

The background of the slide is a dark blue field filled with a complex, glowing network of thin blue lines connecting numerous small dots, creating a sense of digital connectivity and data flow. Some areas of the network are more densely packed, while others are more sparse.

Lecture slides of the course  
**Information hiding & secret sharing**

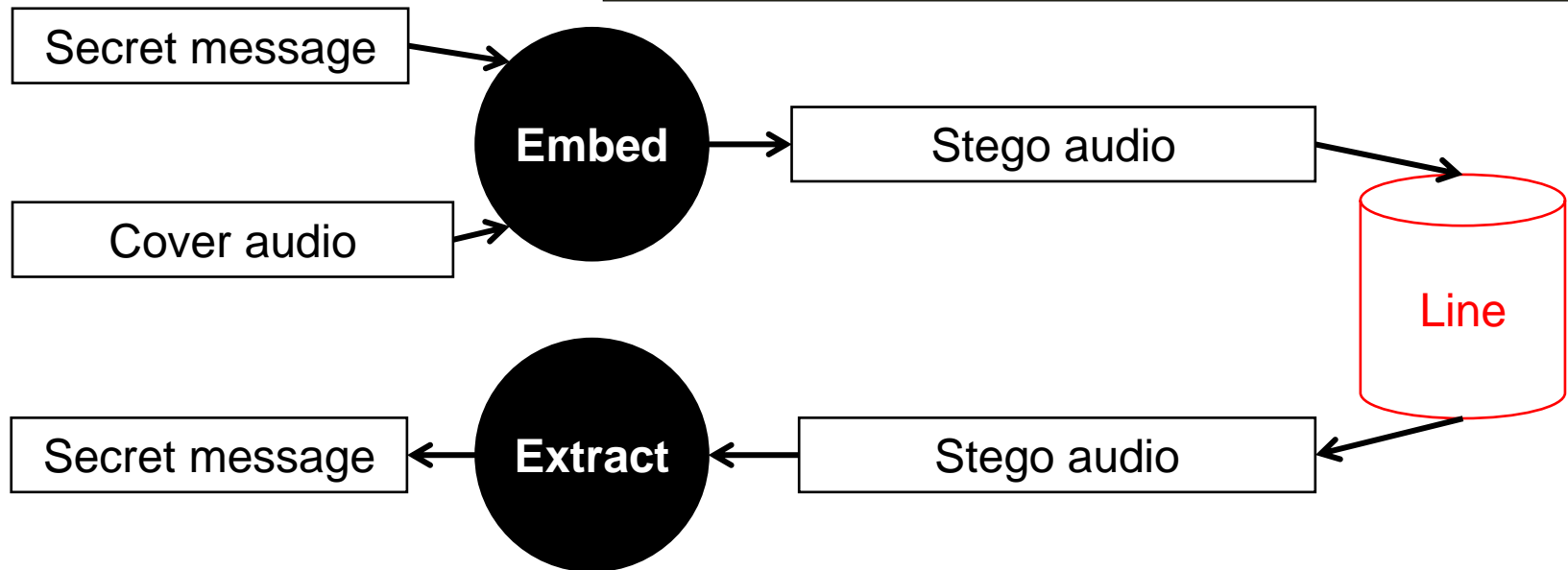
# Audio Steganography

Phạm Trọng Nghĩa  
[ptnghia@fit.hcmus.edu.vn](mailto:ptnghia@fit.hcmus.edu.vn)

# Audio steganography

## Desire:

- **Invisibility:** it must be difficult for a third person to know the existence of secret information in stego audio
- In addition, there are other desires: **capacity**, **robustness**



