

IN4MATX 241: Ubiquitous Computing

**Class 3:
Signal Processing &
Activity Recognition**

Daniel Epstein

Announcements

- Thanks for filling out the discussion leading preferences
- Project: It seems like some of you have similar interests (health), consider talking to one another!

Disclaimer

- This is not my area of expertise
- But it underlies many aspects of Ubicomp,
so I thought it was important to cover
- If you have expertise, feel free to correct me or add supplemental thoughts
- If you don't, the further along we go, the closer to the “human” we will get
- Many thanks to Jon Froehlich (U Washington) & Alex Mariakakis (U Toronto)
for materials

<https://jonfroehlich.github.io/>

<https://mariakakis.github.io/>

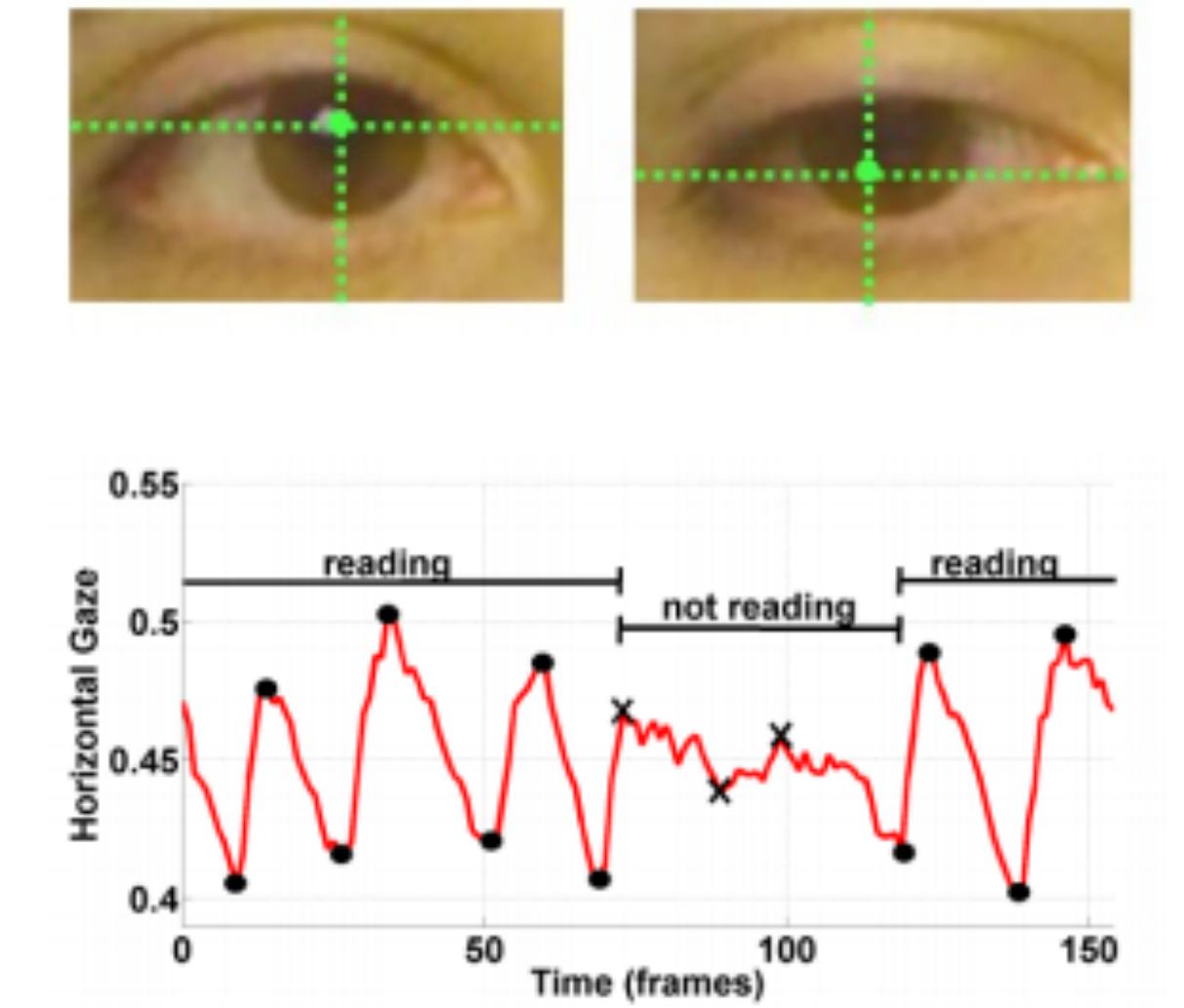
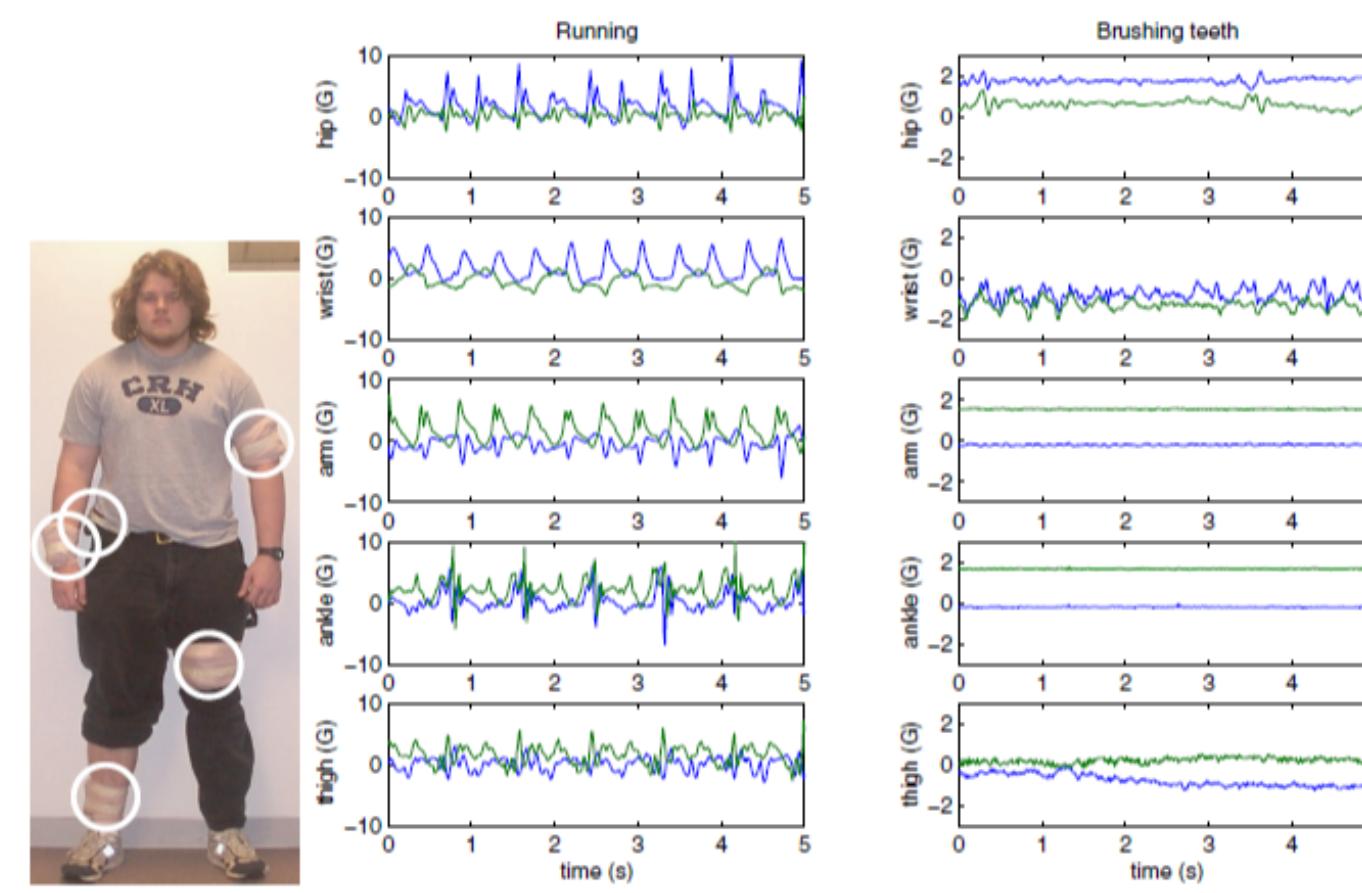
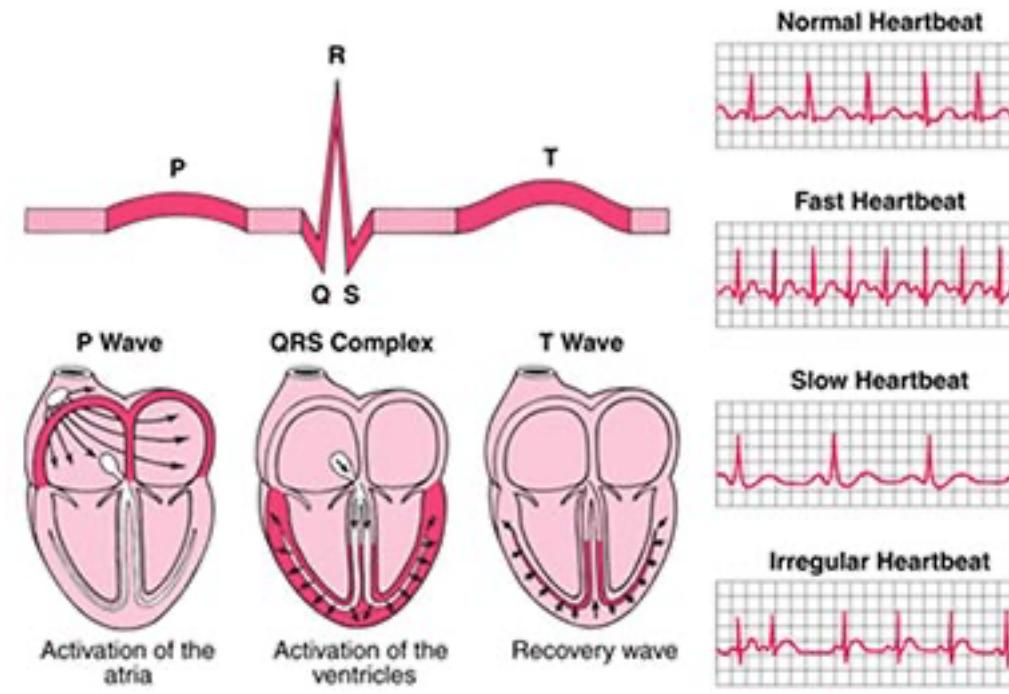
Learning goals

- Understand how sensor data is acquired
- Describe and represent a time-series signal
- Develop strategies for analyzing and processing signals to recognize activities

What are sensors sensing?

What are sensors sensing?
**Physical phenomena, digitized,
and represented as time series arrays**

Signals



Electrocardiograms
for heartbeat tracking

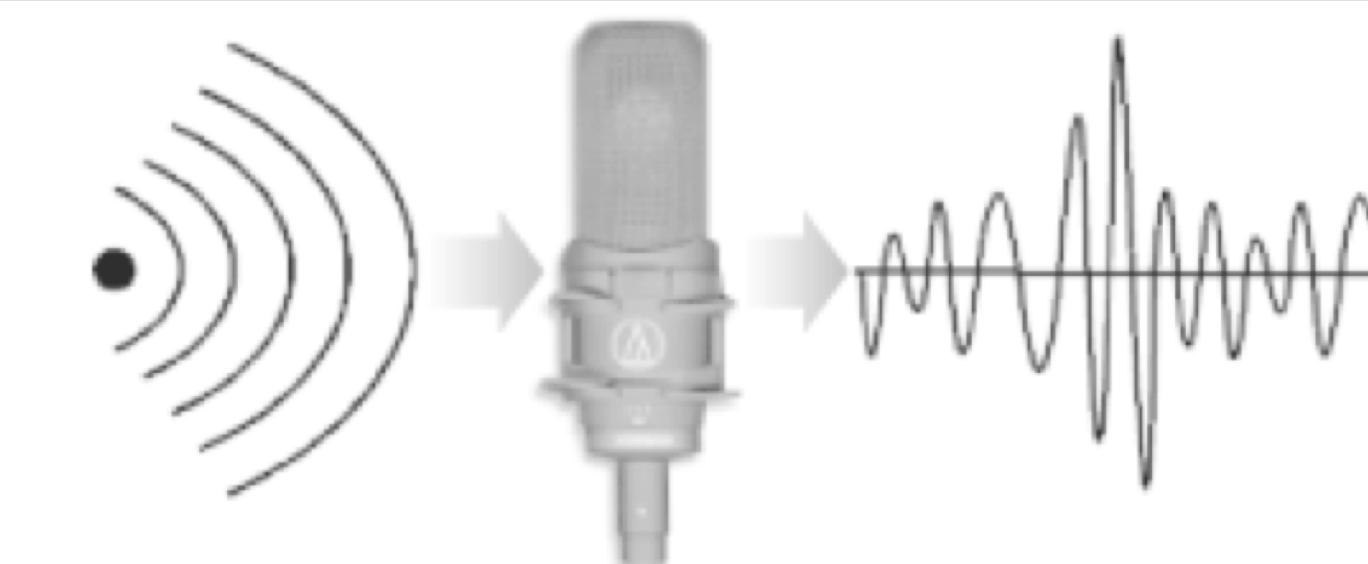
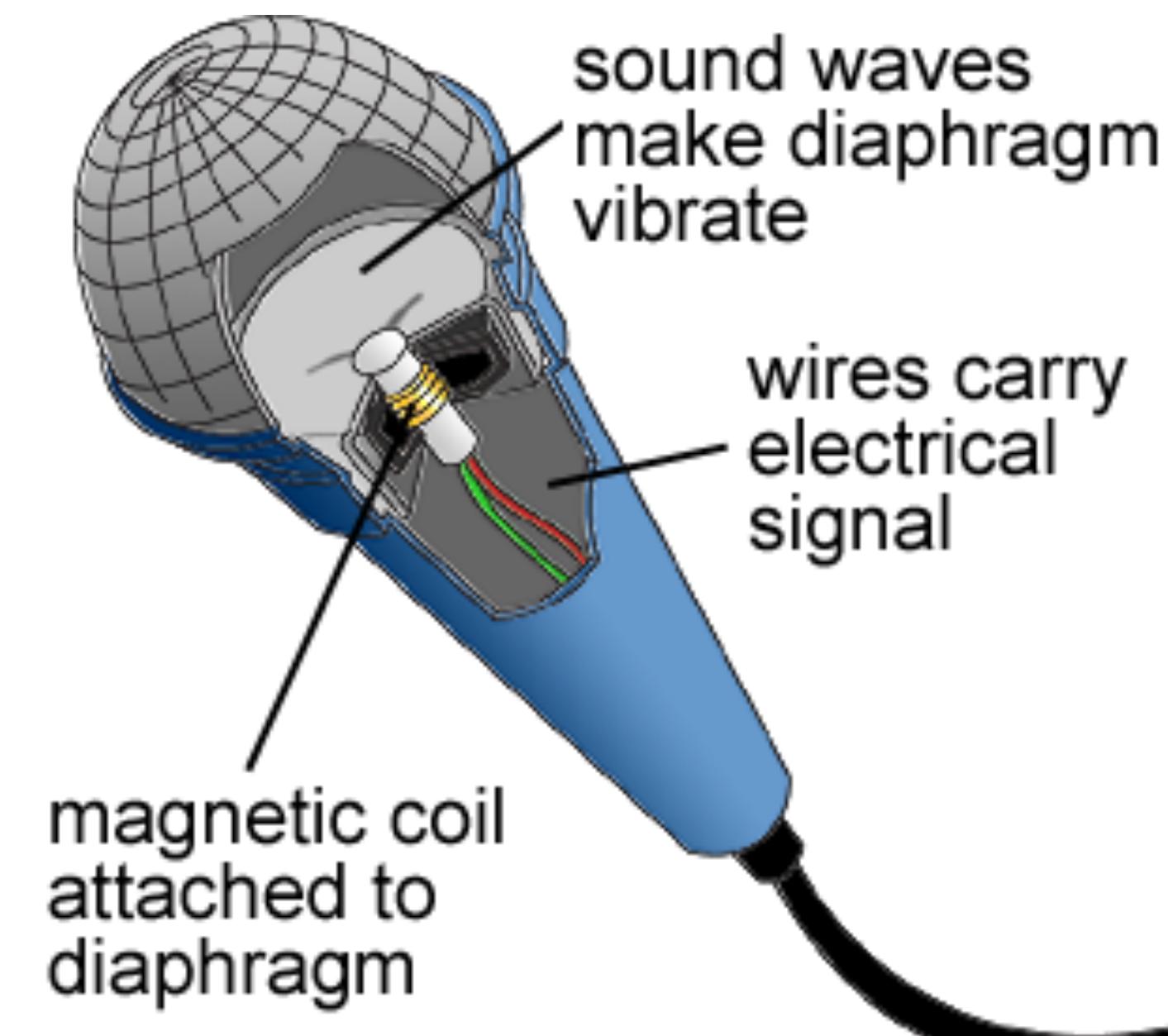
Accelerometers for
activity recognition

