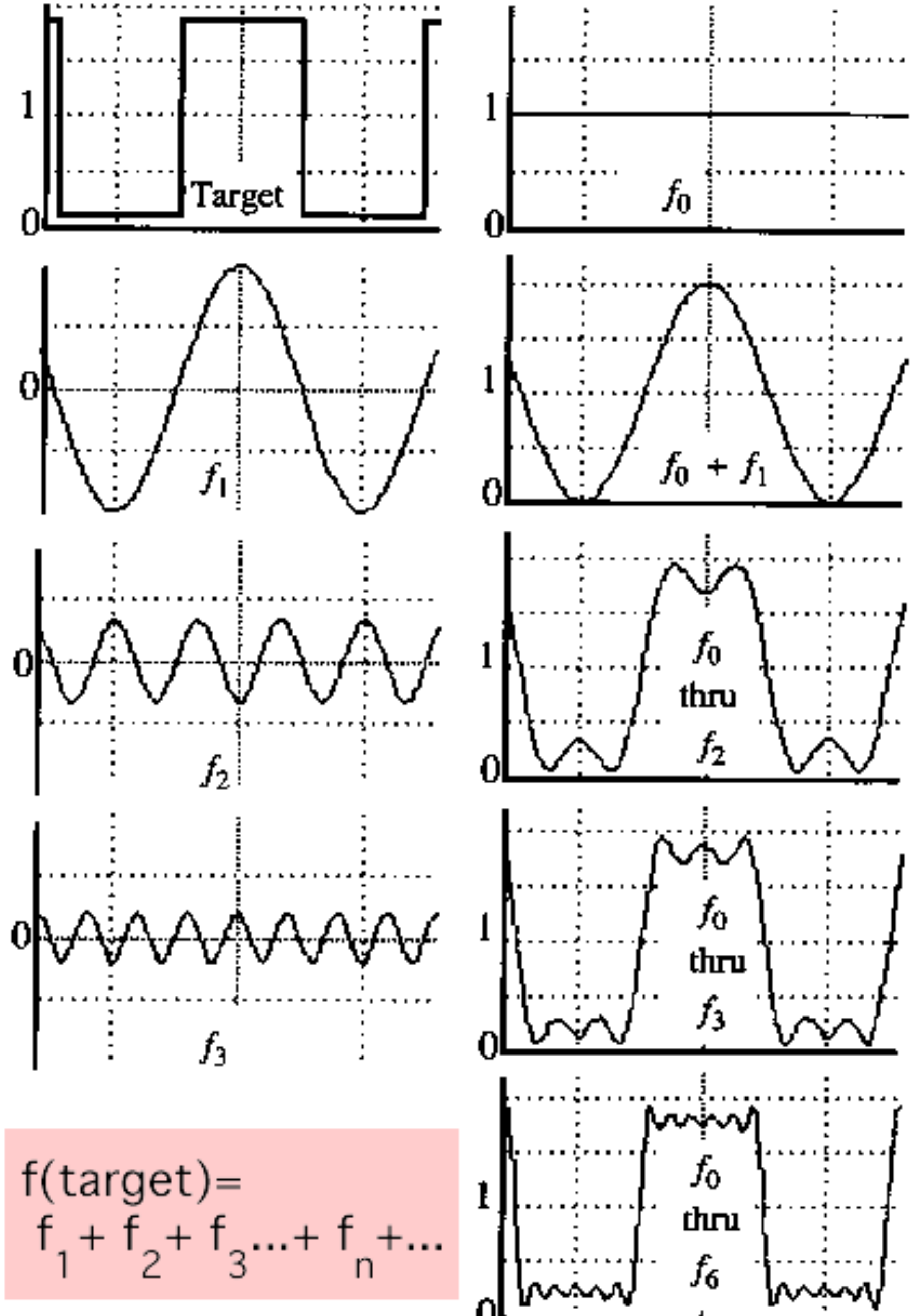


http://youtu.be/0k2IhYk6Lfs

Frequency Spectra

A Sum of Sines

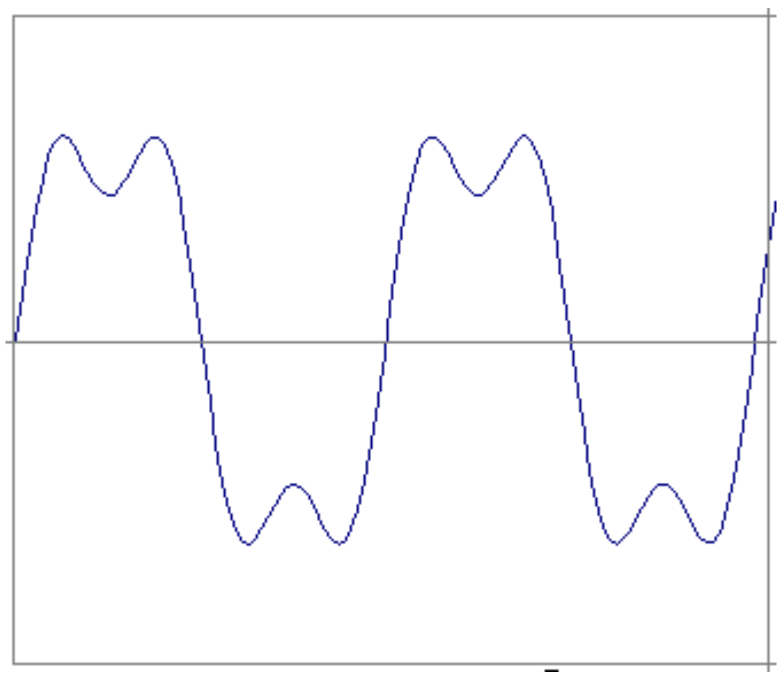
- Our building block:
  - $f(x) = A * \sin(\omega x + \phi)$
- Add enough of them to get any signal you want!
- How many degrees of freedom?
- What does each control?
- Which one encodes the coarse vs. fine structure of the signal?