# Audio steganography

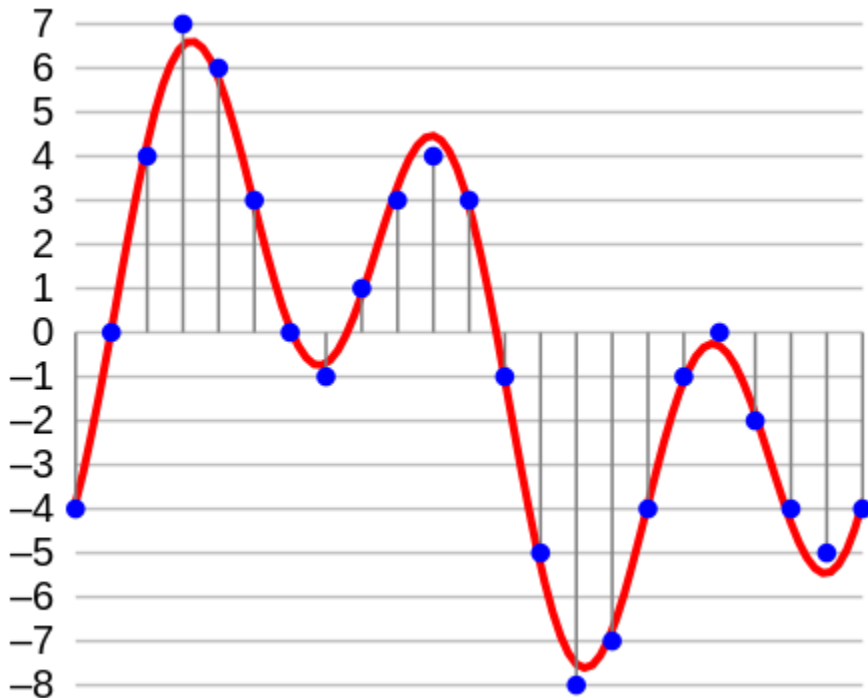
---

**Q: Where should the secret be embedded in the audio?**

**A: First, need to understand the sound...**

# How is sound represented in a computer?

- Try reading an audio file and see ... (demo)
- Sound is represented in the computer as a discrete sound wave.



**Red** line: original wave

**Blue** points: original wave is discretized

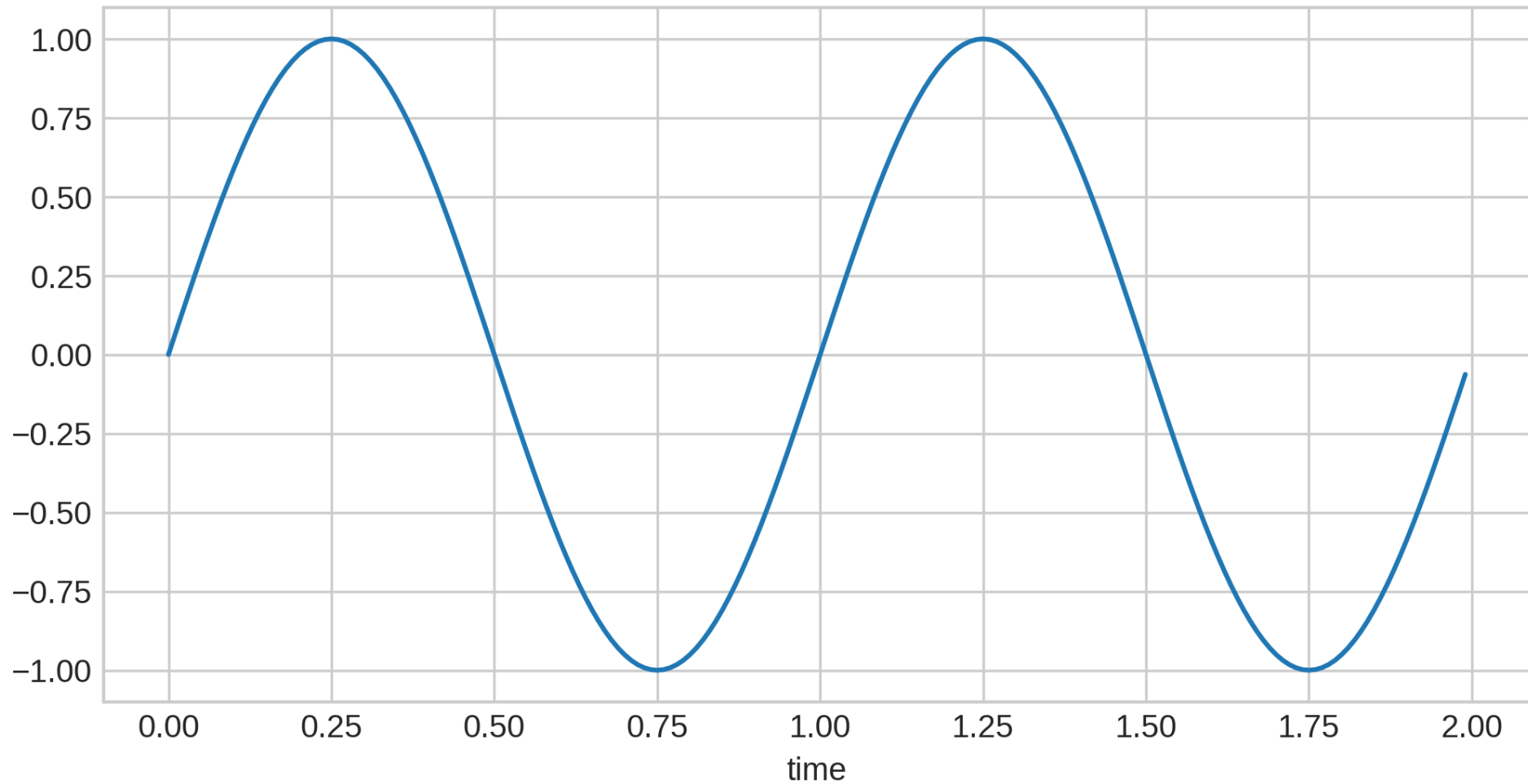
- Each point is called a sample
- The degree of discretization in the **y** direction (intensity) is represented by **bit depth** (the number of bits used to store the intensity value of a sample).
- The degree of discretization in the **x** dimension (time) is represented by the **sample rate** (number of samples/s).