Smartphone camera
for gaze tracking

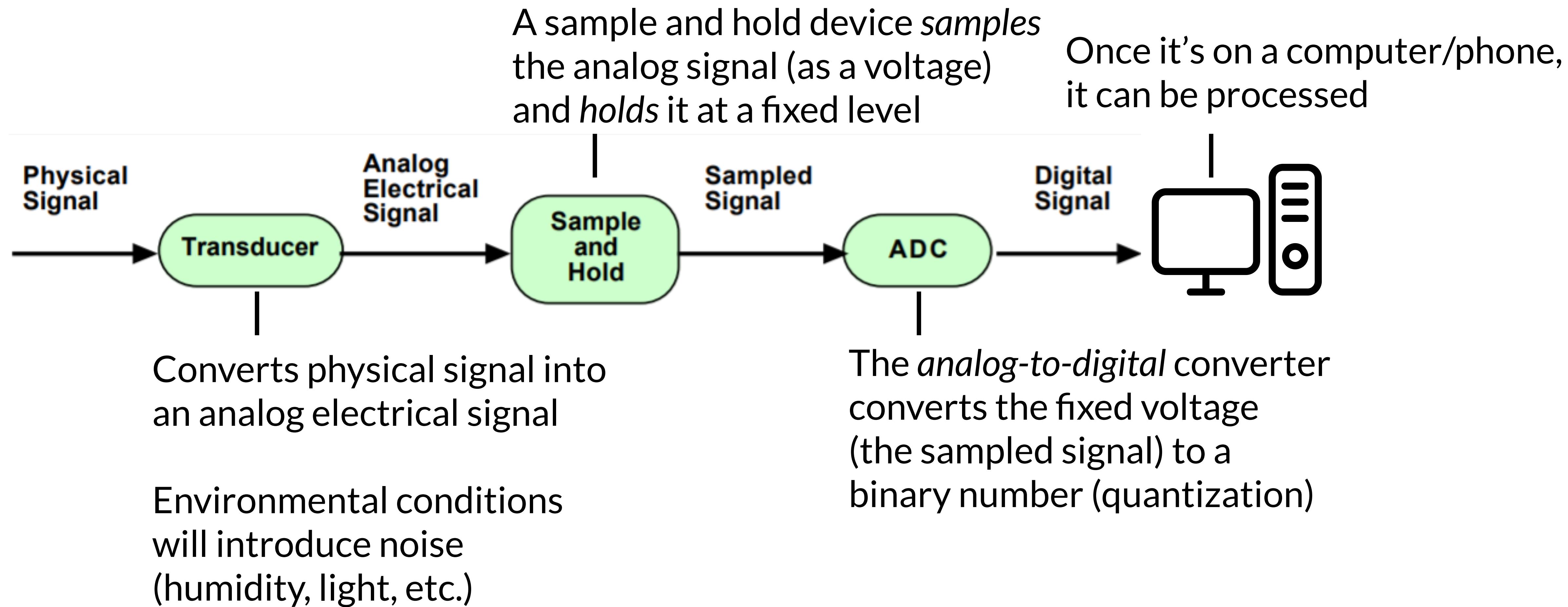
Signals

Example: a microphone

- **Physical stimulus:** when you speak, play an instrument, etc., it vibrates air in a sinusoidal pattern
- **Diaphragm vibration:** Inside a microphone is a small diaphragm that vibrates in response to air movement
- **Electricity generation:** A conductive coil attached to the diaphragm, moves back and forth across a magnet. This generates a current proportional to the sound wave



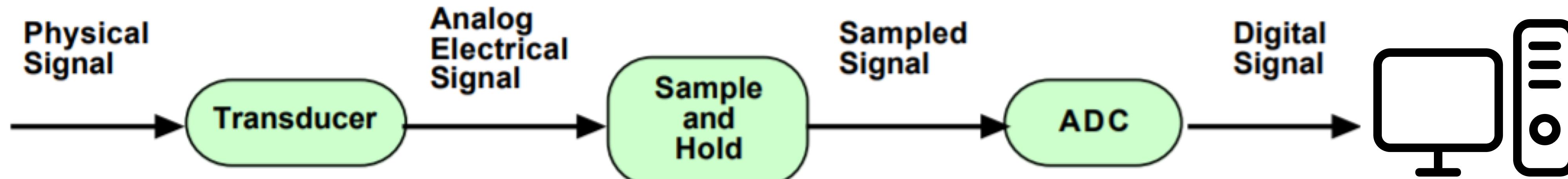
Signals



Signals

Signal acquisition: two key questions

- Sampling rate: how often do I need to sample the “real world”?
- Quantization: How many bits should I use to represent each sample?

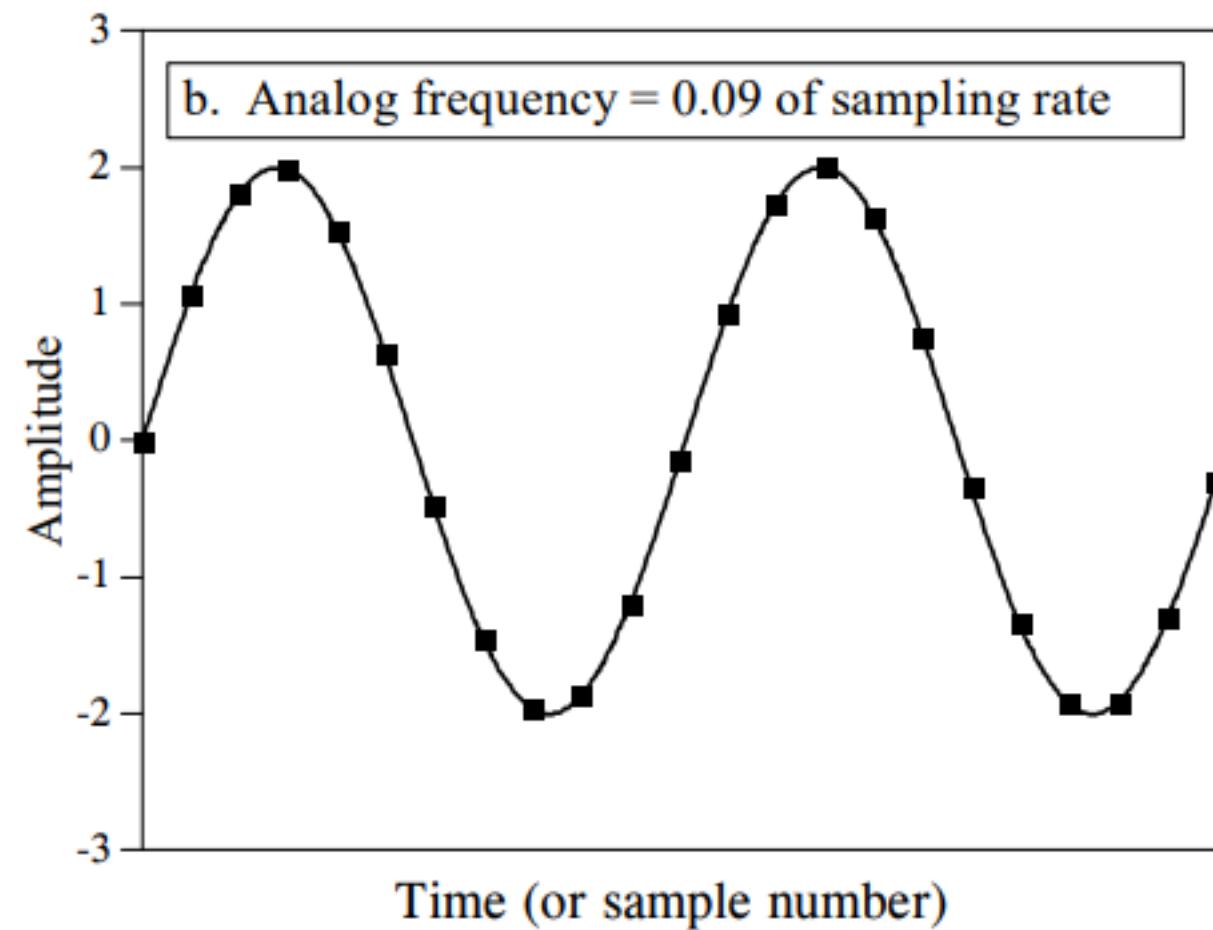


Signals

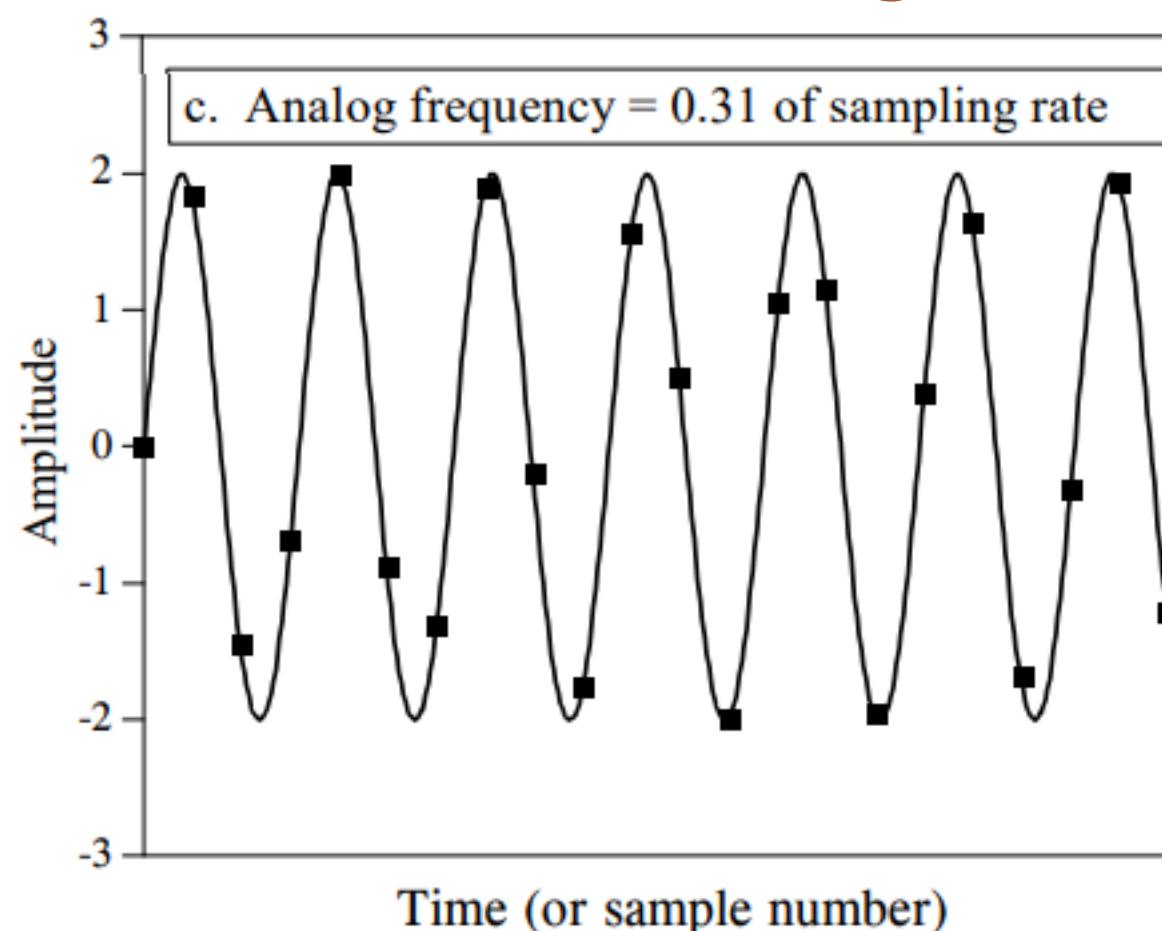
Sampling rate

- You should sample at a rate sufficient to accurately reconstruct the real analog signal

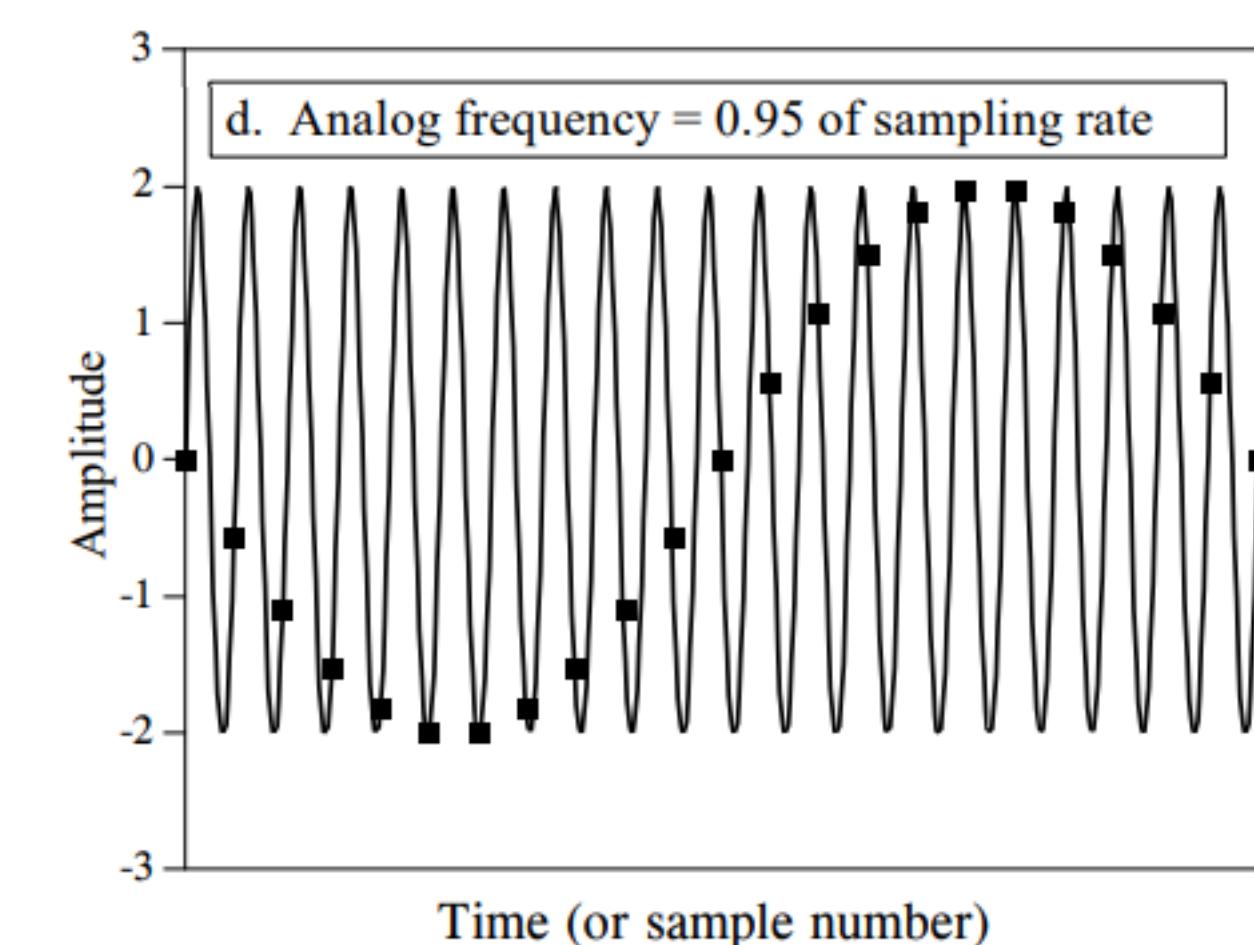
Solid line is the “real world” analog signal, each dot represents a digital sample



This “looks bad”, but there’s enough information (mathematically) to reconstruct the signal



The sampled data now looks like a different signal! This is called *aliasing*.

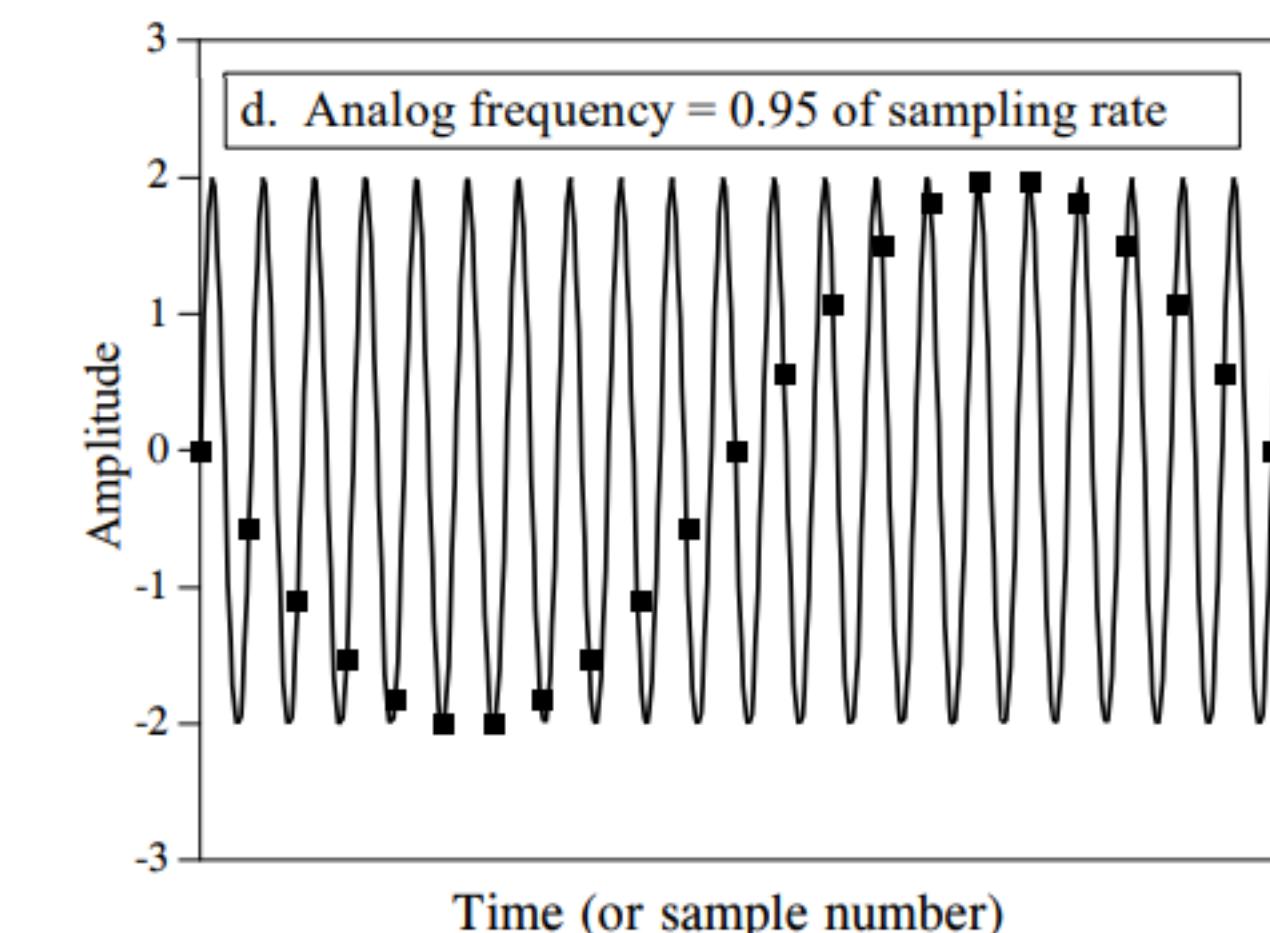


Signals

Sampling rate

- The sampling theorem states that a continuous signal can only be properly sampled if it does not contain frequency components above one-half of the sampling rate
- Sampling rate must be $> 2 * \text{max}(\text{real world signal frequency})$

The sampled data now looks like a different signal! This is called *aliasing*.