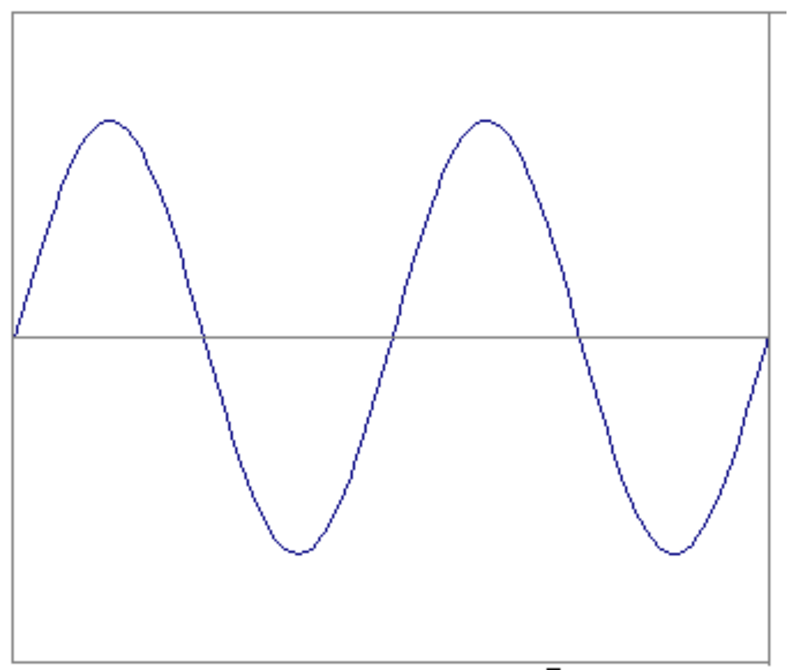
f(target)=
$f_1 + f_2 + f_3 ... + f_n + ...$

Frequency Spectra

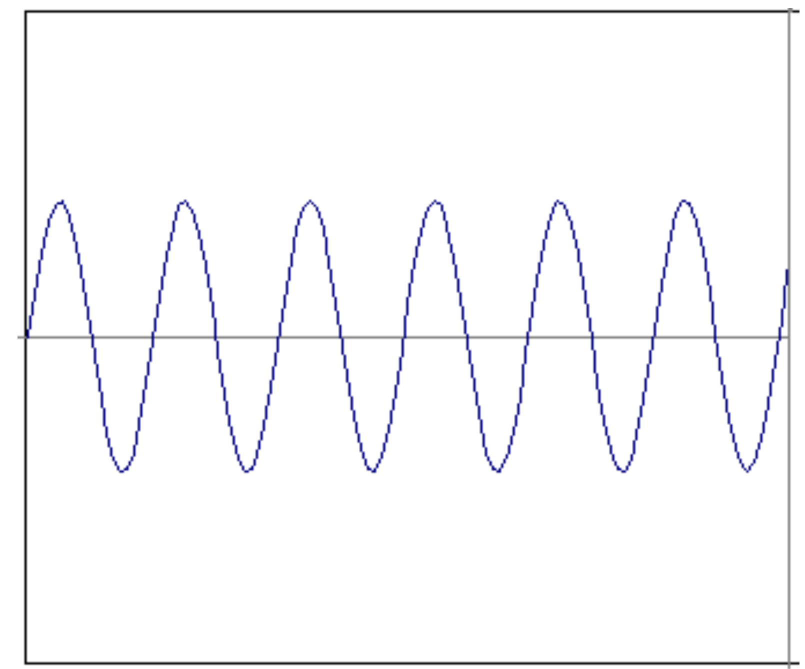
- example : $g(x) = \sin(2\pi f * x) + (1/3) \sin(2\pi(3f) * x)$