Image source:

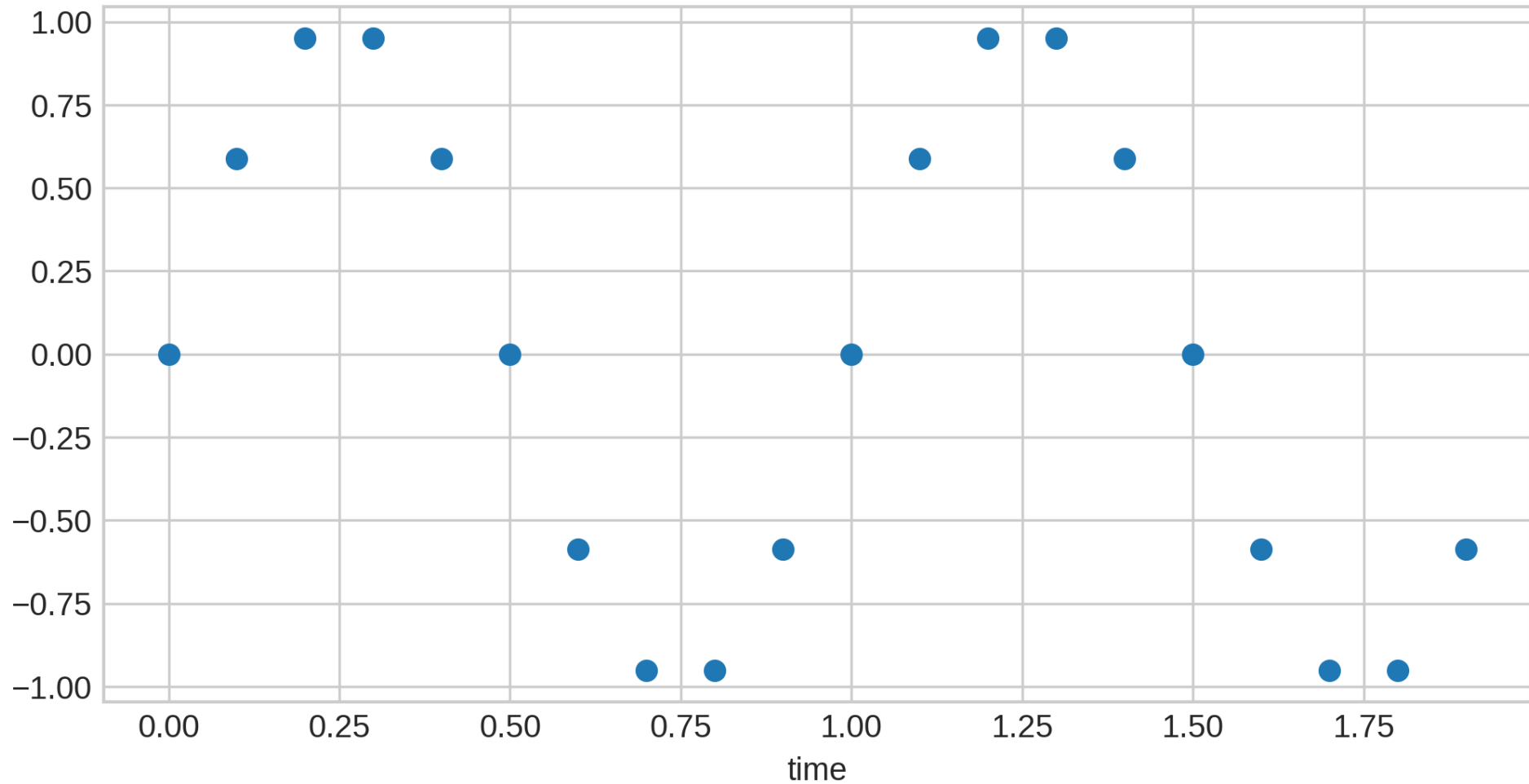
[https://en.wikipedia.org/wiki/Audio\\_bit\\_depth](https://en.wikipedia.org/wiki/Audio_bit_depth)