Signals



<https://www.youtube.com/watch?v=AYQAKwCxScc>

Signals

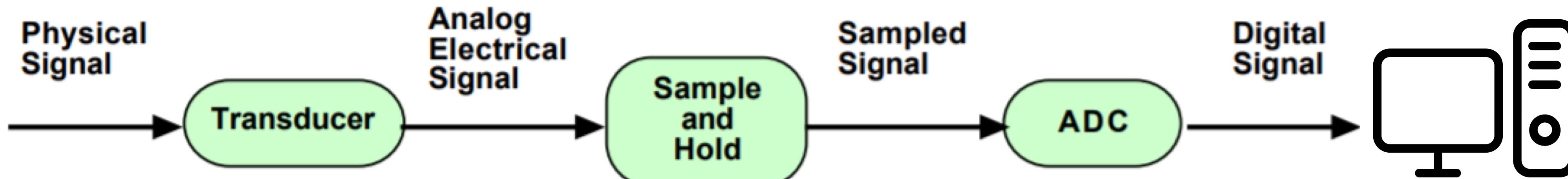


<https://www.youtube.com/watch?v=AYQAKwCxScc>

Signals

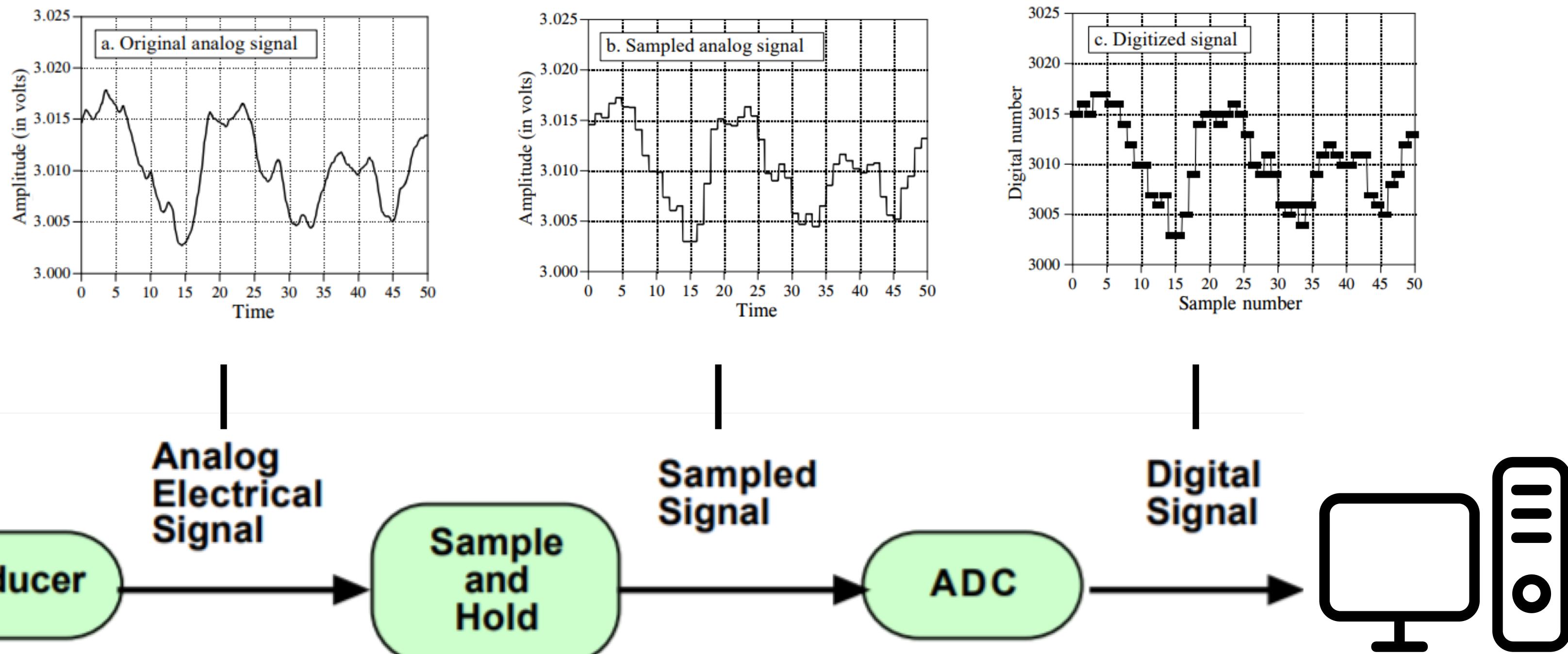
Signal acquisition: two key questions

- Sampling rate: how often do I need to sample the “real world”?
- Quantization: How many bits should I use to represent each sample?



Signals

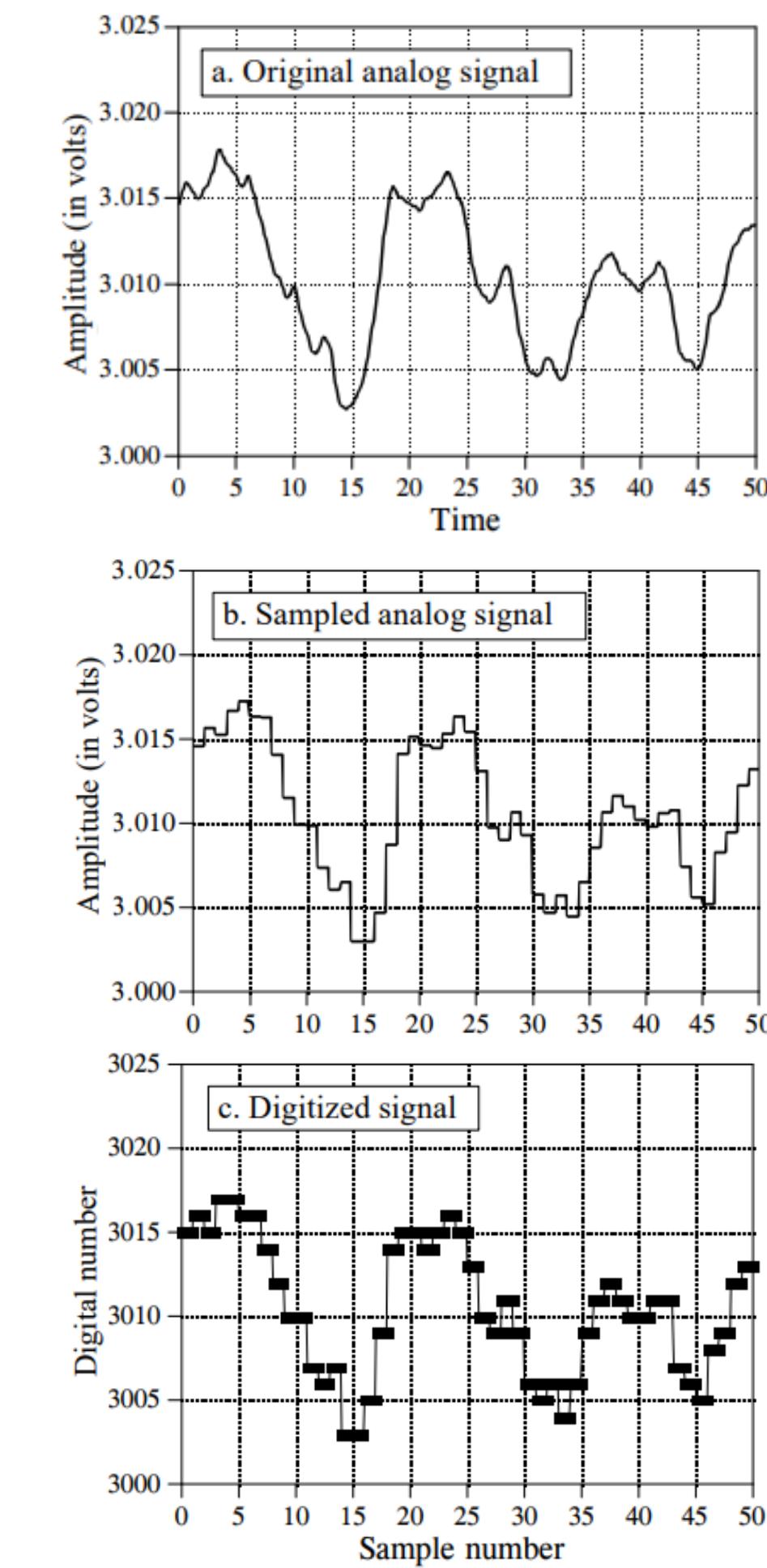
Quantization



Signals

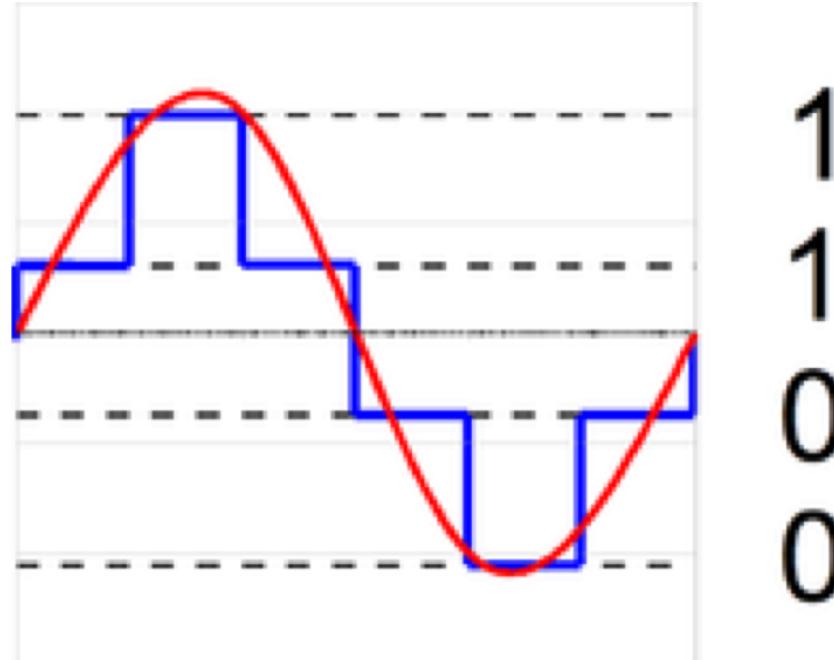
Quantization

- Why must we quantize signals?
 - Input is typically continuous, thus not possible to represent digitally
 - If we're sampling with high frequency, there's a *lot* of data to store
 - Quantization is the basis for basically all compression algorithms