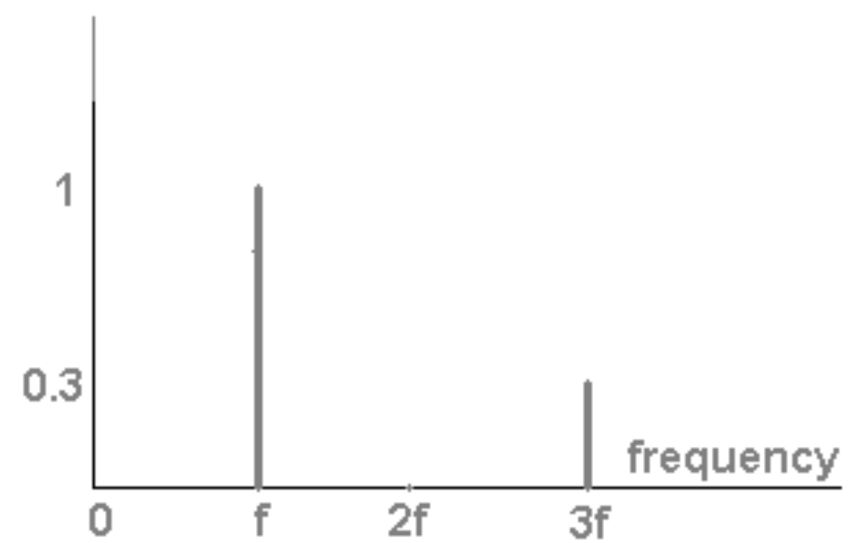
=



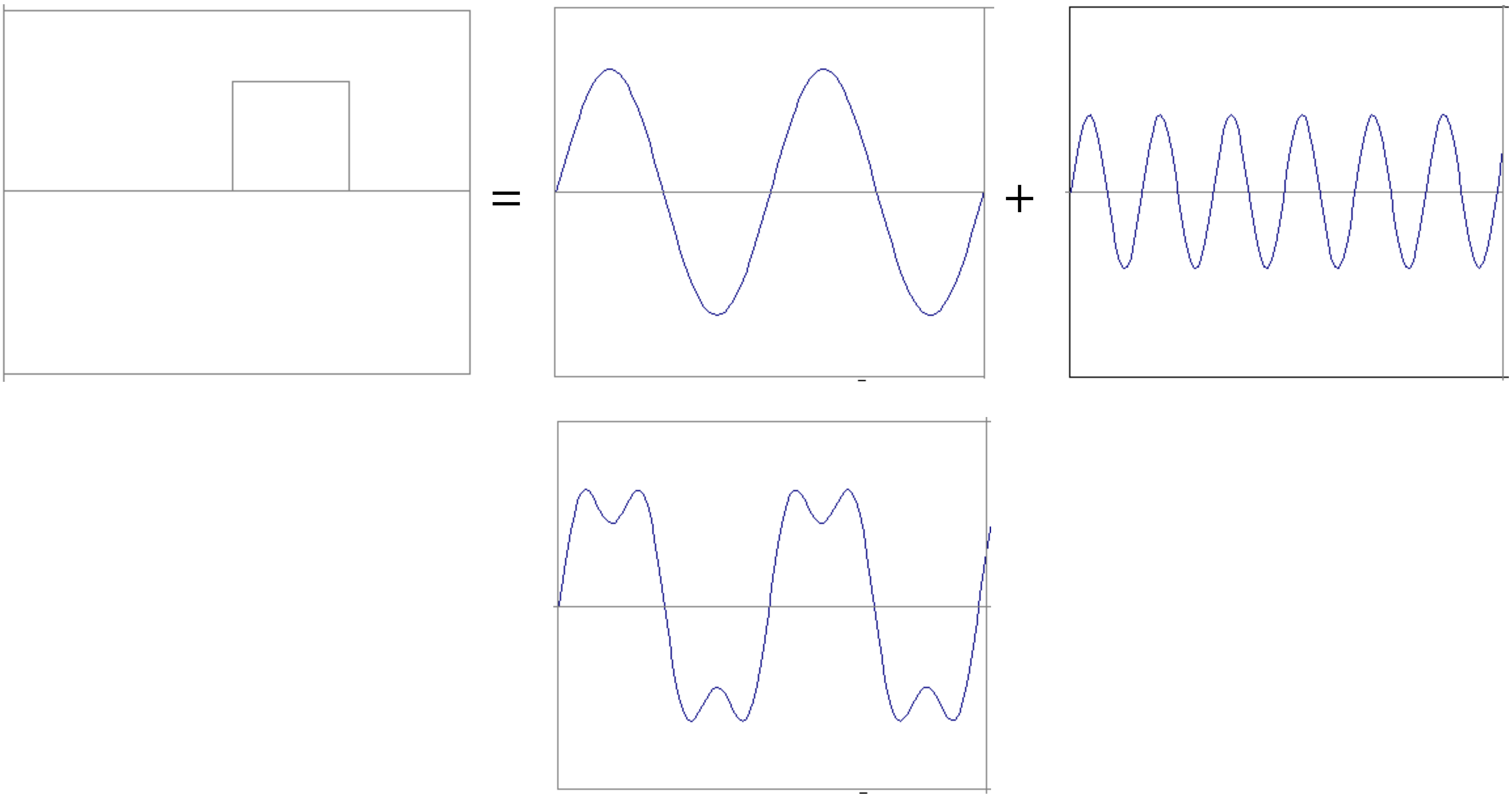
+



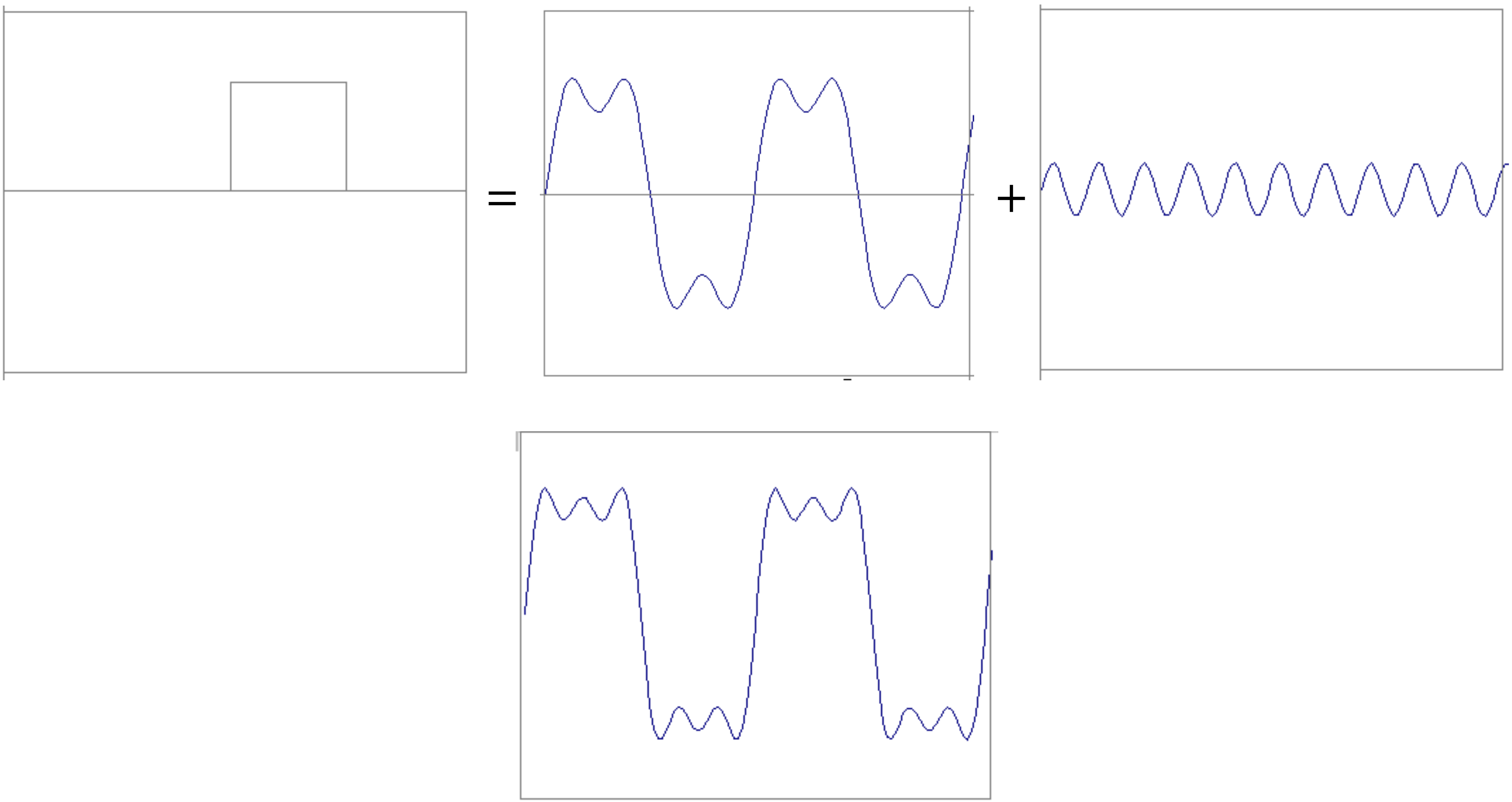
We can plot the amplitudes and frequencies to describe g(x)