# Original wave

---

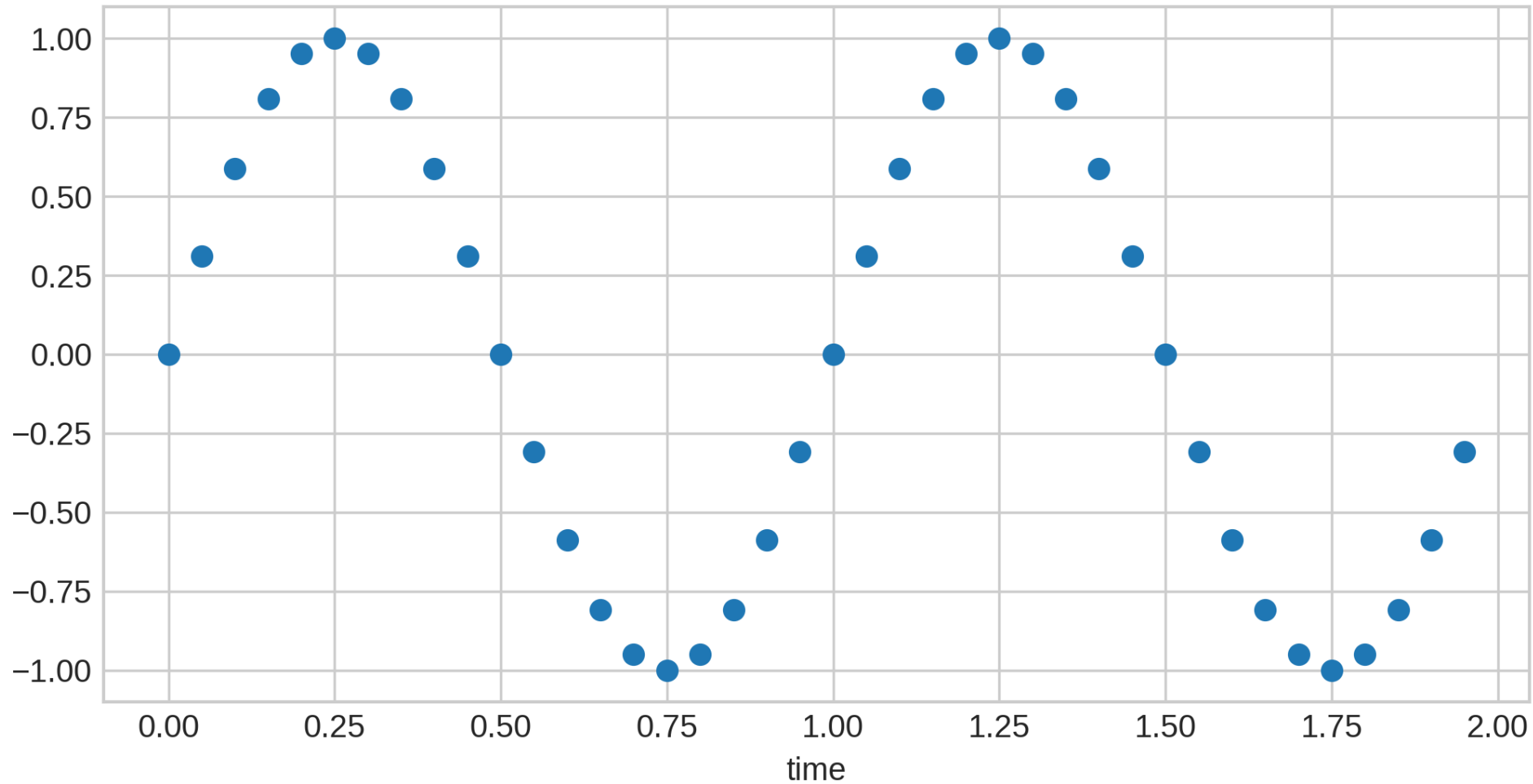


# Sample 20 points



# Sample 40 points

---



# How is sound represented in a computer?

---

- Audio files can be **mono** or **stereo**
  - Mono: one wave
  - Stereo: 2 different wave (same amplitude); when emit, one wave plays on the left speaker, one wave plays on the right speaker  
→ gives a more realistic feel
- Audio files can be **lossless** or **lossy**
  - Lossy: read data from file, edit data, write edited data to file, read data back up, the read data will not be the same as previously recorded data; for example: file \*.mp3
  - Lossless: eg, file \*.wav



# How to hid secret message on audio file (lossless file like\*.wav)

---