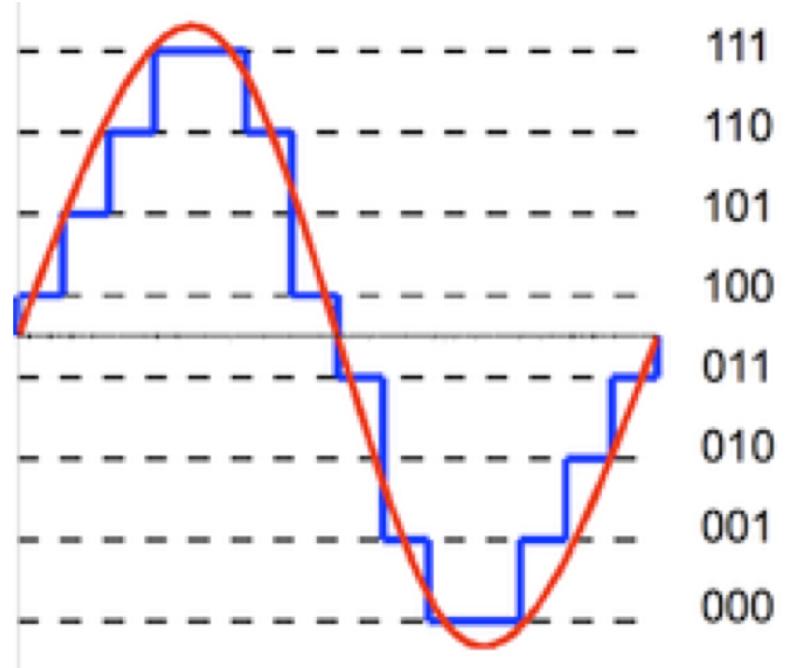


Signals

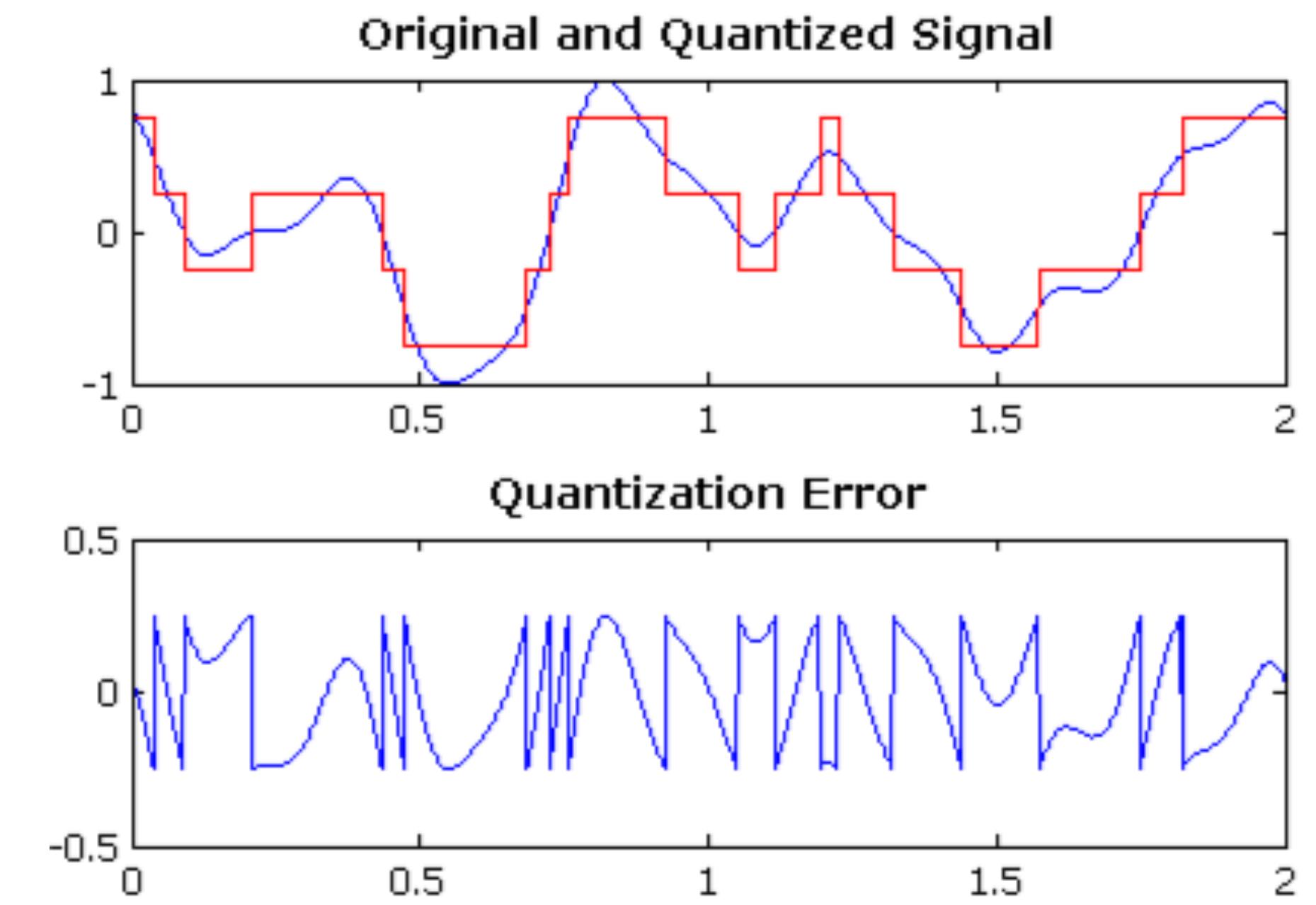
Quantization



2-bit resolution allows for four discretization levels of an analog signal

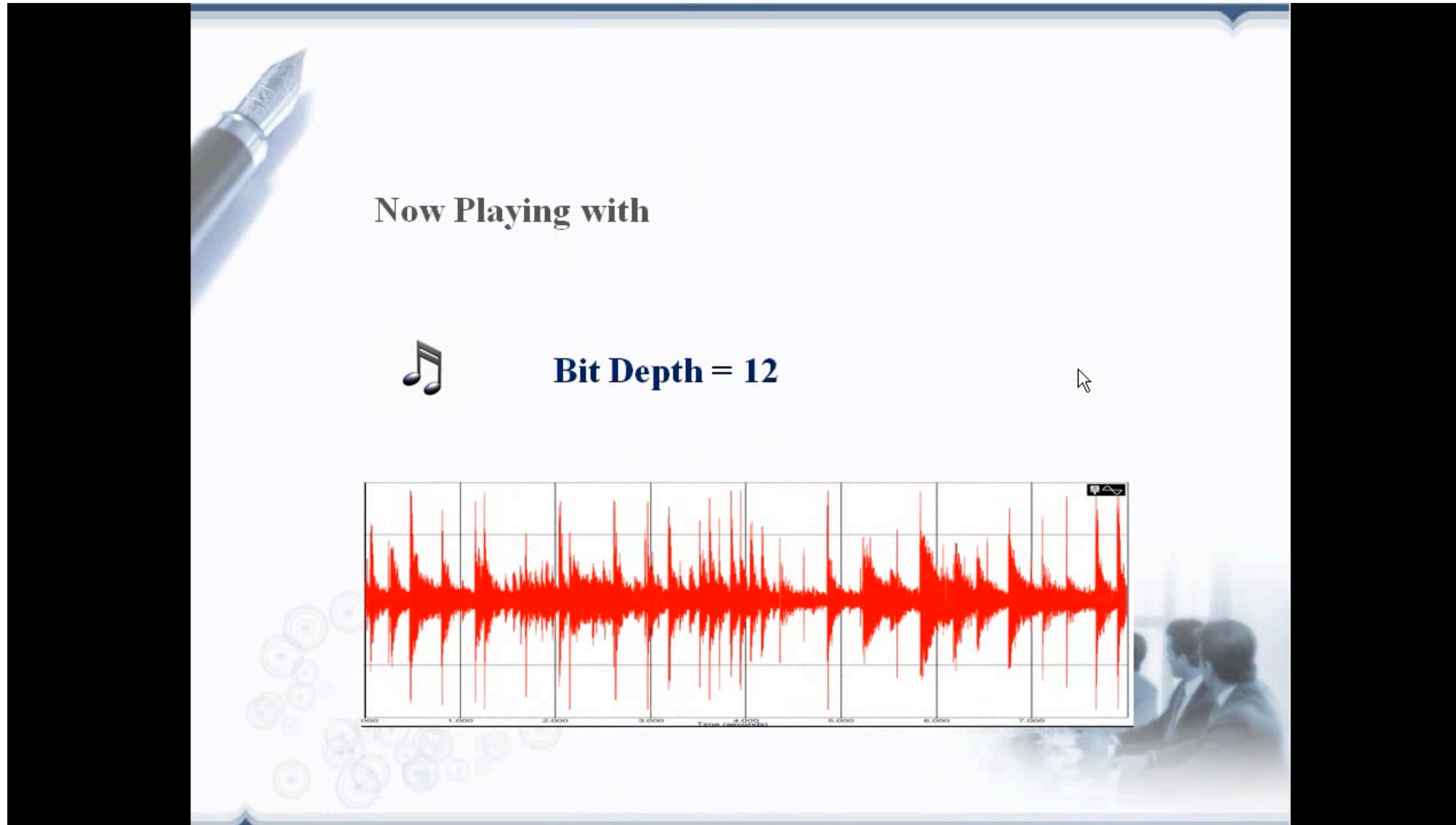


3-bit resolution allows for eight discretization levels of an analog signal

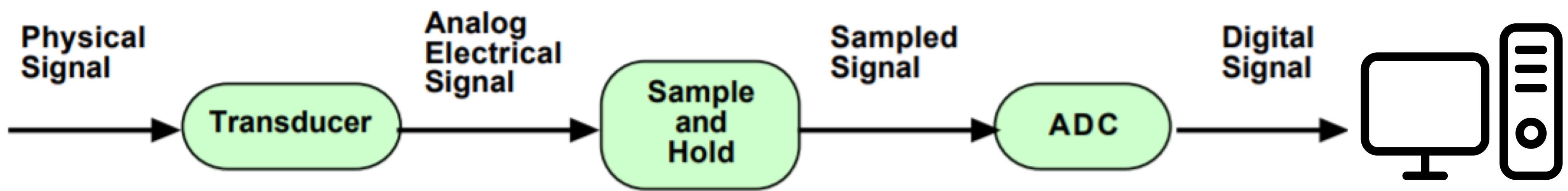


The quantization error is the difference between an input value and its quantized value

Signals



https://www.youtube.com/watch?v=_cRFBBnUFug

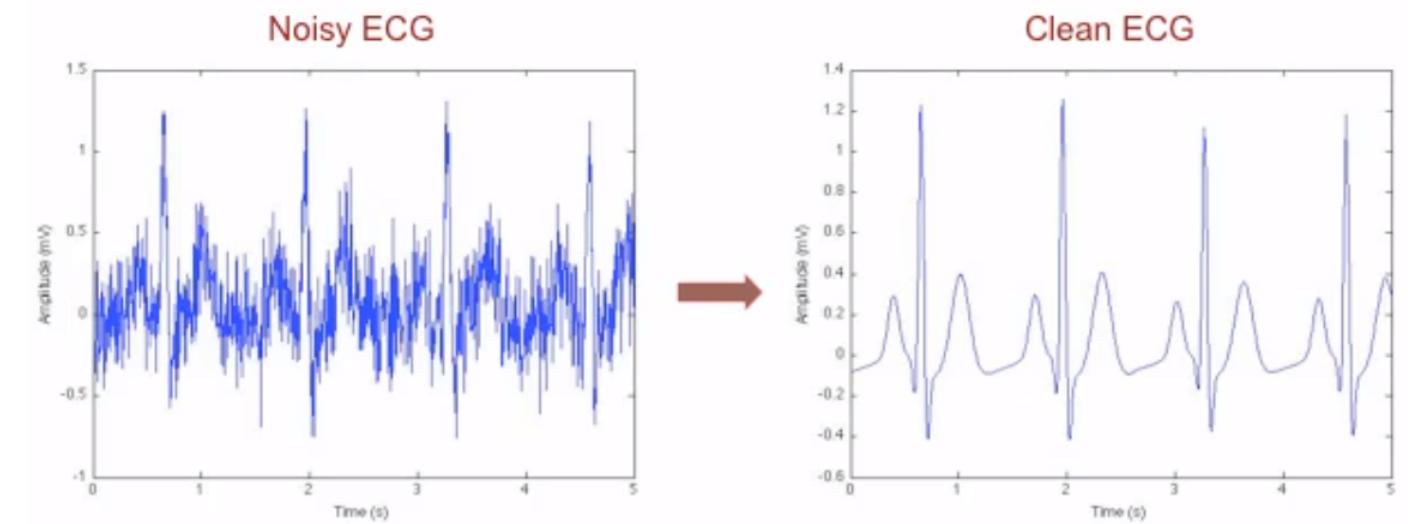


What is *signal processing*?

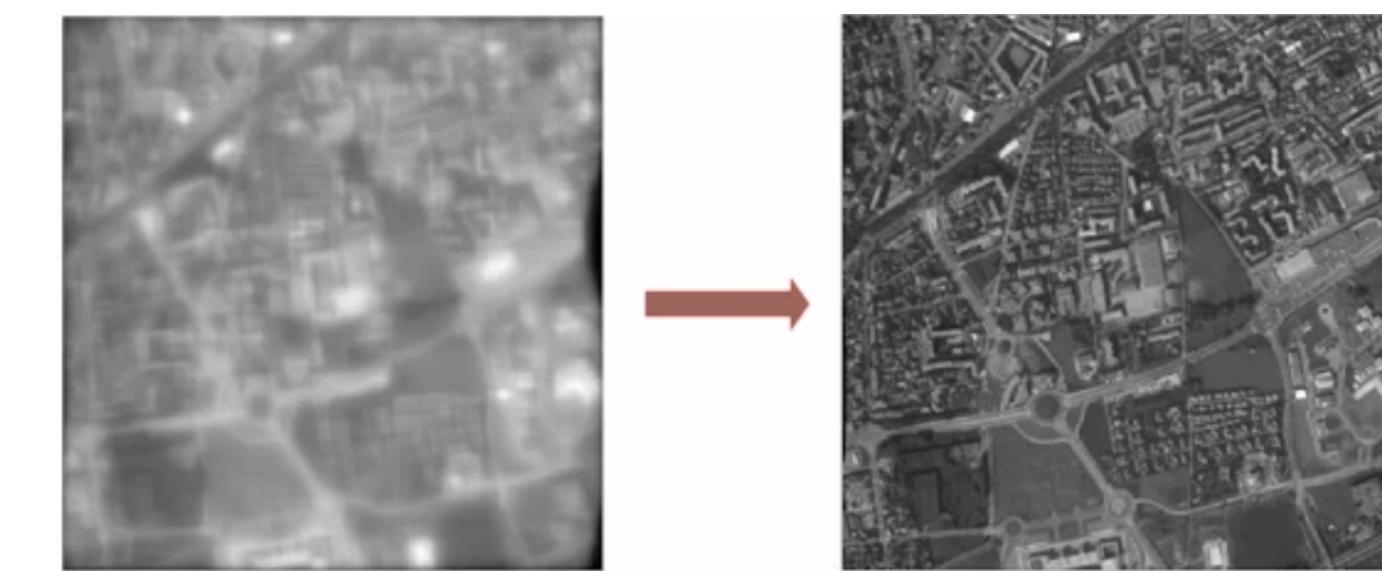
What is *signal processing*?

**Manipulating a signal to change its
characteristics or extract information**

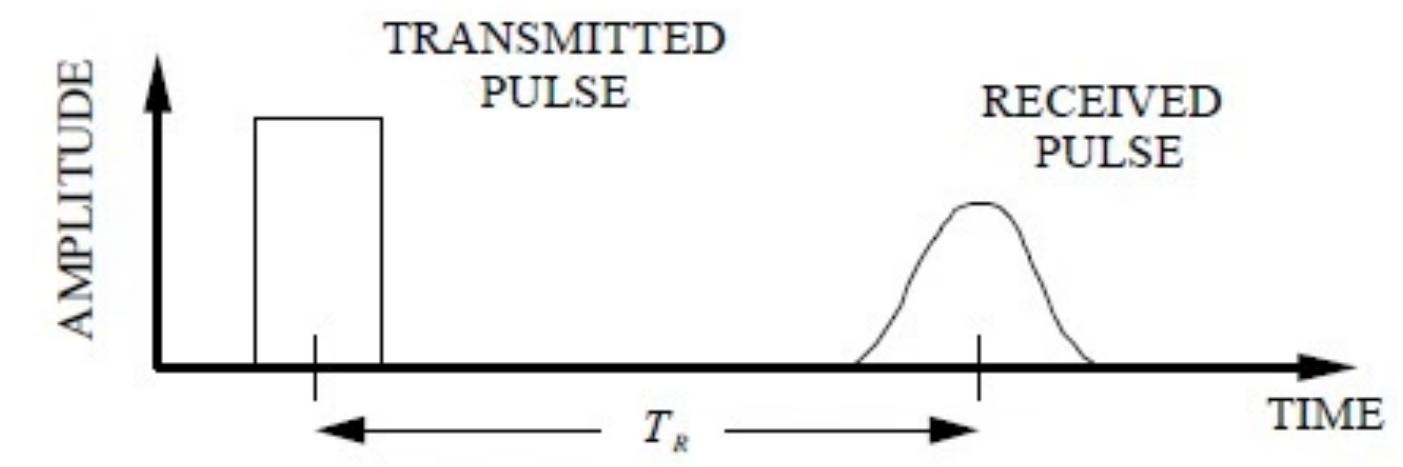
Signal processing



Eliminating noise
(e.g., smoothing ECG)



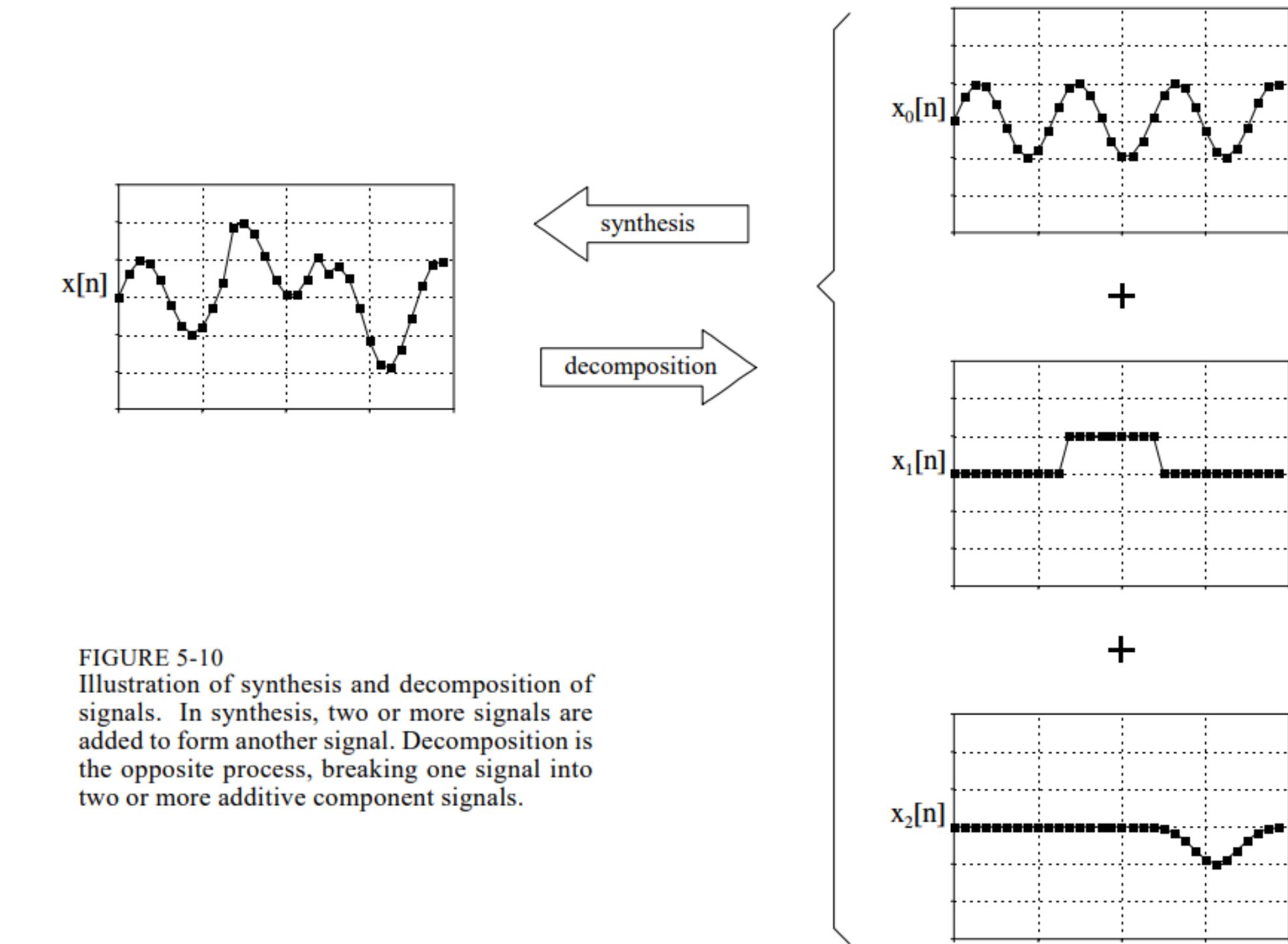
Correcting distortion
(e.g., Hubble lens)



Extracting information
(e.g., distance & velocity
from radar pulse)

Signal processing

- With a lot of math, you can break a signal down into component signals, which can be useful for recognition
- But we'll stop here :-)



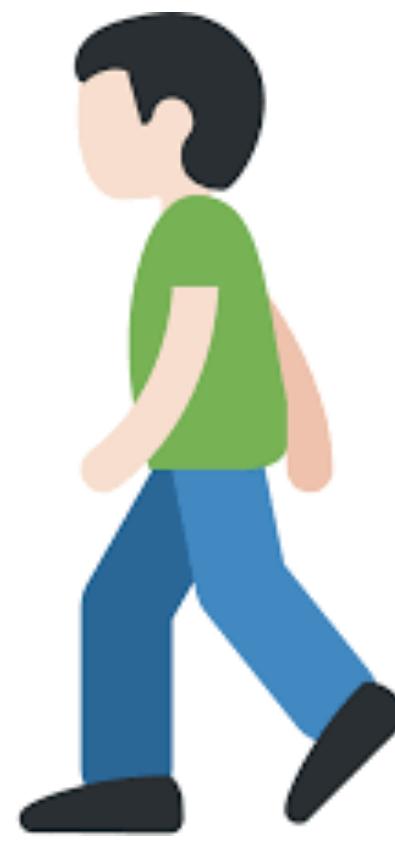
**How do we translate these
theoretical approaches
to the applied nature of Ubicomp?**

Signal Processing in Ubicomp

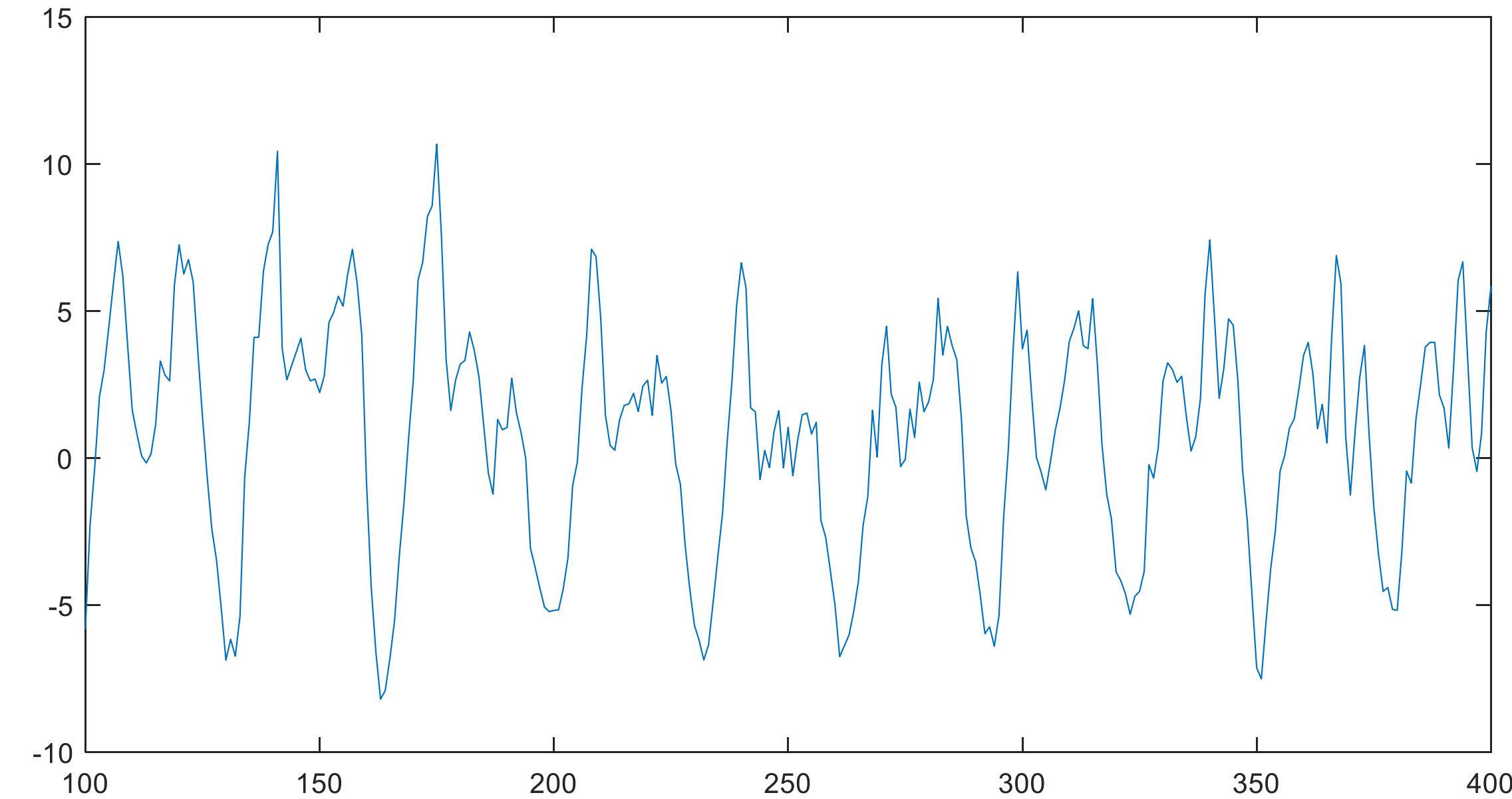
- Capture signals under controlled conditions with “ground truth” (e.g., walk ten steps, wait, walk ten steps, wait). Collect easy & hard samples.
- Use tools to visualize, study, & process the signal. Identify patterns, brainstorm potential approaches. Calculate descriptive stats (mean, median)
- Search for previous solutions to comparable problems (e.g., Google Scholar). What can be re-appropriated for your problem?
- Generate and test approaches offline using test data. Iterate.
- When offline works well, adapt approach to perform in real-time. If approach is not as effective as desired, collect more naturalistic samples & repeat