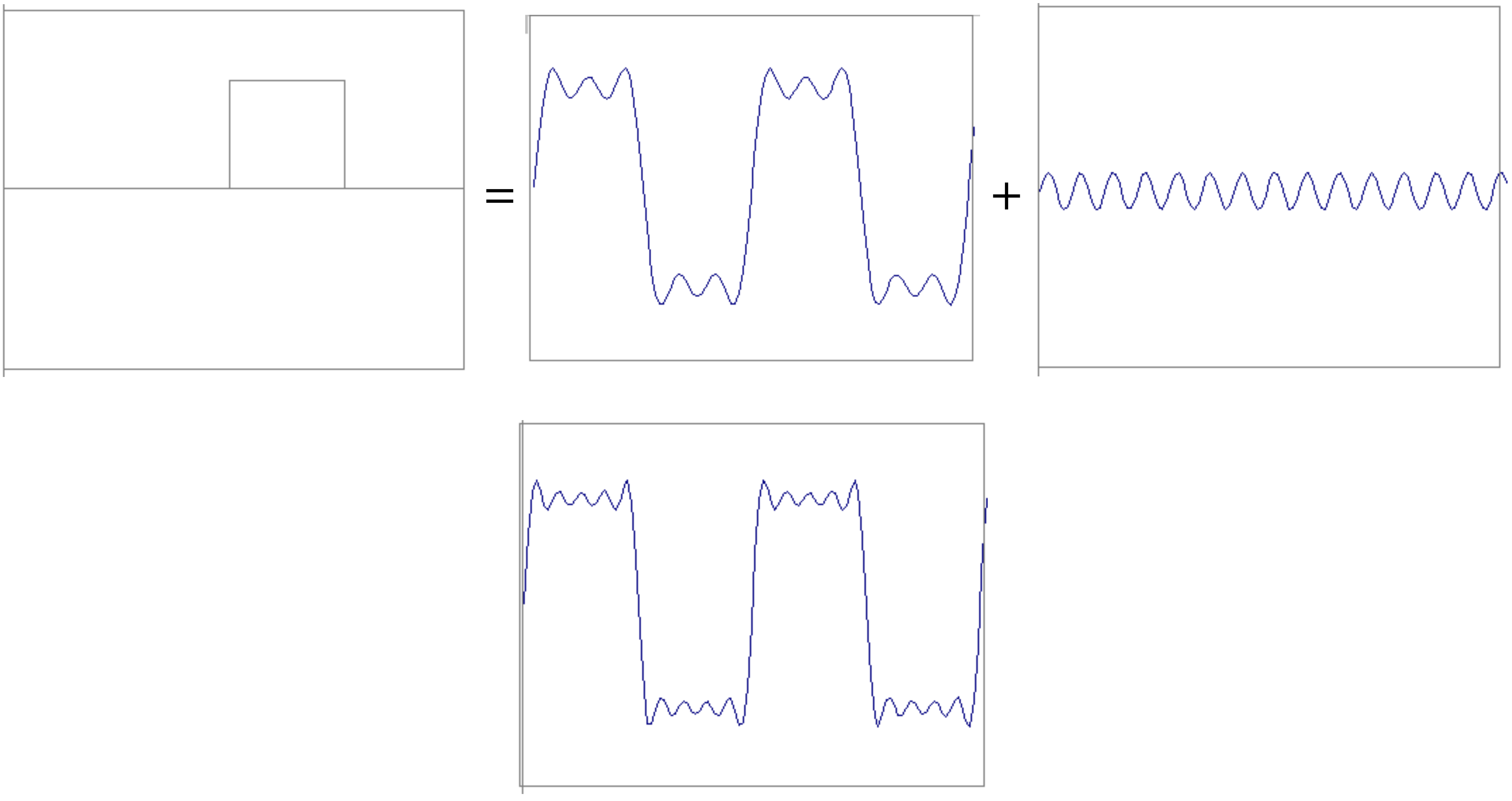
Frequency Spectra



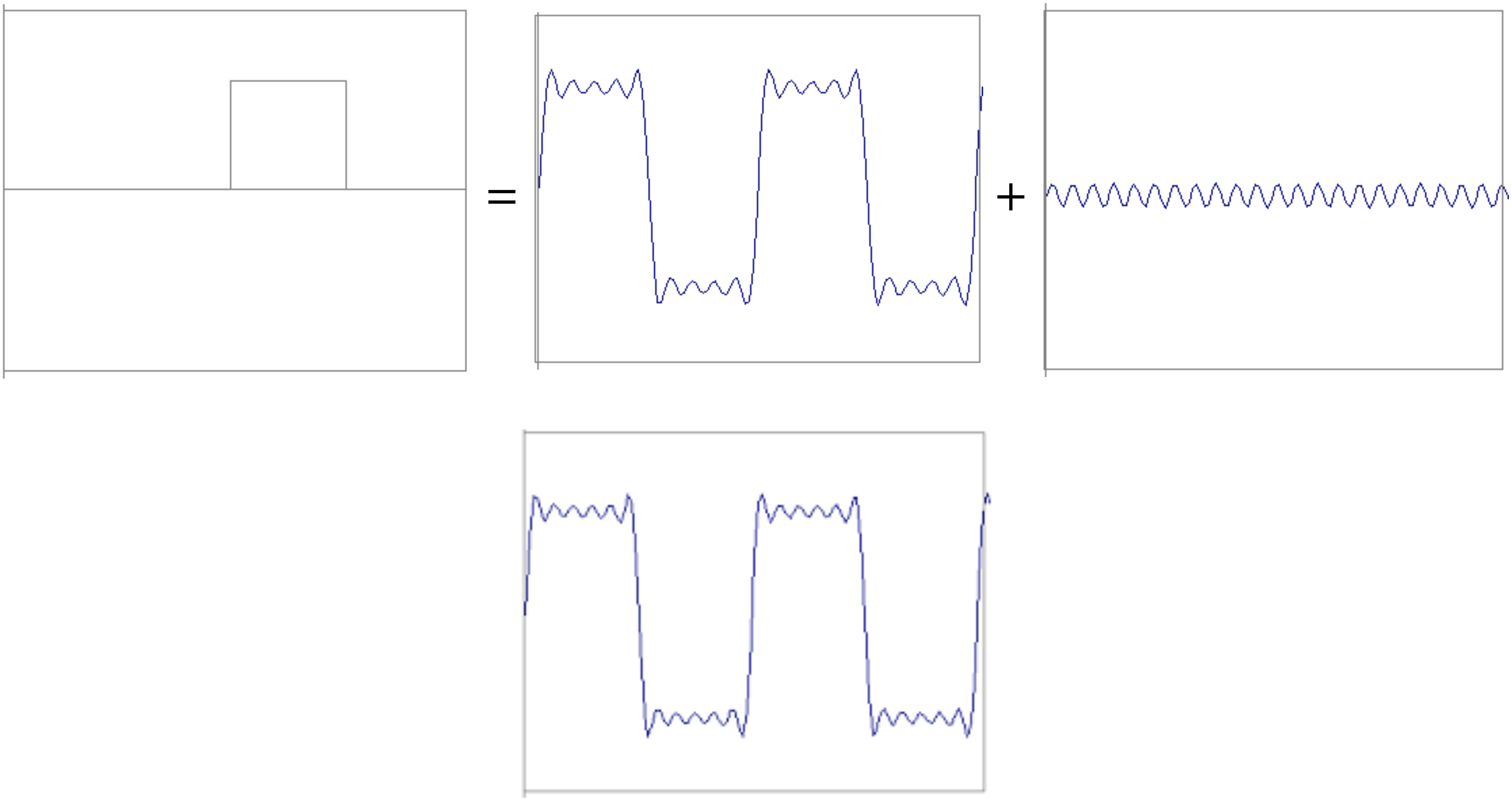
Frequency Spectra



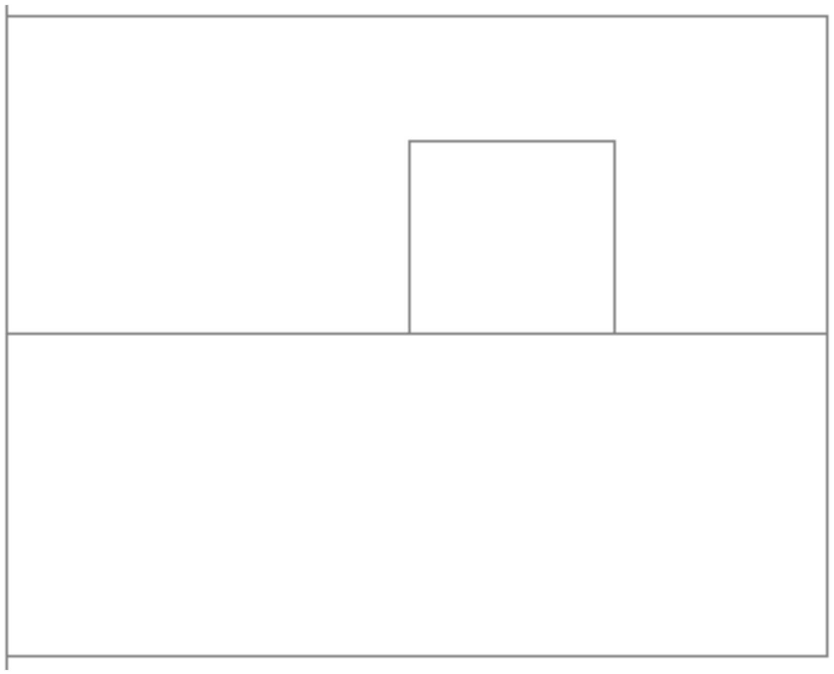