- The simple way is to hide the LSB bits of each sample
- Demo ...

# Where to hid secret message on audio file

---

Other wave is using **echo**

- What is Echo?
- How to use echo to embed secret message?
- How to extract?

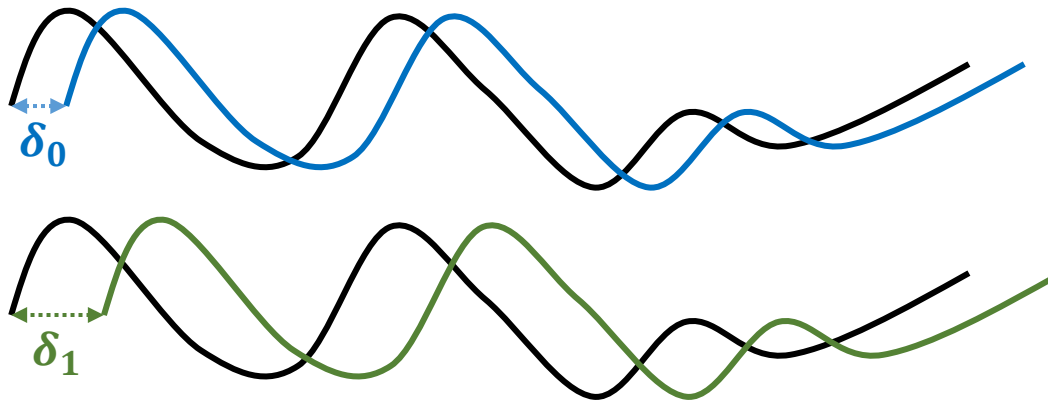
# What is echo?

---

- Watch [video](#)
- Demo create an sound with echo

# How to use echo to embed secret bits??

- To embed **bit 0**: add to cover audio an echo with delay  $\delta_0$
- To embed **bit 1**: add to cover audio an echo with delay  $\delta_1$
- With  $\delta_0$  and  $\delta_1$  small enough, human cannot hear echo.

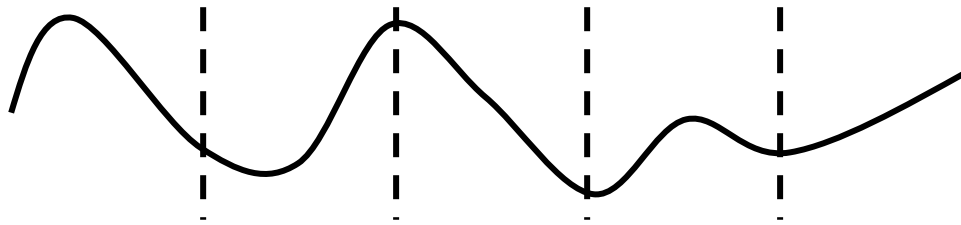


- ☐ So only one bit can be embedded?
- ☐ How to embed multiple bits?