Example: Step Counting

Step counting



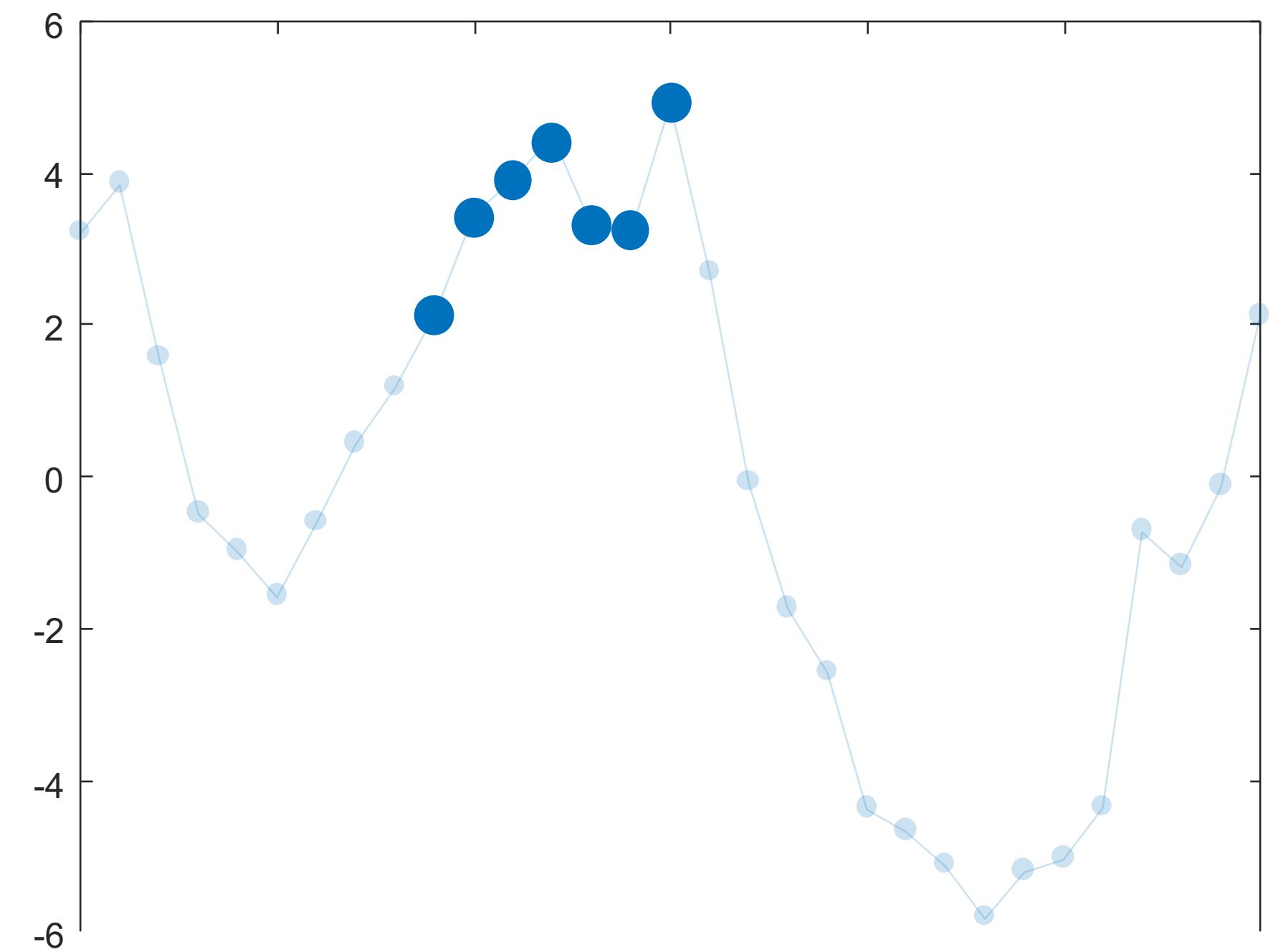
Acceleration signal



Step counting

Sliding Window

- As your signal gets longer, it does not make sense to analyze all of that data at once
 - Infeasible for memory/performance
 - Older information is usually less useful
- When a new value comes, add it to your list and get rid of the oldest
- How large of a window? It depends!

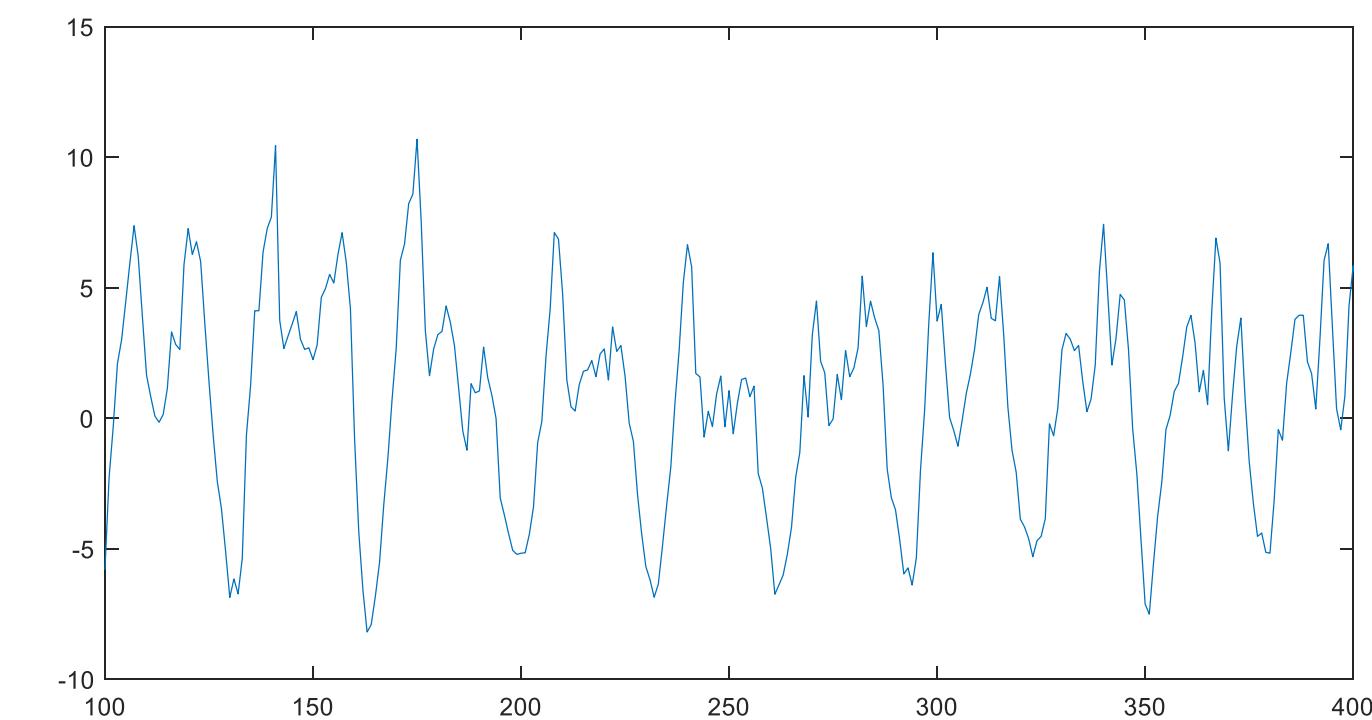


Window size: 5

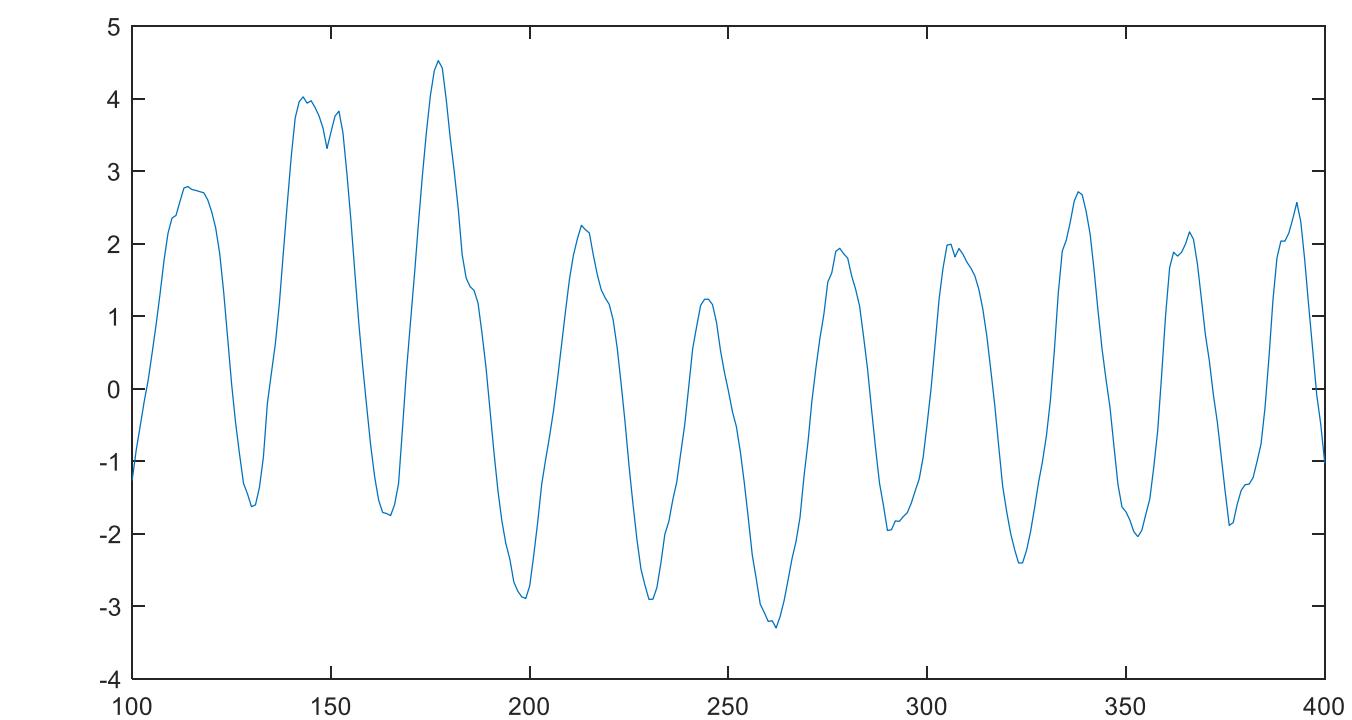
Step counting

Filtering

- As previously discussed, signals will almost never look as clean as you want them to be
- Sensors will have noise that break assumptions you make in your algorithm



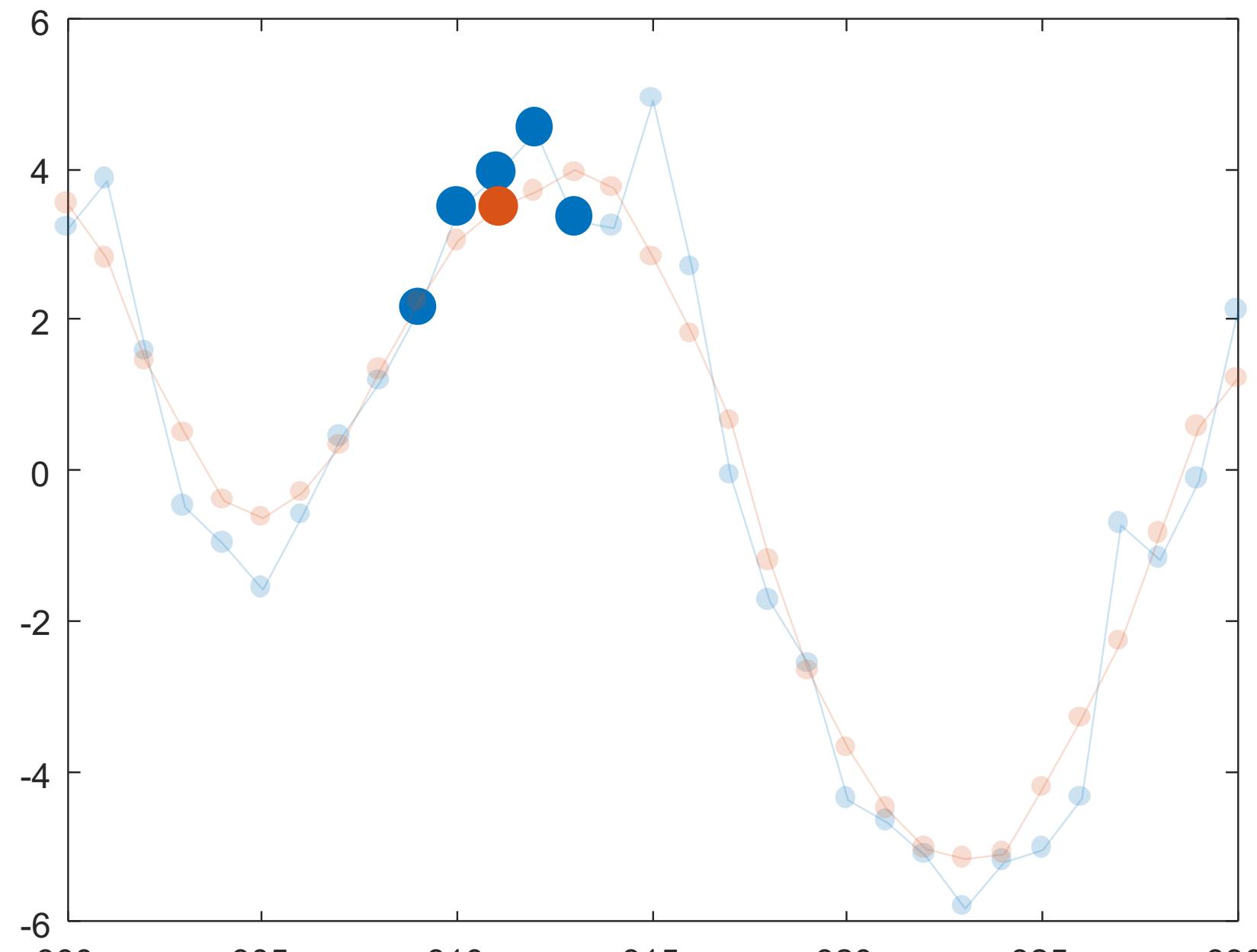
Unfiltered



Filtered

Step counting

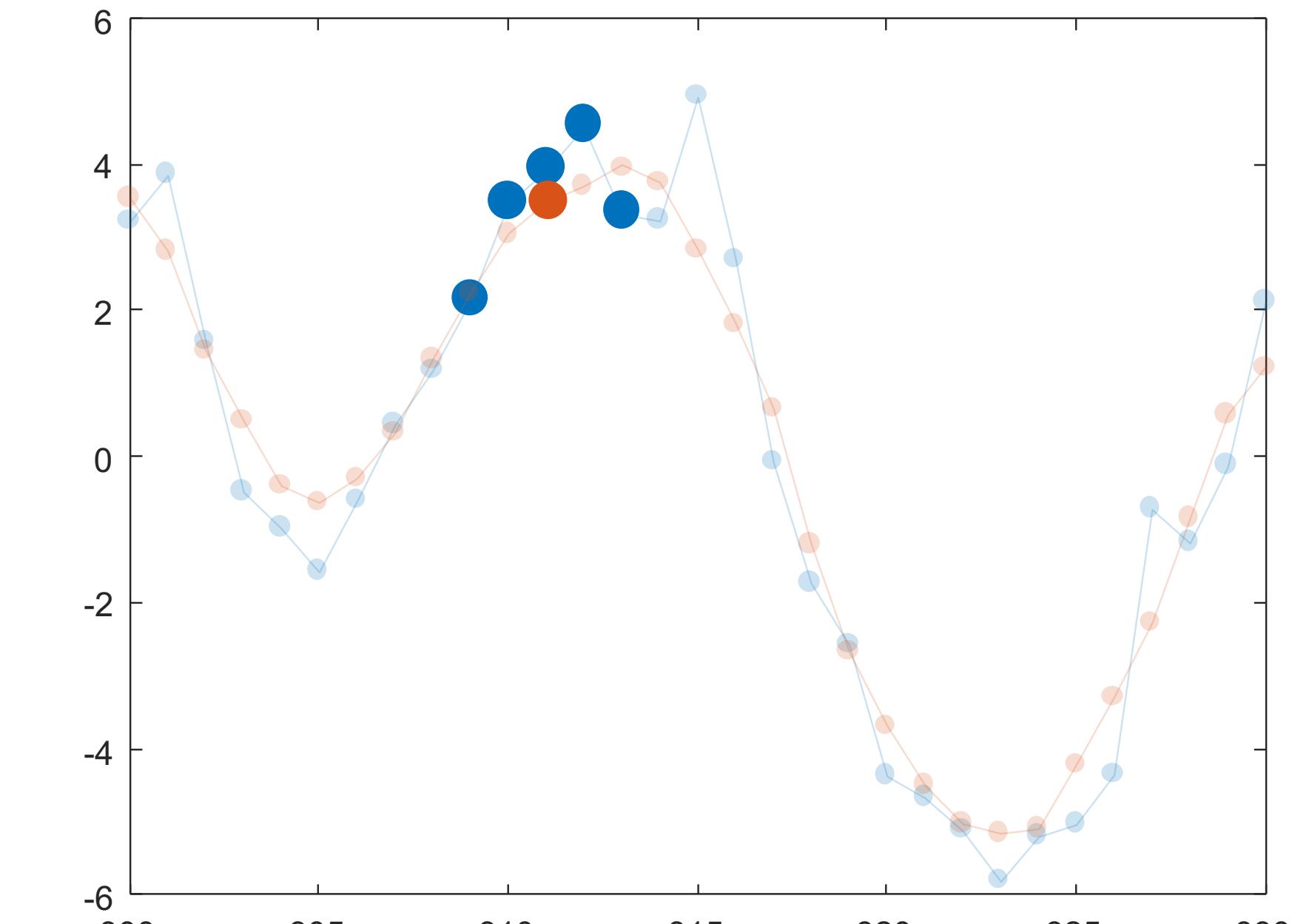
Filtering



Filter Length [n] = 5

Input = [2.4, 4, 4.5, 5, 3.5]