Frequency Spectra



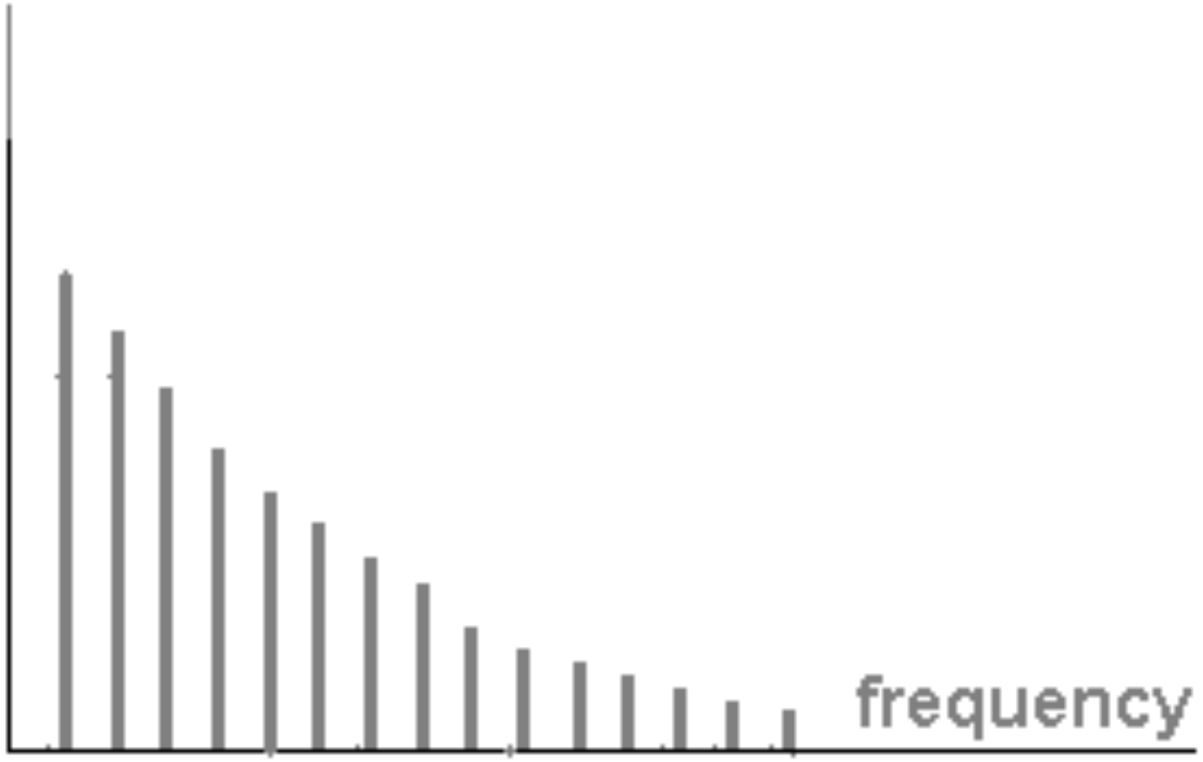
Frequency Spectra



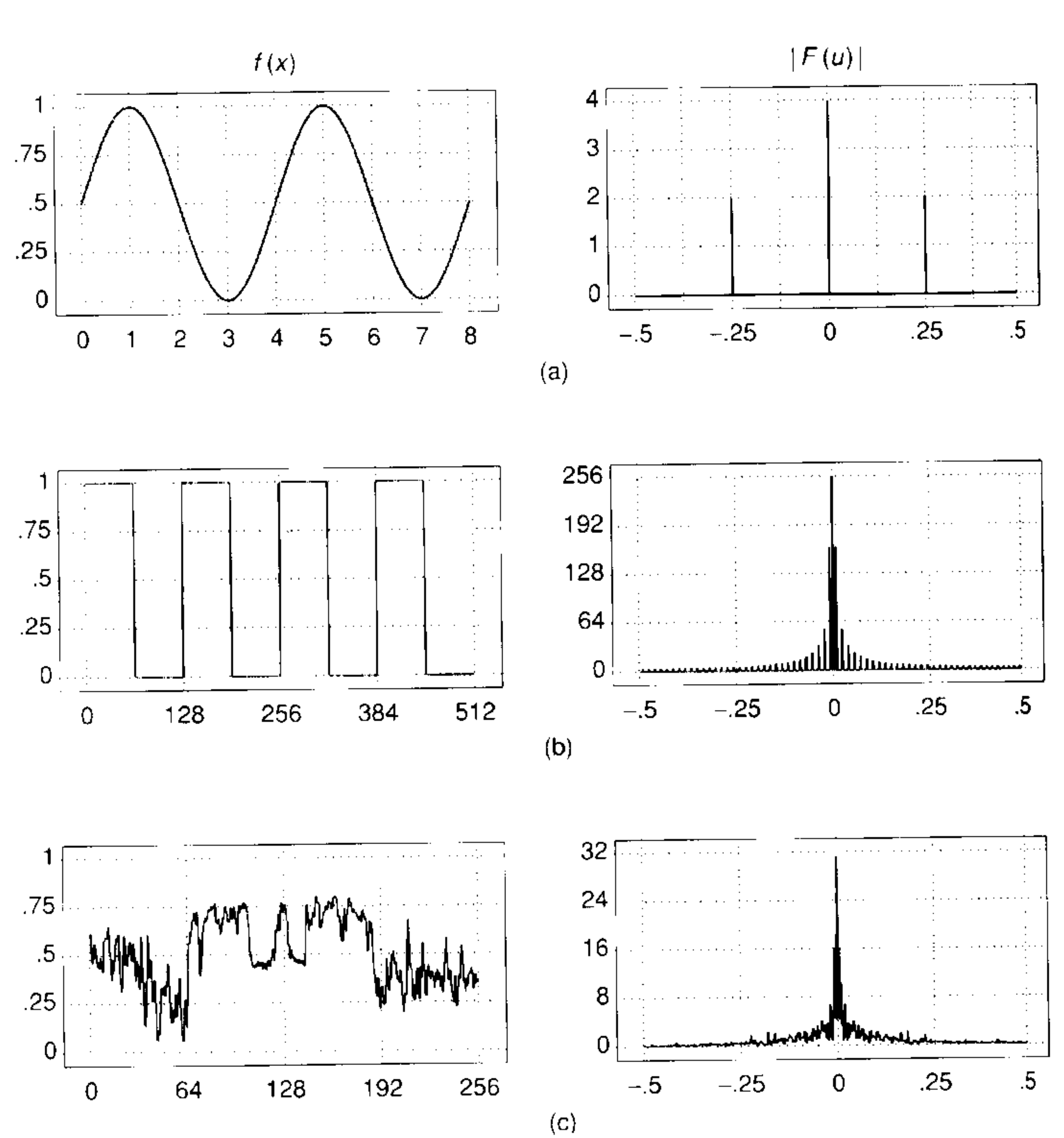
Frequency Spectra



$$= \boxed{A\sum_{k=1}^{\infty}\frac{1}{k}\sin(2\pi