# How to use echo to embed secret bits??

---

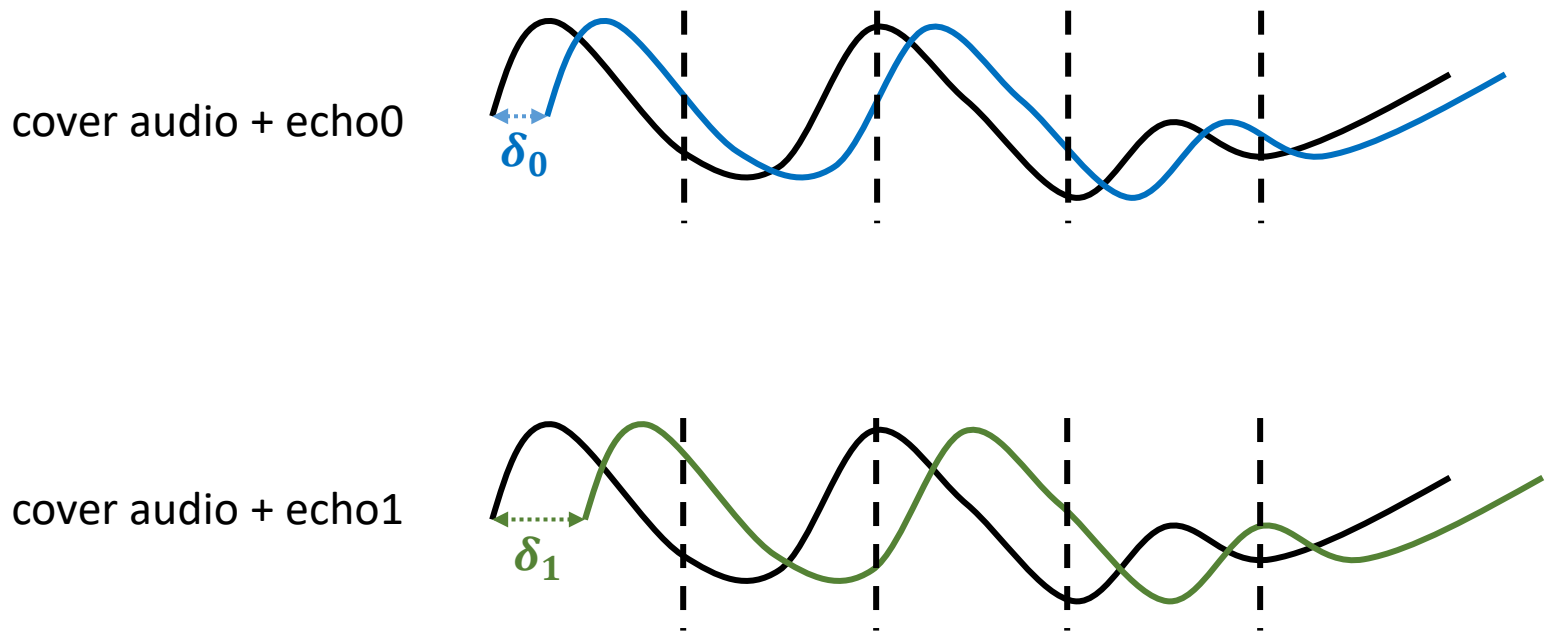
To embed multiple bits, we divide the cover audio into several segments and embed one bit in each segment by adding to that segment echo with delay  $\delta_0$  or  $\delta_1$



# How to use echo to embed secret bits??

More detail about the embedding process :