Output= $(2.4, 4, 4.5, 5, 3.5)/5 = 3.9$



Filter Length [n] = 5

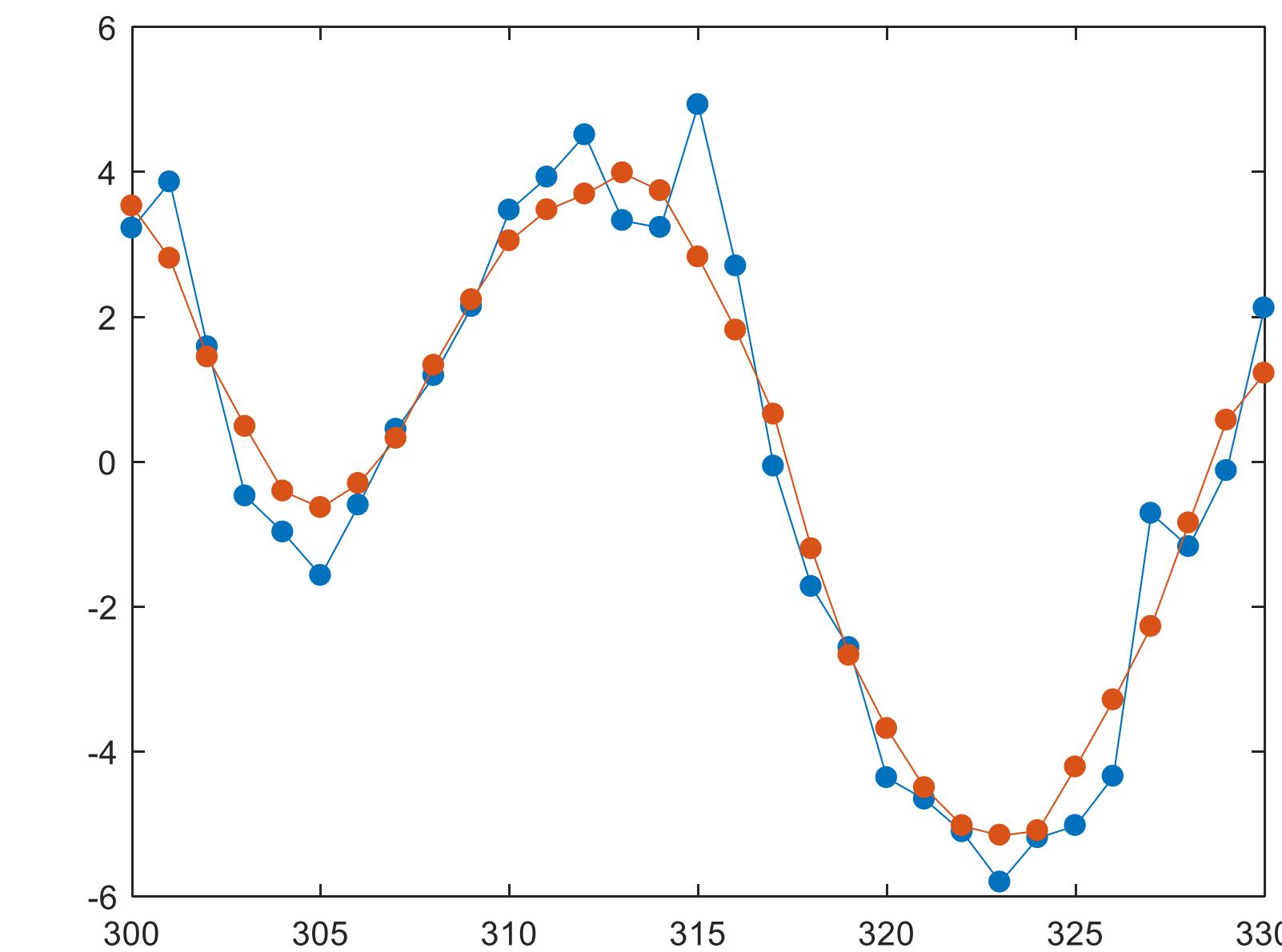
Input = [4, 4.5, 5, 3.5, 3.2]

Output= $(4, 4.5, 5, 3.5+3.2)/5 = 4.0$

Step counting

Filtering

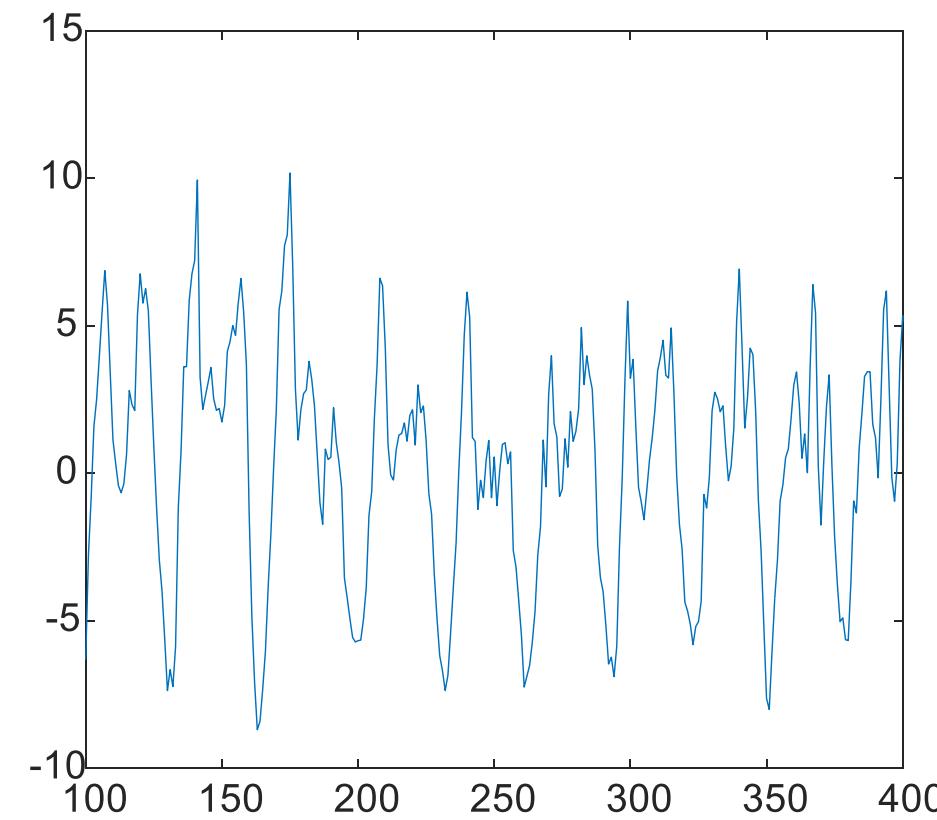
- Filter length $[N] = 5$
- Does not have future information
- *Delay of filter length*