kt)}$$



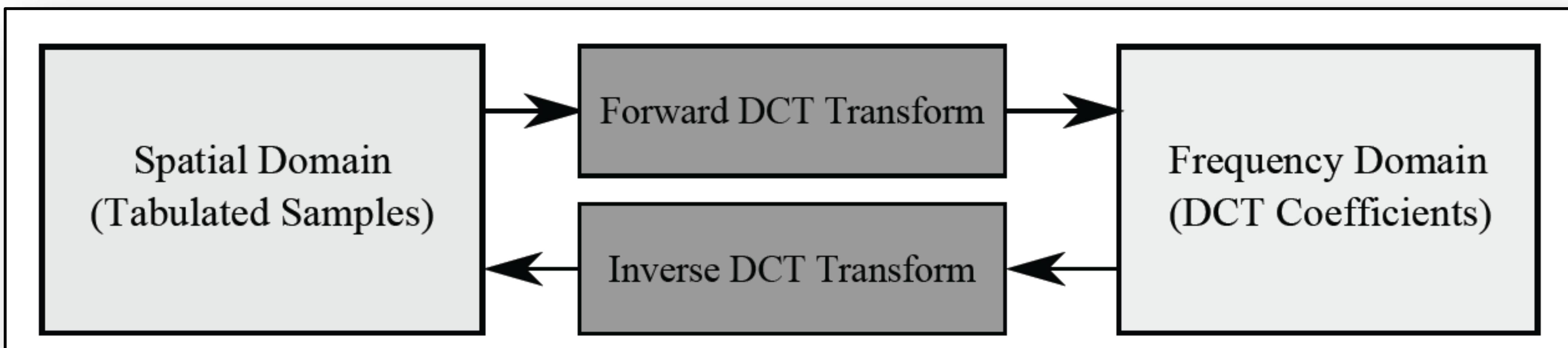
Examples of Frequency Spectra



Discrete Cosine Transform (DCT)

How to determine the Amplitudes?