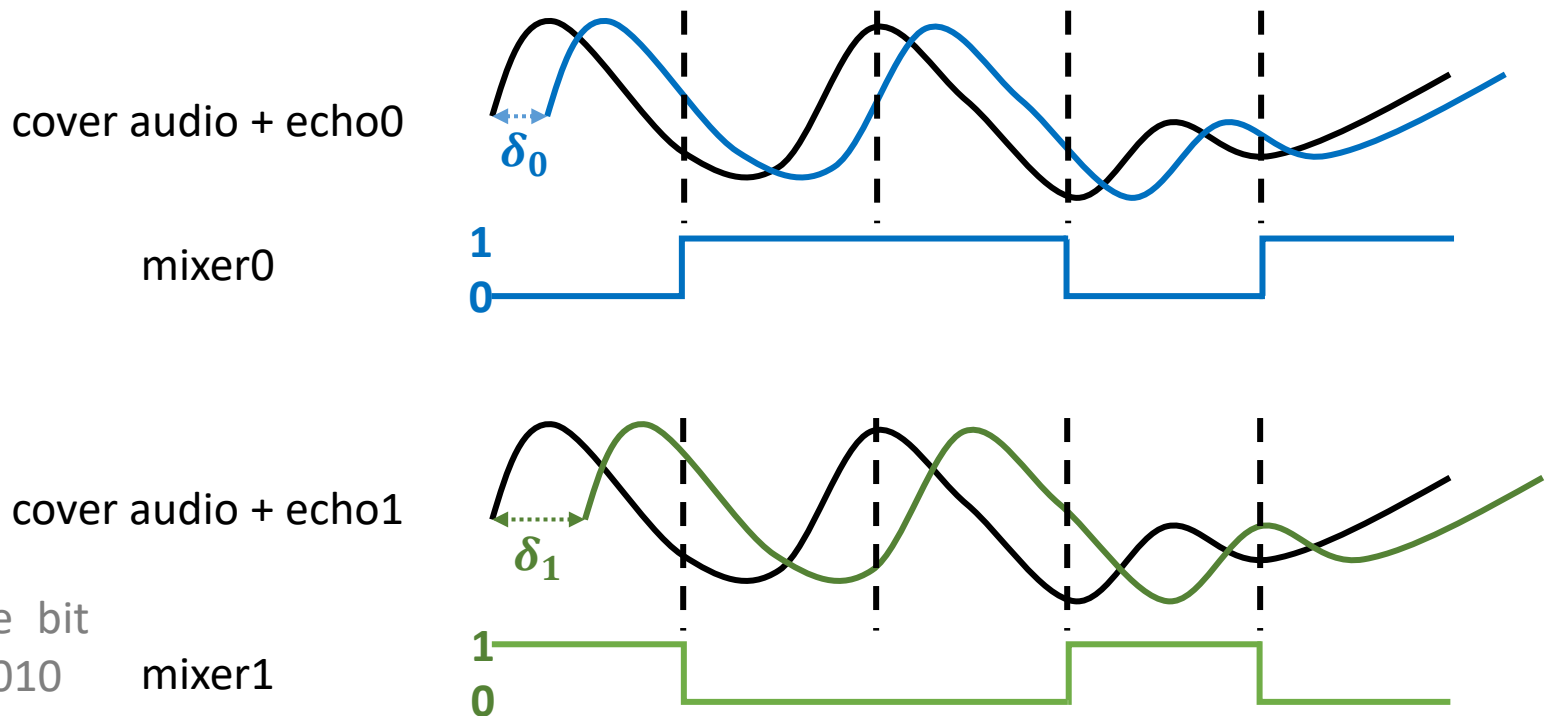
- Step 1: generates 2 cover audio, one with echo corresponding to bit 0, one with echo corresponding to bit 1



# How to use echo to embed secret bits??

More detail about the embedding process :

- Step 2: create mixer0 và mixer1; mixer1 is created based on secret bit string, mixer0 = 1 – mixer1



Let say the bit  
string is 10010

mixer1

# How to use echo to embed secret bits??

More detail about the embedding process ::