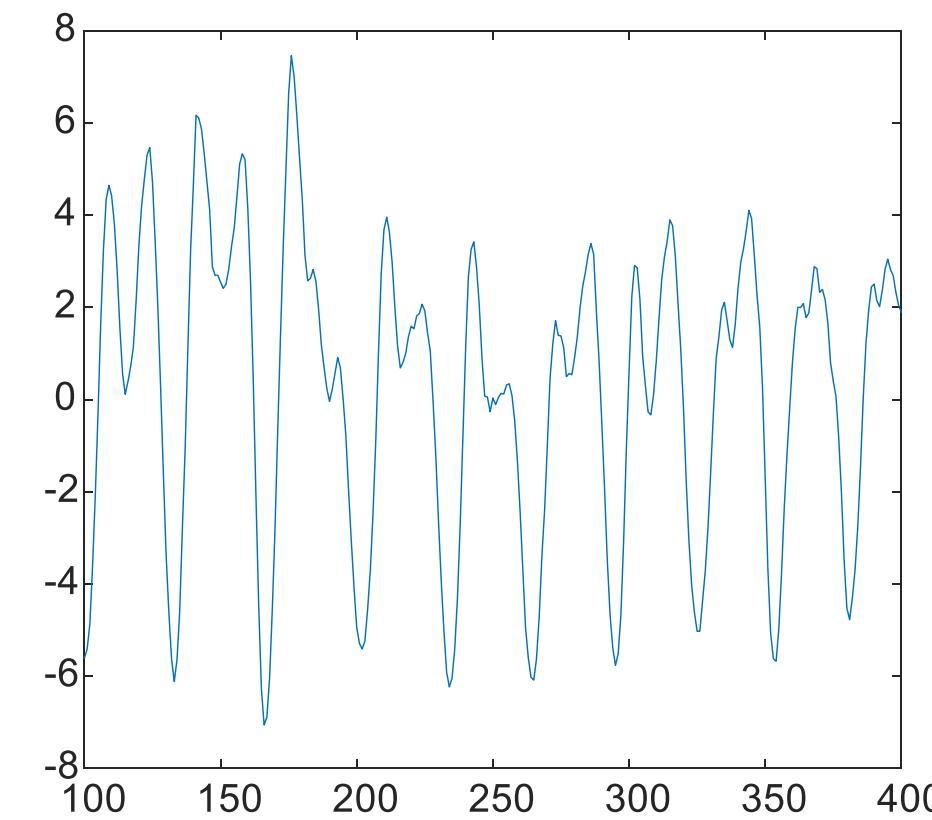
Noisy signal
Filtered signal

Step counting

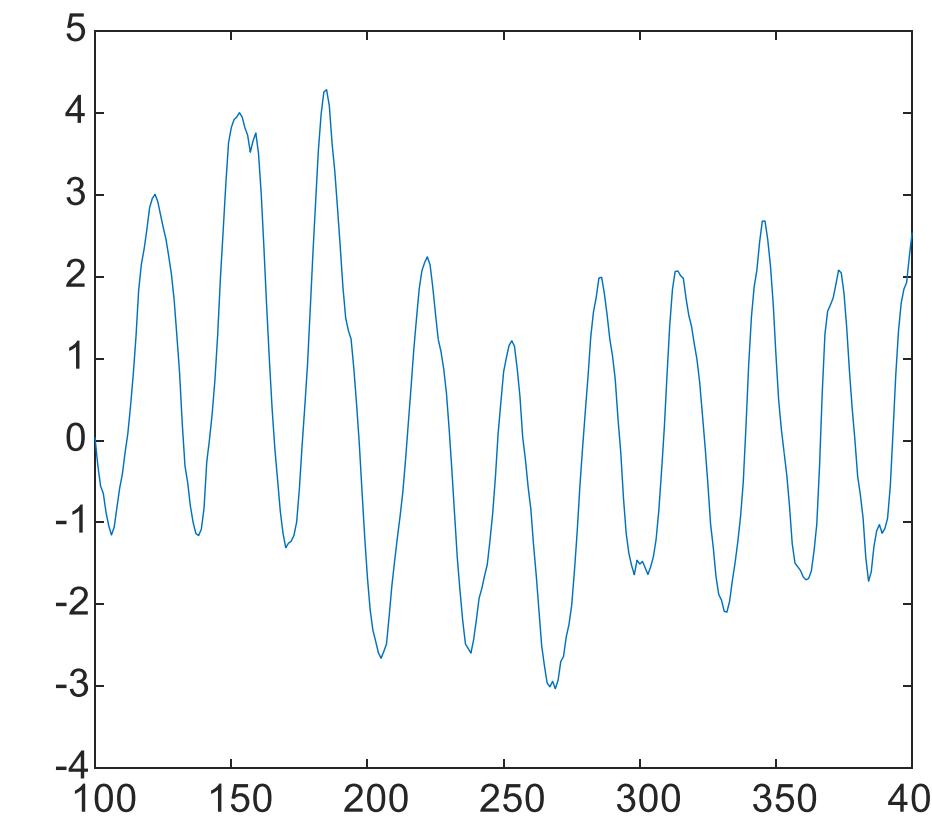
Filtering



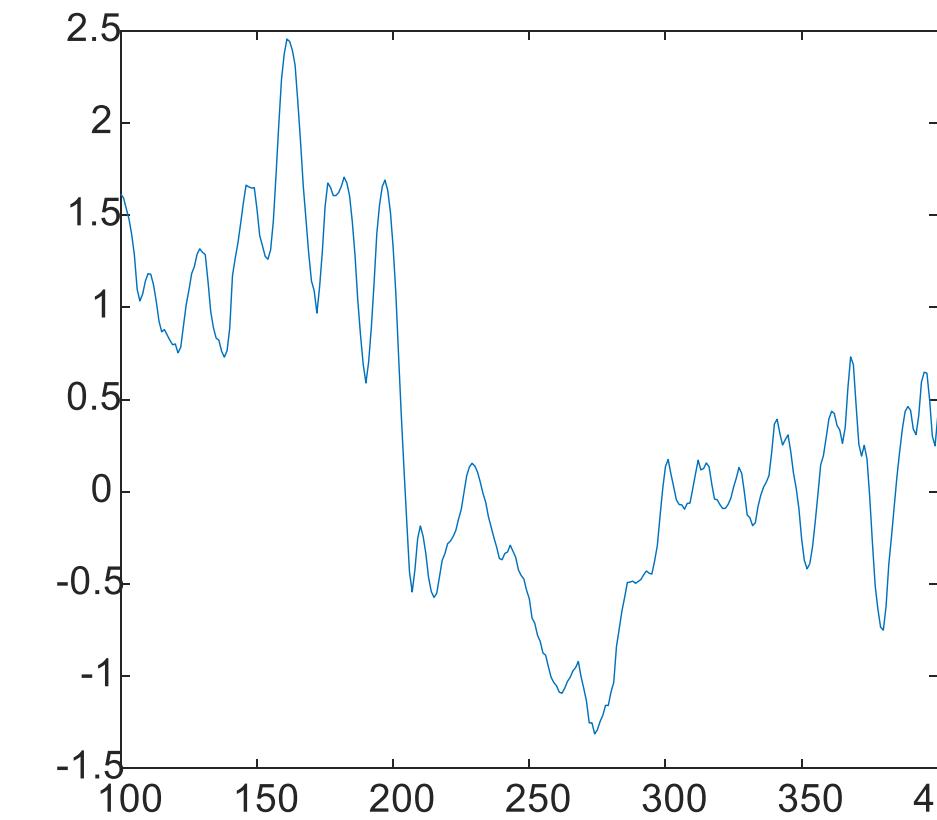
$N=1$



$N=5$



$N=15$

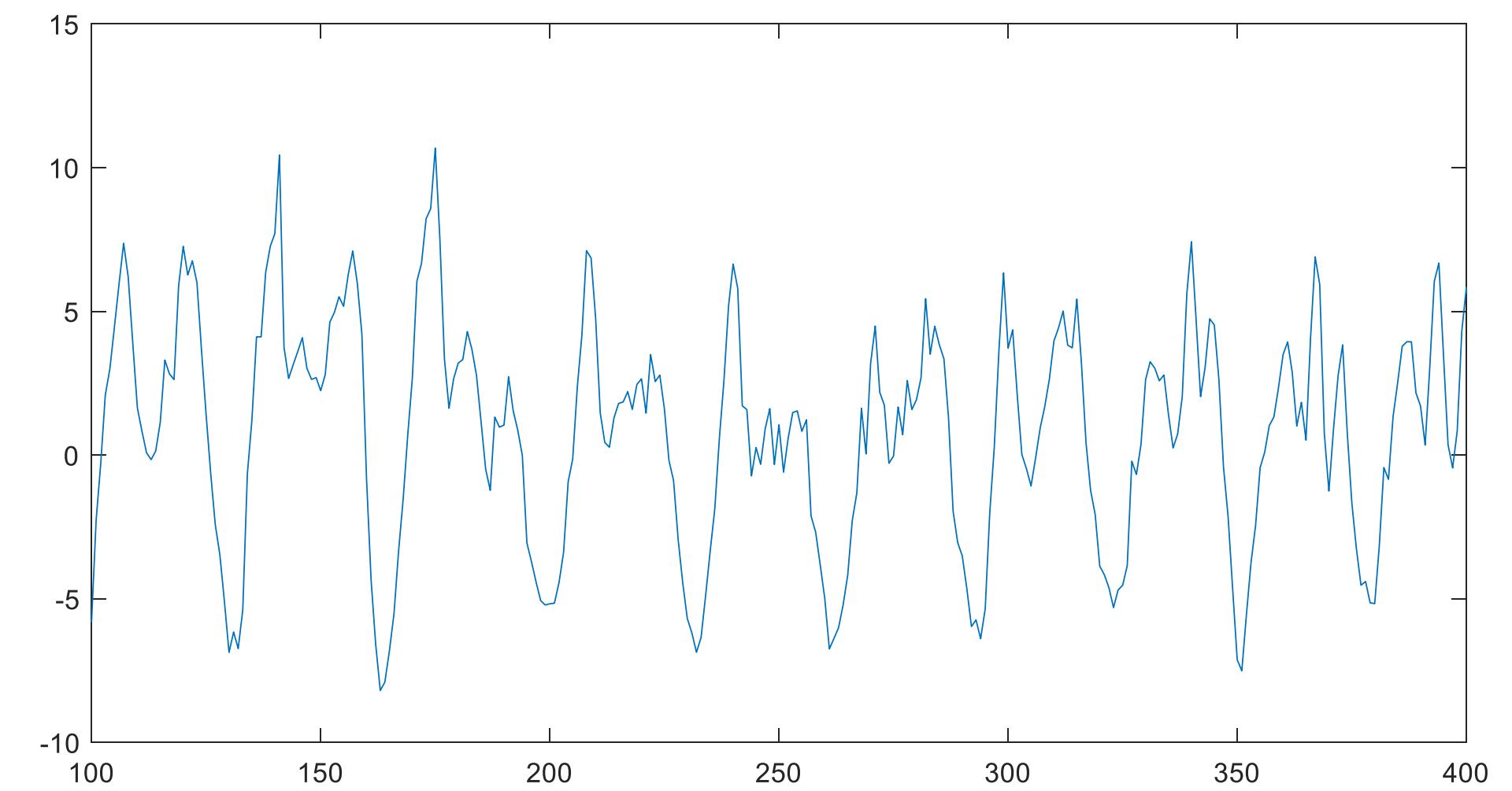


$N=30$

Step counting

Two potential strategies

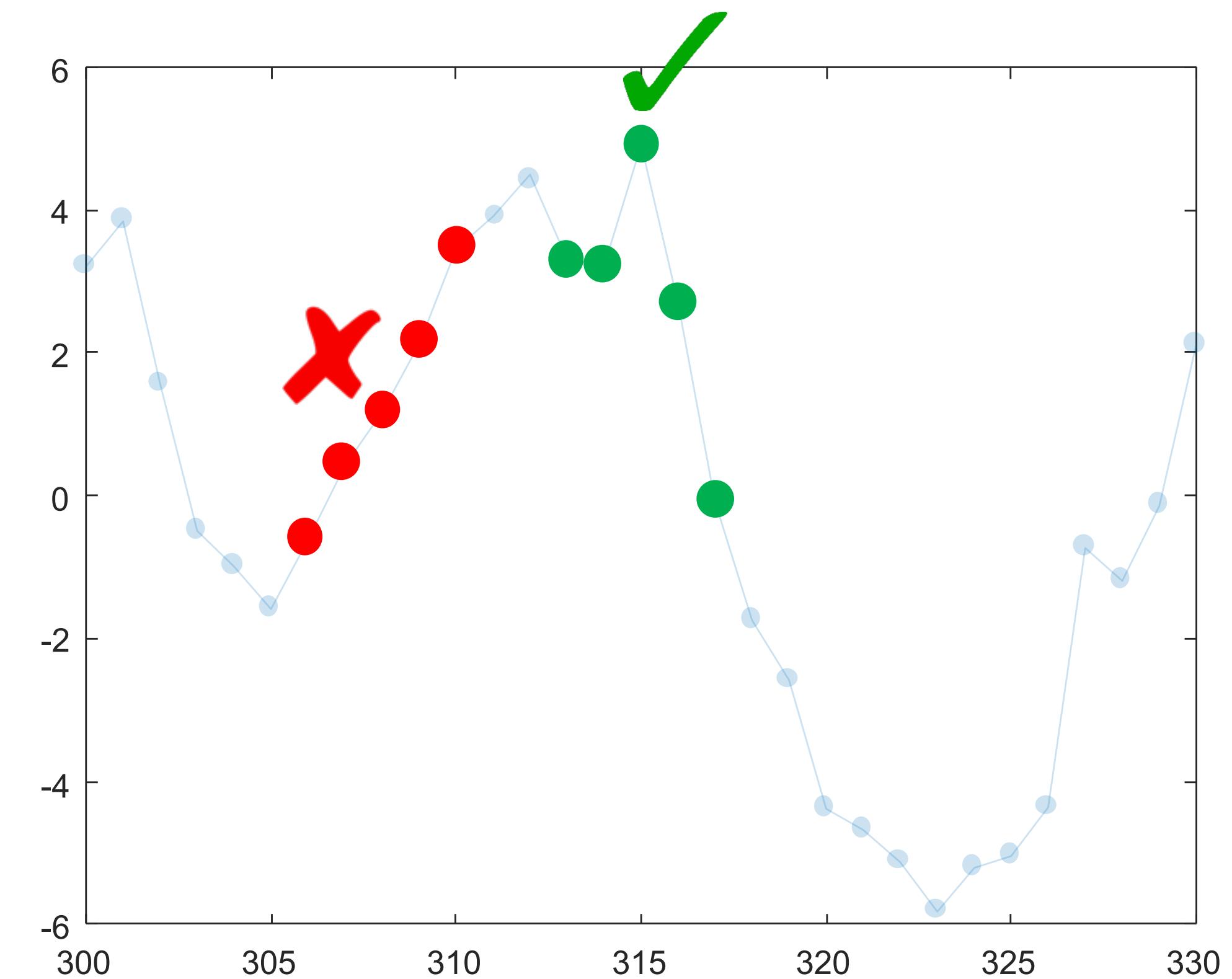
- Peak finding
- Zero crossing



Step counting

Peak finding

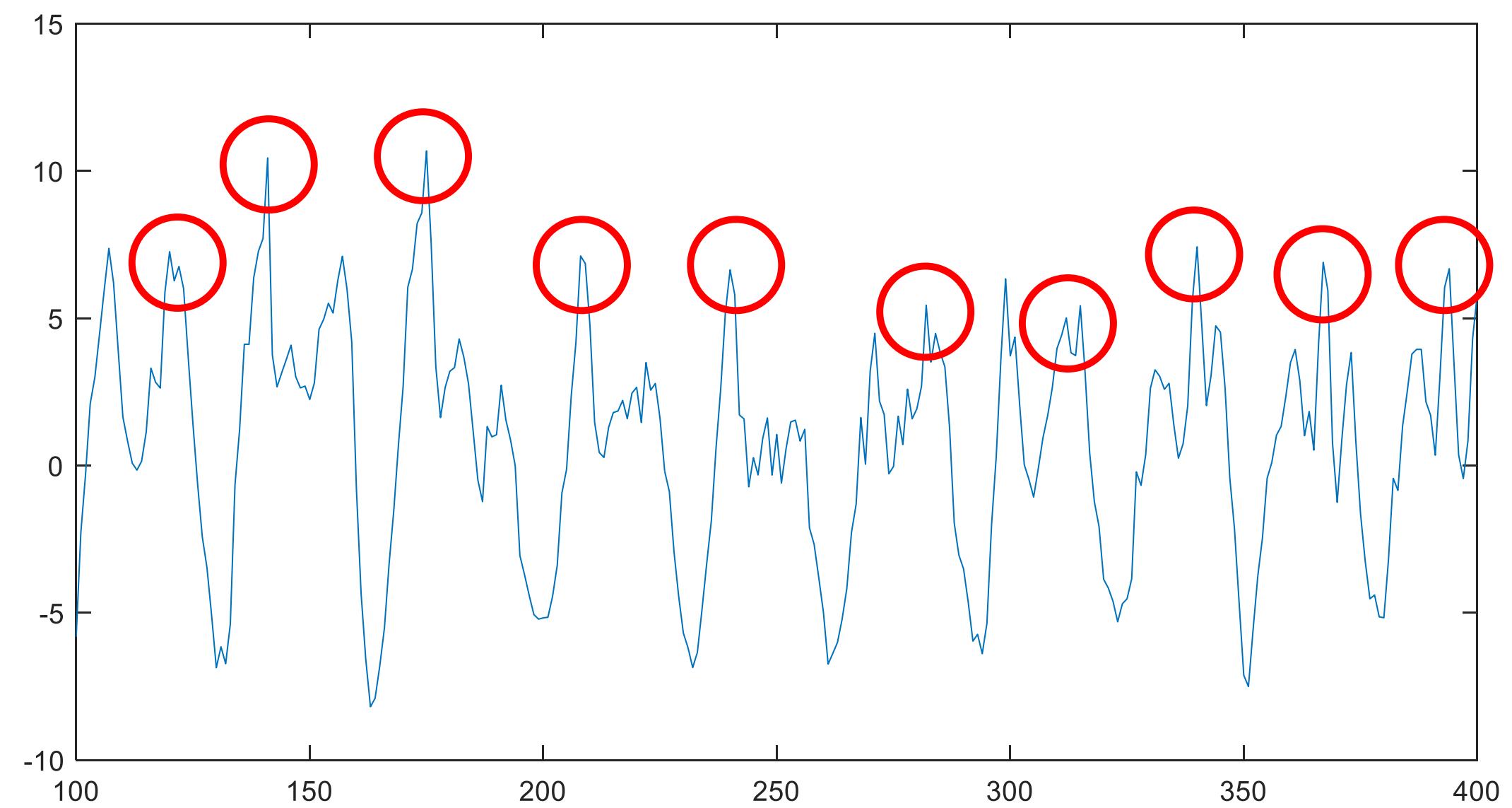
- Is the value in the middle of the sliding window a peak?



Step counting

Peak finding

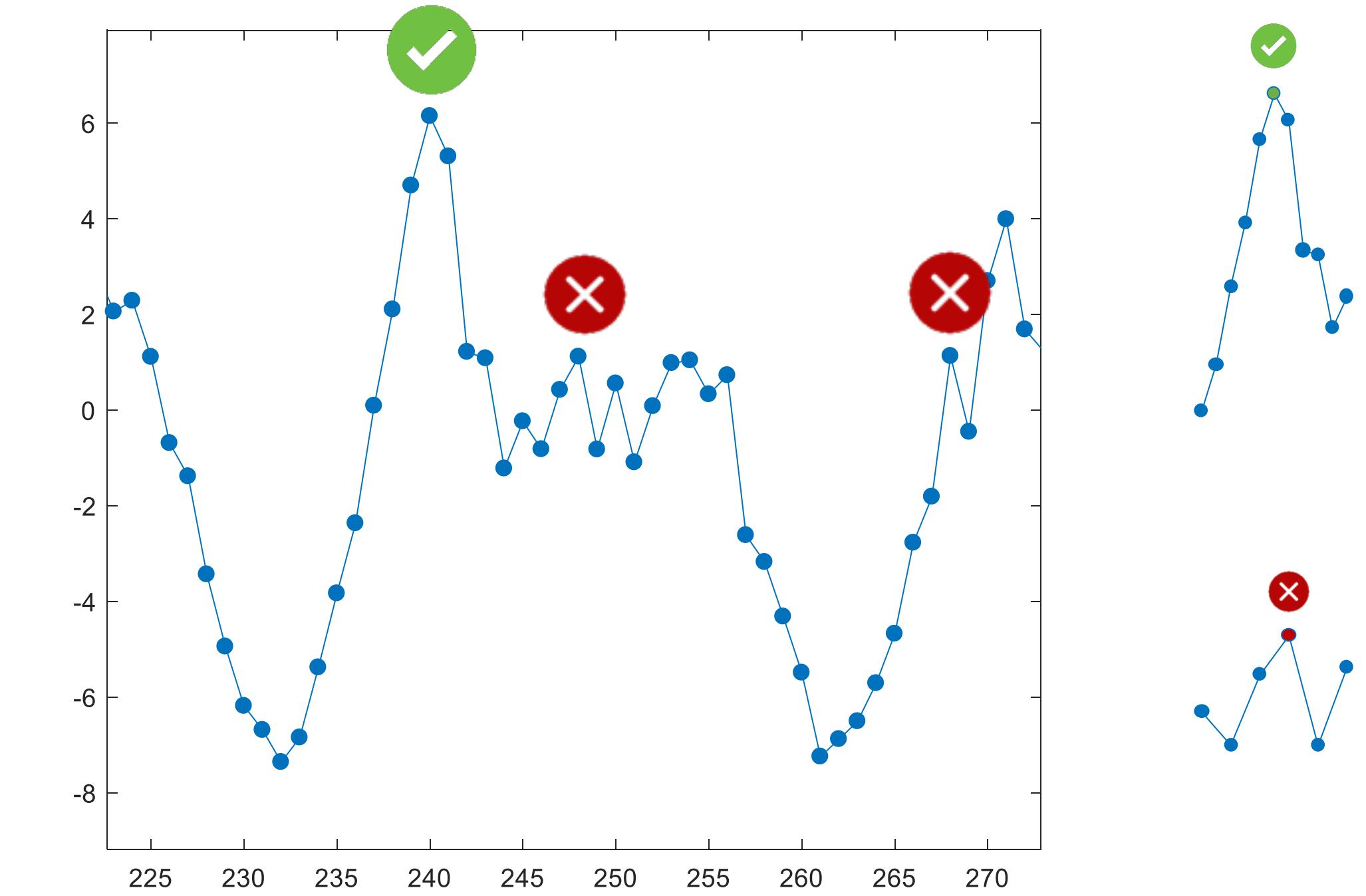
- Is the value in the middle of the sliding window a peak?



Step counting

Peak finding

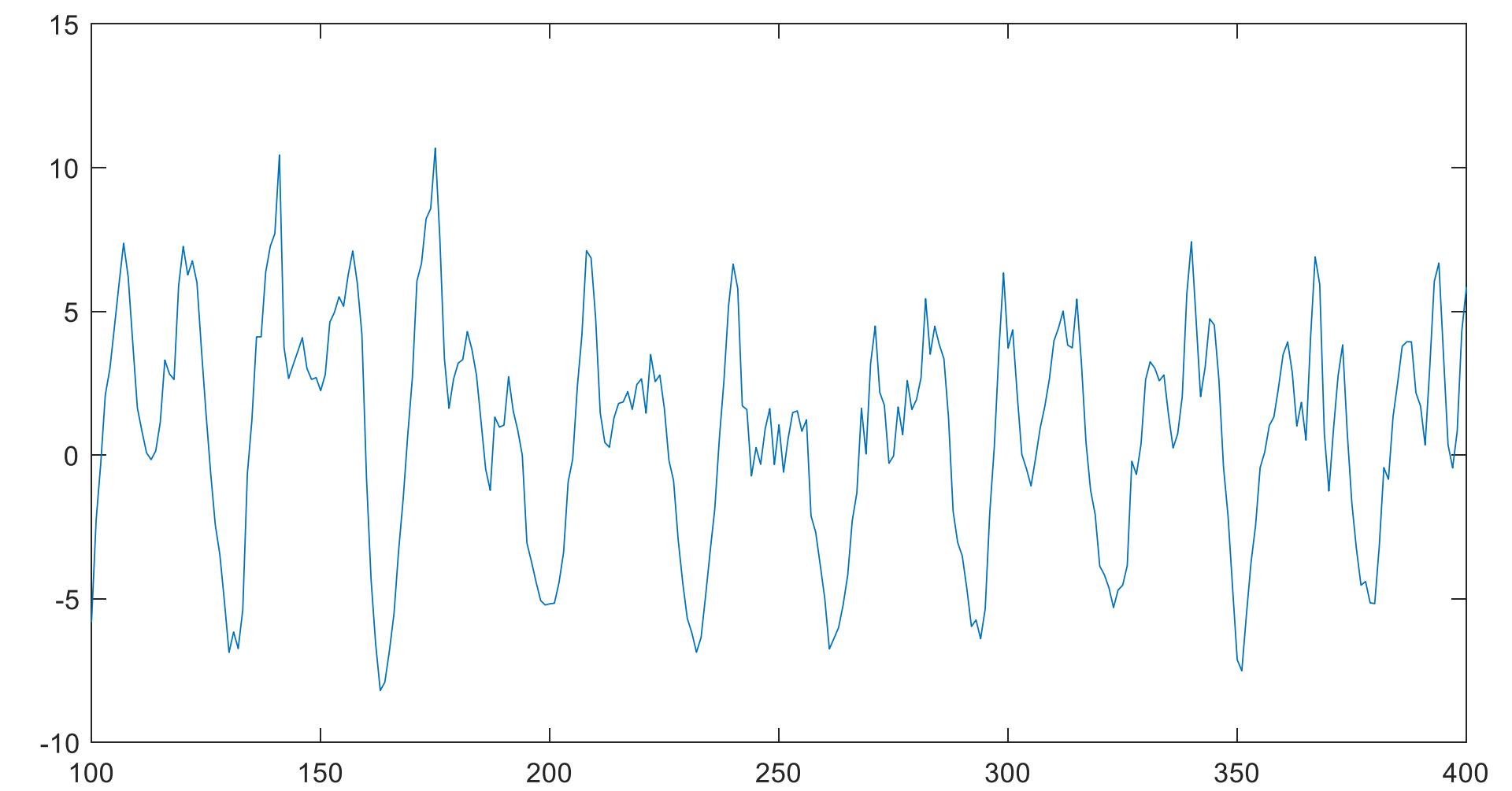
- Distinguishing actual peaks from local maxima
- When is a peak significant enough to indicate a step?
- How long should the window be?
- Could use derivatives in place of pure peak finding