- The DCT transformation is a way of decomposing a spatial image into sinusoidal basis images.



- The forward DCT goes from spatial to frequency. The inverse DCT goes from frequency to spatial.

- The DCT is invertible – no loss of information either way

DCT (one-dimensional)

- Discrete cosine transform

$$
\begin{aligned}
C(u) &= \alpha(u) \sum_{x=0}^{N-1} f(x) \cos\left(\frac{(2x+1)u\pi}{2N} \right) \\
\alpha(k) &= \left\{ \begin{array}{ll} \sqrt{\frac{1}{N}} & \text{if } k = 0 \\ \sqrt{\frac{2}{N}} & \text{otherwise} \end{array} \right\}.
\end{aligned}
$$

- The strength of the ‘u’ sinusoid is given by C(u)
  - Project f onto the basis function
  - All samples of f contribute the coefficient
  - C(0) is the zero-frequency component – the average value!

DCT

- Consider a digital image such that one row has the following samples

x	0	1	2	3	4	5	6	7
f(x)	20	12	18	56	83	10	104	114

- There are 8 samples so N=8
- u is in [0, N-1] or [0, 7]
- Must compute 8 DCT coefficients: C(0), C(1), …, C(7)
- Start with C(0)

$$C(0) = \sqrt{\frac{1}{N}} \sum_{x}^{N-1} f(x)$$

DCT: Computing C(0)

$$
\begin{aligned}
C(0) &= \sqrt{\tfrac{1}{8}}\sum_{x=0}^{7} f(x)\cos\left(\frac{(2x+1)\cdot 0\pi}{2\cdot 8}\right) \\
&= \sqrt{\tfrac{1}{8}}\sum_{x=0}^{7} f(x)\cos(0) \\
&= \sqrt{\tfrac{1}{8}}\cdot\{f(0)+f(1)+f(2)+f(3)+f(4)+f(5)+f(6)+f(7)\} \\
&= .35\cdot\{20+12+18+56+83+110+104+115\} \\
&= 183.14
\end{aligned}
$$

DCT: Computing the Remaining Coefficients

Spatial Domain

f(0)	f(1)	f(2)	f(3)	f(4)	f(5)	f(6)	f(7)
20	12	18	56	83	110	104	114

Frequency Domain

$C(0)$	$C(1)$	$C(2)$	$C(3)$	$C(4)$	$C(5)$	$C(6)$	$C(7)$
183.1	-113.0	-4.2	22.1	10.6	-1.5	4.8	-8.7

DCT Implementation (Brute Force)

- Since the DCT coefficients are reals, stored with an array of floats
- This approach is O(?)

```
public static float[] forwardDCT(float[] data) {
  final float alpha0 = (float) Math.sqrt(1.0 / data.length);
  final float alphaN = (float) Math.sqrt(2.0 / data.length);
  float[] result = new float[data.length];

  for (int u = 0; u < result.length; u++) {
    for (int x = 0; x < data.length; x++) {
      result[u] += data[x]*(float)Math.cos((2*x+1)*u*Math.PI/(2*data.length));
    }
    result[u] *= (u == 0 ? alpha0 : alphaN);
  }
  return result;
}
```

DCT (two-dimensional)

- The 2D DCT is given below where the definition for alpha is the same as before

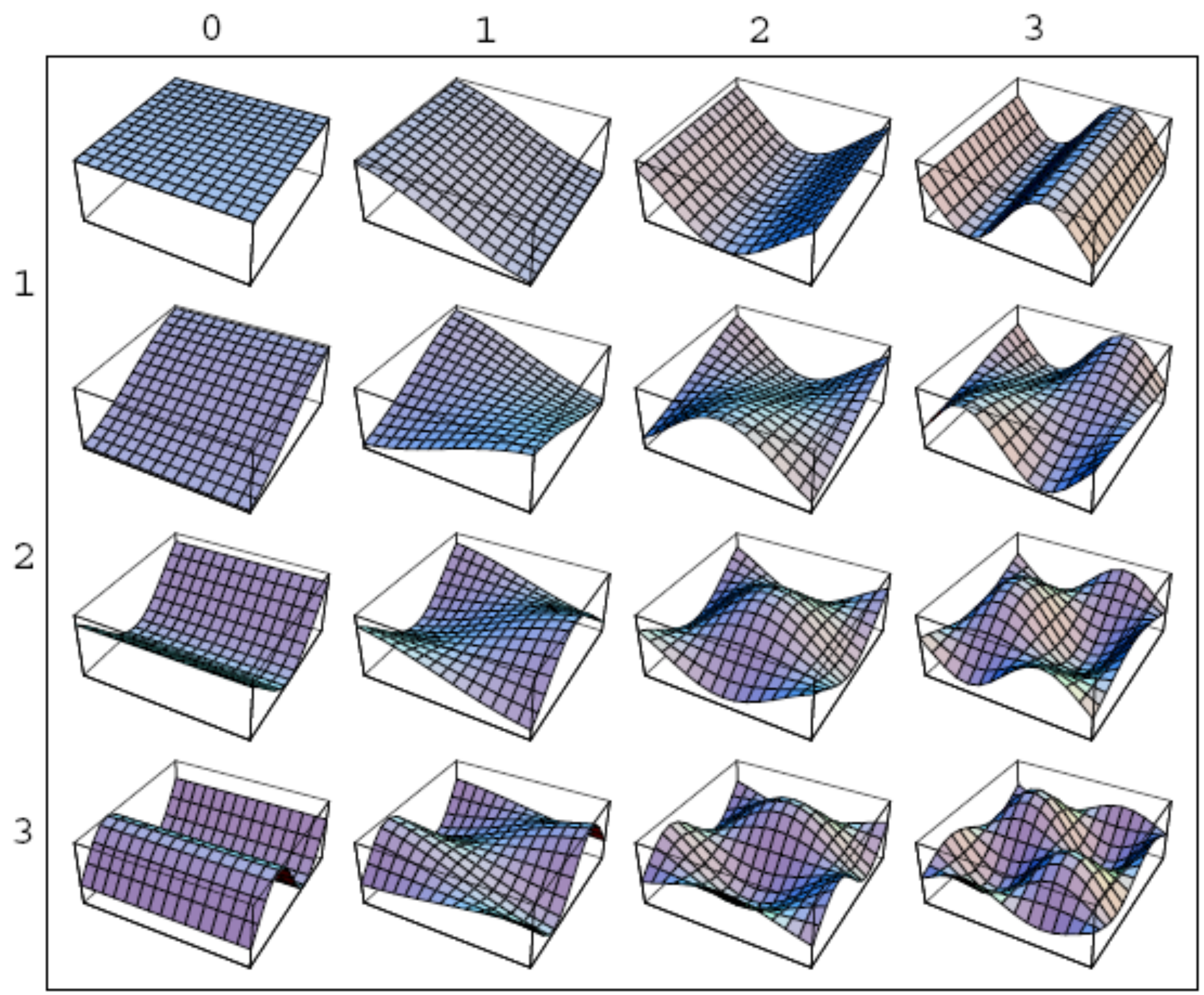
$$C(u,v) = \alpha(u)\alpha(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos\left(\frac{(2x+1)u\pi}{2N}\right)\cos\left(\frac{(2y+1)v\pi}{2N}\right)$$

