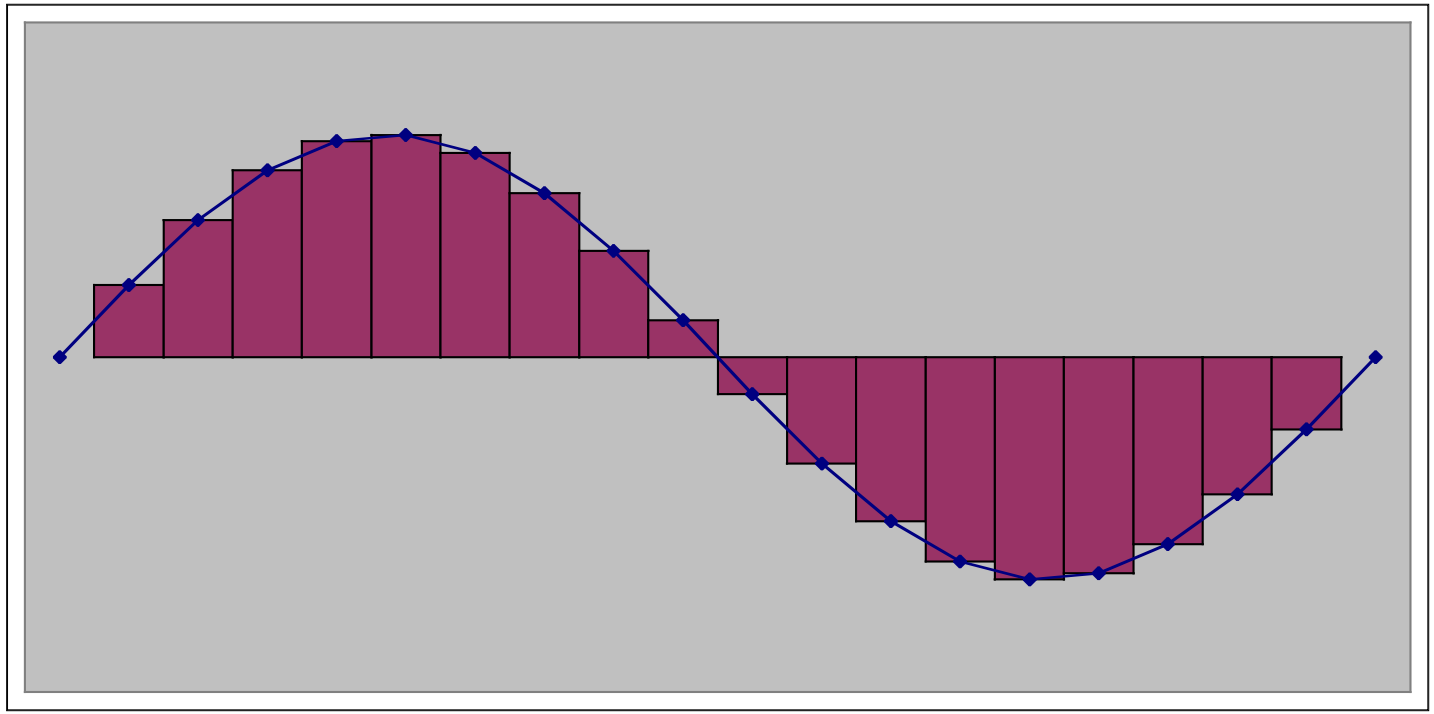
Digitizing Sound: How Do We Get That into Numbers?

- We can estimate the sound curve
  - Analog-to-digital conversion (ADC) will give us the amplitude at an instant as a number: a sample
  - How many samples do we need?



Nyquist Theorem

- We need twice as many samples as the maximum frequency in order to represent (and recreate, later) the original sound.
- The number of samples recorded per second is the sampling rate
  - If we capture 8000 samples per second, the highest frequency we can capture is 4000 Hertz
    - That's how phones work
  - If we capture more than 44,000 samples per second, we capture everything that we can hear (max 22,000 Hertz)
    - CD-quality is 44,100 samples per second

Digitizing Sound in the Computer

- Each sample is stored as a number (two bytes)
- What's the range of available combinations?
  - 16 bits, 216 = 65,536
  - But we want both positive and negative values
    - To indicate compressions and rarefactions.
  - What if we use one bit to indicate positive (0) or negative (1)?
  - That leaves us with 15 bits
  - 15 bits, 215 = 32,768
  - One of those combinations will stand for zero
    - We'll use a "positive" one, so that's one less pattern for positives

Sounds as Arrays

- Samples are just stored one right after the other in the computer's memory

(Like pixels in a picture)

- That's called an array
  - It's an especially efficient (quickly accessed) memory structure

Recognize Some Similarities?

```
def increaseVolume(sound):
  for sample in getSamples(sound):
    value = getSampleValue(sample)
    setSampleValue(sample, value*2)
```

```
def increaseRed(picture):
  for p in getPixels(picture):
    value=getRed(p)
    setRed(p,value*1.2)
```

```
def decreaseVolume(sound):
  for sample in getSamples(sound):
    value = getSampleValue(sample)
    setSampleValue(sample, value*0.5)
```

```
def decreaseRed(picture):
  for p in getPixels(picture):
    value=getRed(p)
    setRed(p,value*0.5)
```