Step counting

Two potential strategies

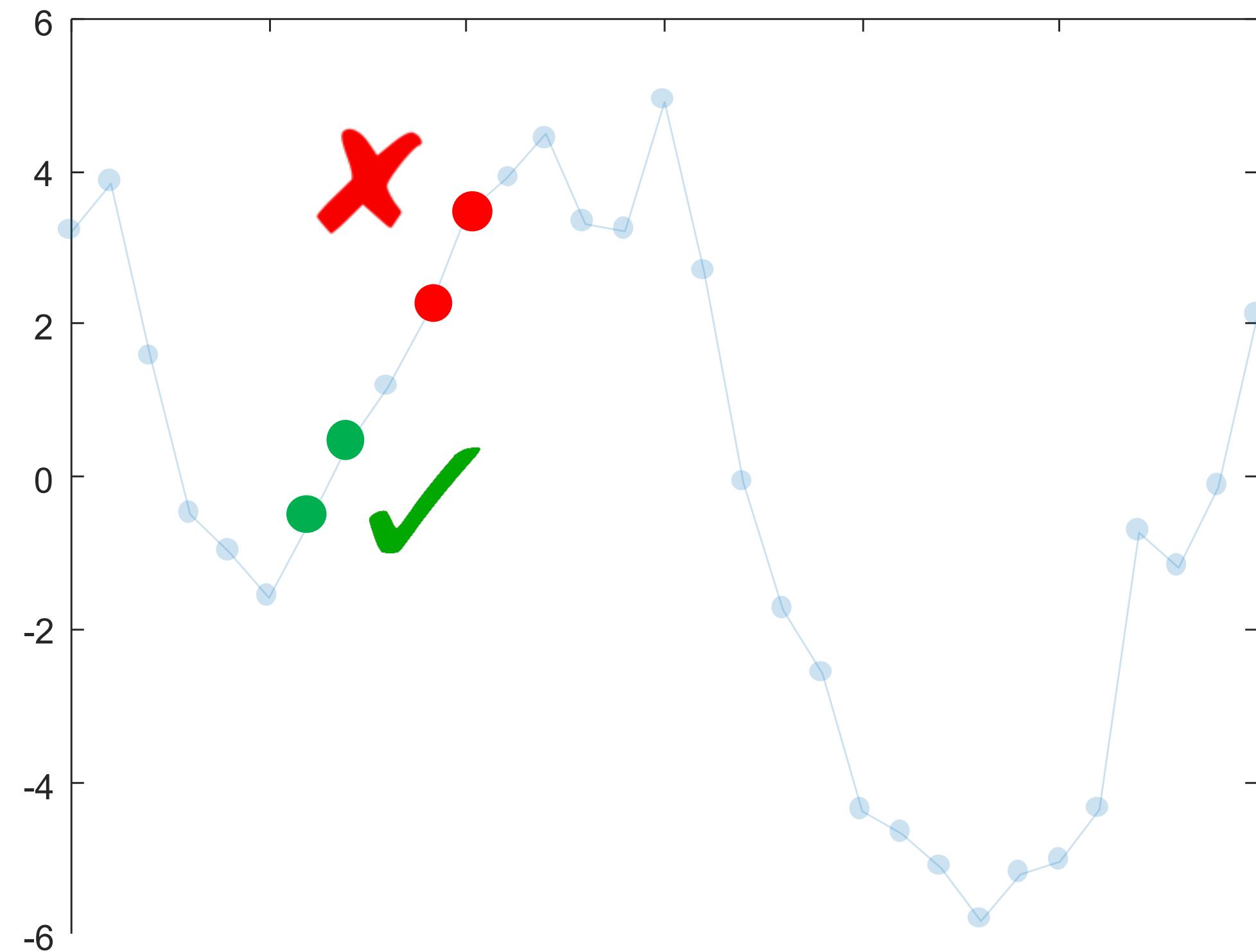
- Peak finding
- Zero crossing



Step counting

Zero crossing

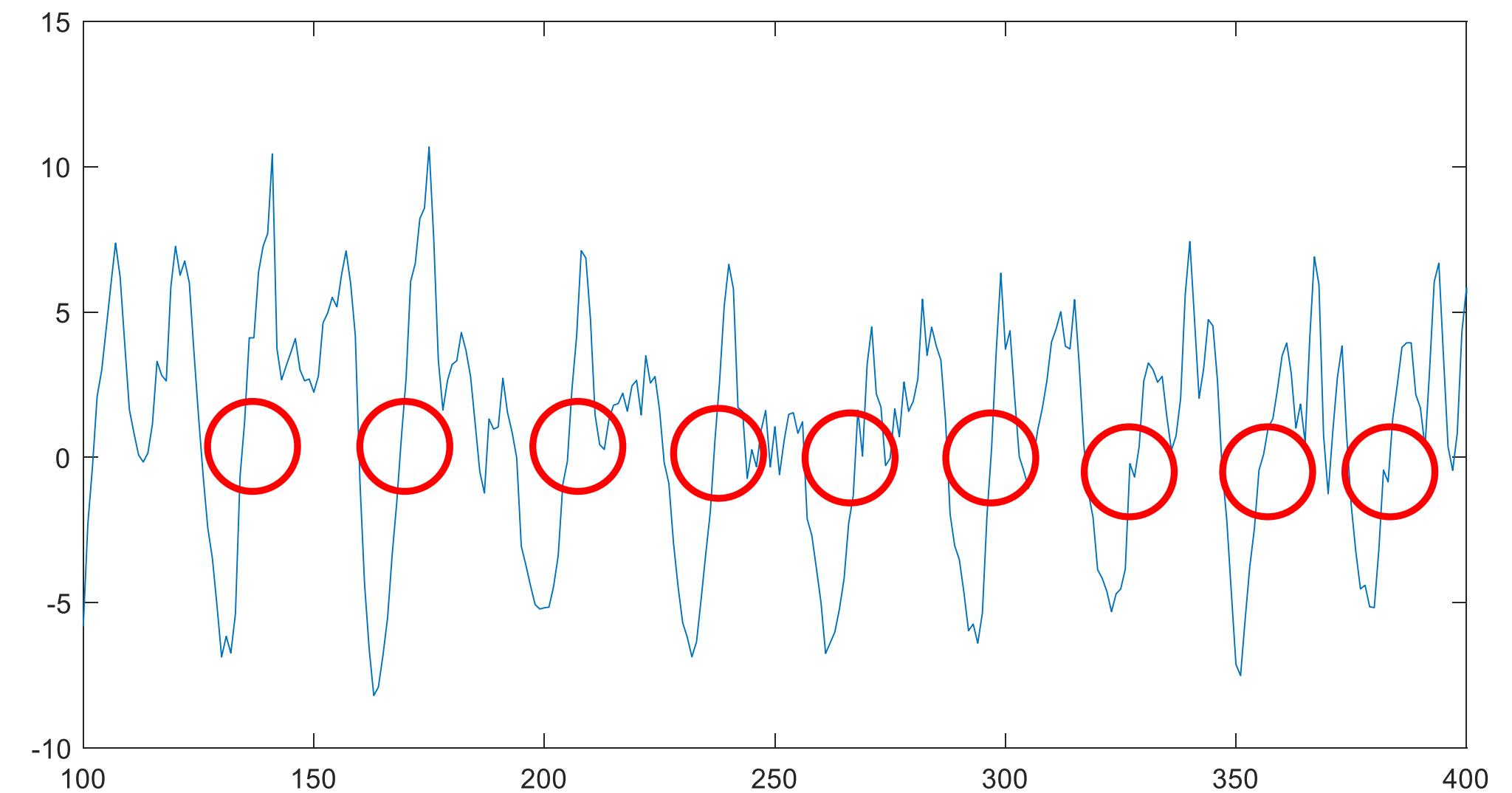
- Does the signal cross the x-axis?
(Note: pick increasing or decreasing)



Step counting

Zero crossing

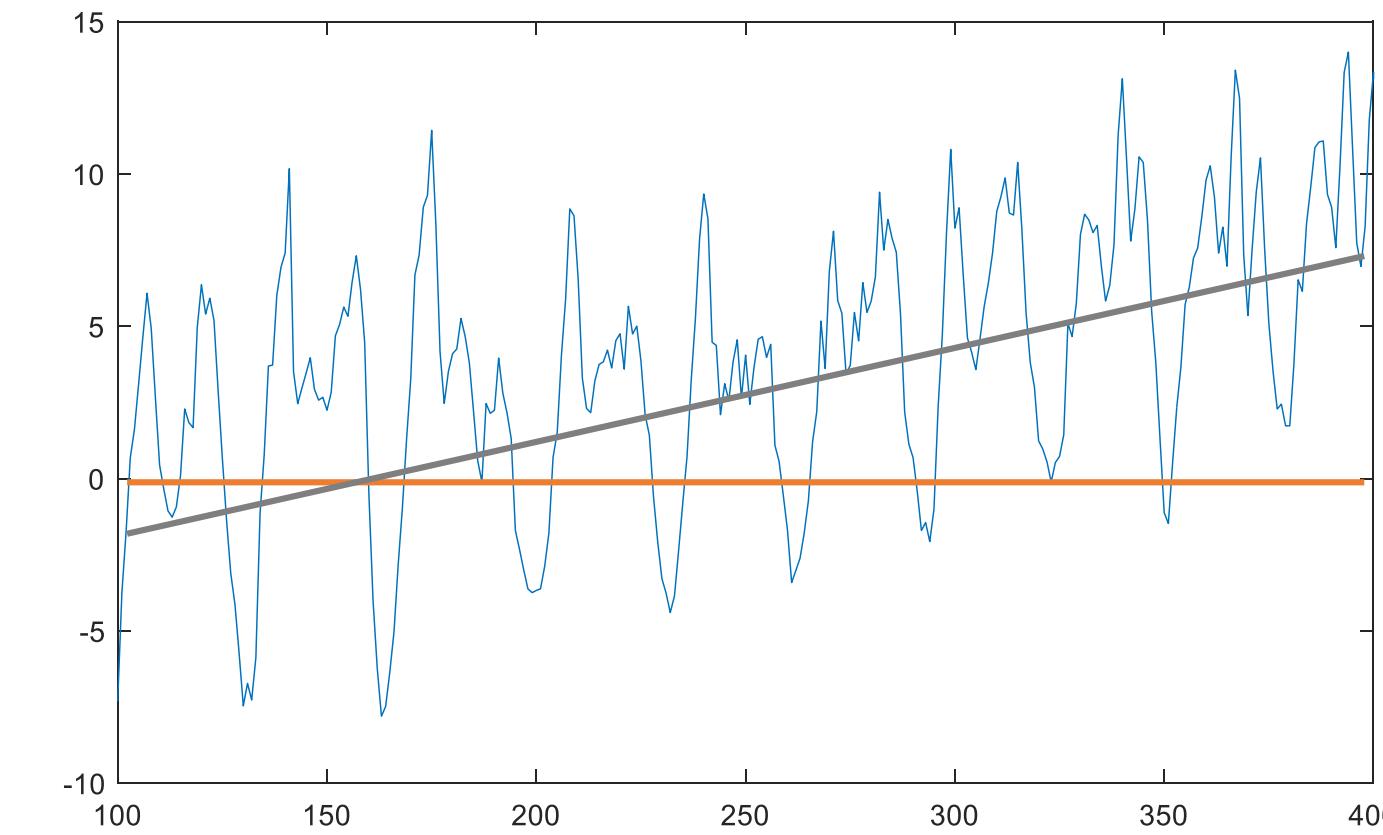
- Does the signal cross the x-axis?
(Note: pick increasing or decreasing)



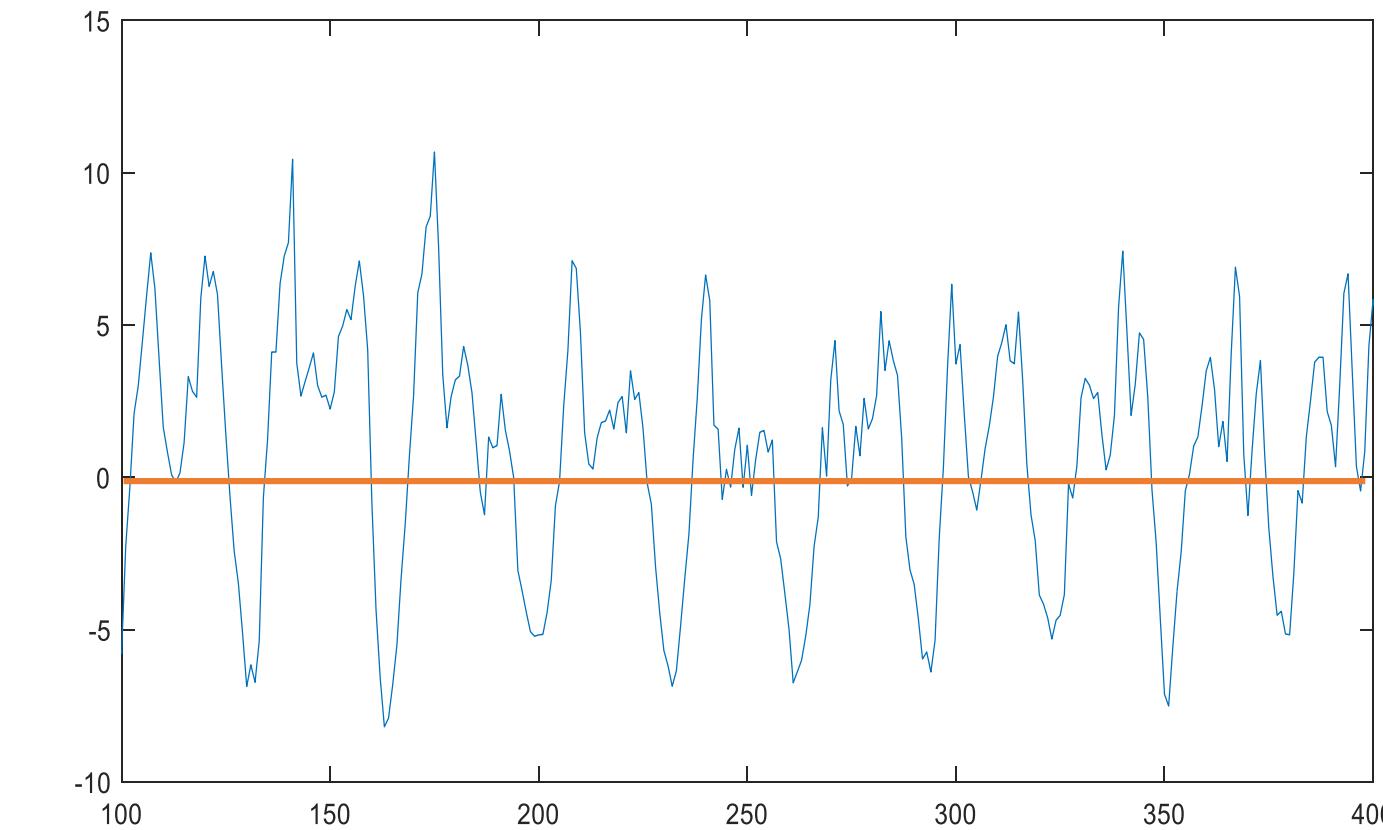
Step counting

Zero crossing

- Signal isn't always centered around zero
- False positive if there are perturbations in the middle
- Smoothing can help with this

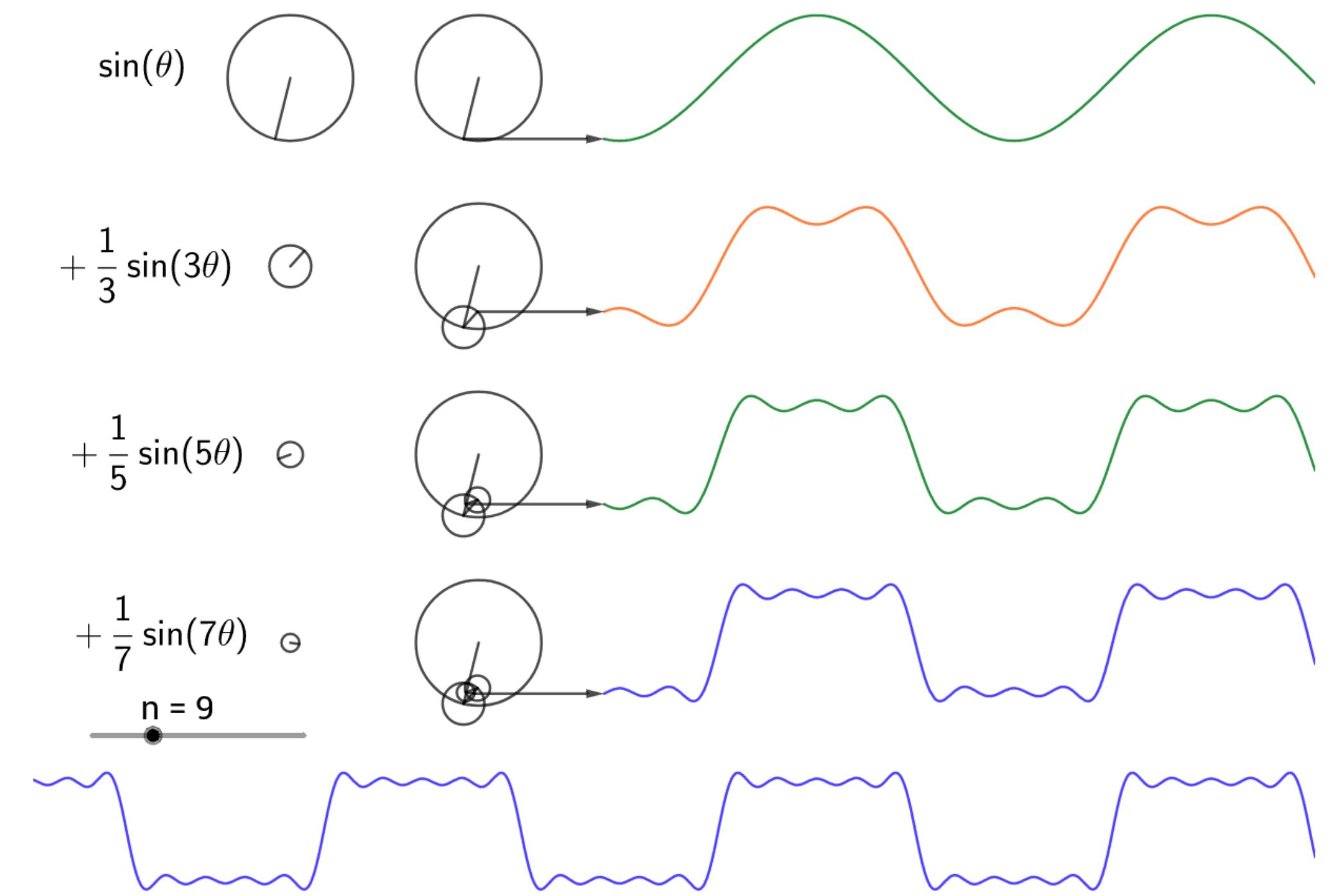


De-mean (subtract average)



Fourier transforms

- There are more complex ways of analyzing signals, such as viewing the signal in the frequency domain
- e.g., when & at what frequencies does the signal oscillate
- You use a *Fourier transform* to translate a signal from the time domain to the frequency domain



https://en.wikipedia.org/wiki/Frequency_domain

Fourier transforms

- There are more complex ways of analyzing signals, such as viewing the signal in the frequency domain
- e.g., when & at what frequencies does the signal oscillate
- You use a *Fourier transform* to translate a signal from the time domain to the frequency domain



https://en.wikipedia.org/wiki/Frequency_domain

Signal Processing in Ubicomp

- Capture signals under controlled conditions
- Use tools to visualize, study, & process the signal
- Search for previous solutions to comparable problems
- Generate and test approaches offline using test data
- When offline works well, adapt approach to perform in real-time

IN4MATX 241: Ubiquitous Computing

**Class 3:
Signal Processing &
Activity Recognition**

Daniel Epstein