- For an MxN image there are MxN coefficients

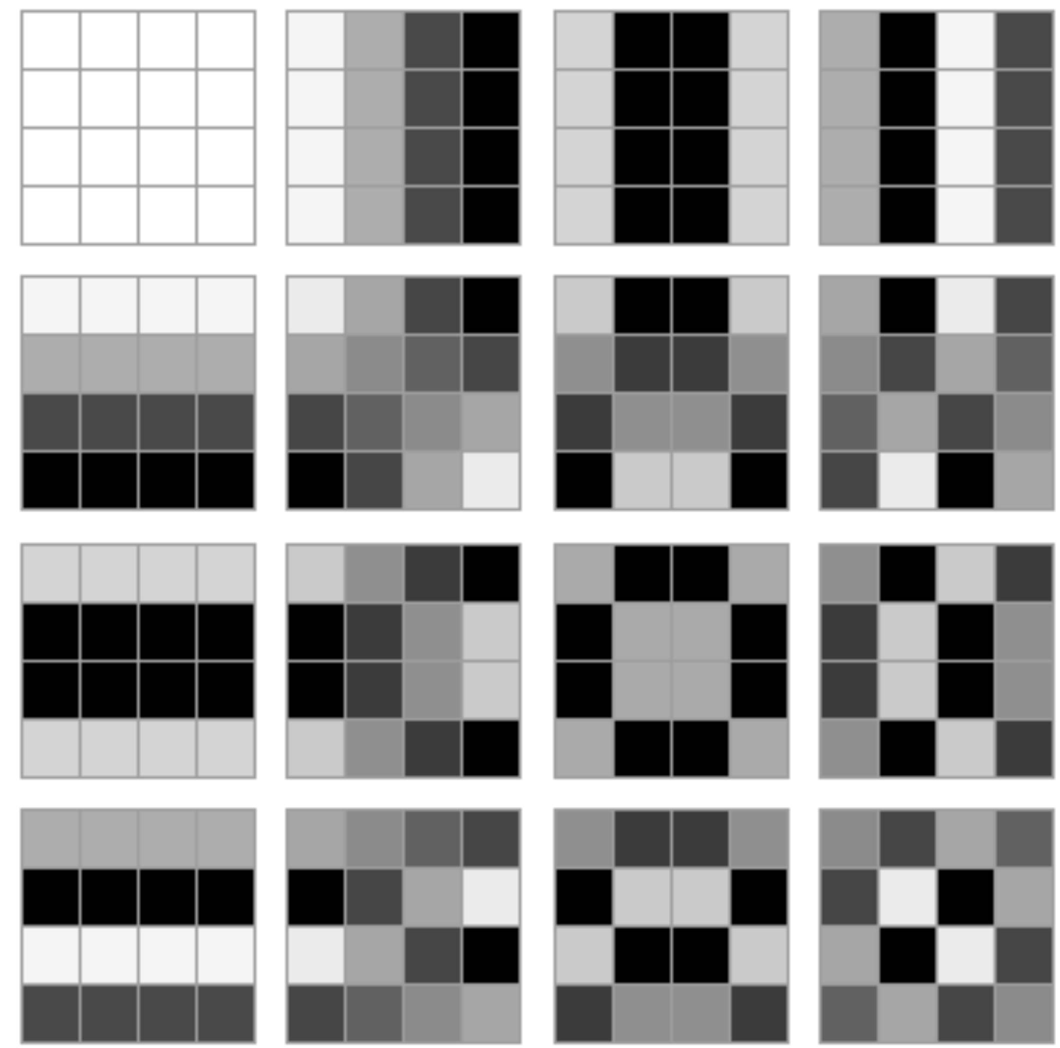
- Each image sample contributes to each coefficient

- Each (u,v) pair corresponds to a ‘pattern’ or ‘basis function’

DCT Basis Functions



Basis Functions



Imaged

The DCT is Invertible

- Spatial samples can be recovered from the DCT coefficients

$$f(x) = \sum_{u=0}^{N-1} \alpha(u) C(u) \cos\left(\frac{(2x+1)u\pi}{2N}\right)$$

$$f(x,y) = \sum_{u=0}^{N-1} \sum_{v}^{N-1} \alpha(u)\alpha(v) C(u,v) \cos\left(\frac{(2x+1)u\pi}{2N}\right) \cos\left(\frac{(2y+1)v\pi}{2N}\right)$$

Frequency Domain

- The central idea in frequency domain representation is to
  - Find a set of orthogonal sinusoidal patterns that can be combined to form any image
  - Any image can be expressed as the weighted sum of these basis images.
  - The DCT, as well as the discrete Fourier transform, are ways of decomposing an image into the sum of sinusoidal basis images

Lec23 Required Reading

- Hunt, Ch. 9.4
- House, Ch. 14