- Step 3: stego audio = (cover audio + echo0) \* mixer0 + (cover audio + echo1) \* mixer1

cover audio + echo0

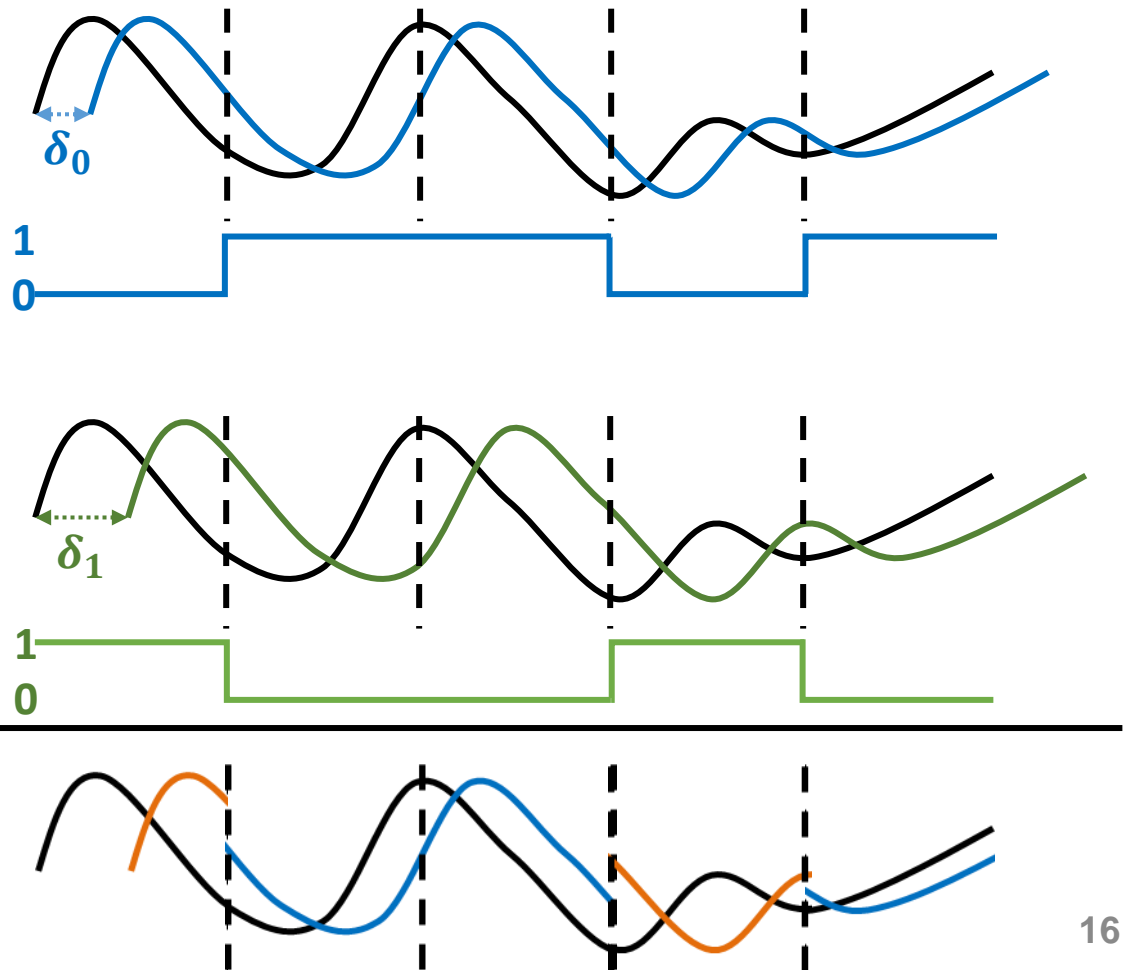
\*  
mixer0

+

cover audio + echo1

\*  
mixer1

stego audio





# Input and output of the embedding process

---

- Input
  - message bits
  - cover audio
  - $\delta_0$ ,  $\delta_1$
  - decay rate  $\in [0, 1]$ : scale ratio of the echo's intensity to the original sound wave
    - What do you gain and lose with a small decay rate??
- Output
  - stego audio

# How to extract?

---

- To extract, it is necessary to know how many audio segments are divided when embedding
  - How can Bob and Alice exchange this information?
  - One way is that Alice and Bob can meet once and agree on the maximum length of the secret bit string → when embedding always divides into the maximum length of the secret bit string + 1 audio track, add 100... to the secret bit string and embed over all audio tracks

# How to extract?

---

- For an audio segment of stego audio, it is necessary to determine what delay echo is present
- Assume that there is echo at delay 0.5 s  $\rightarrow$  bit 0 or bit 1?
- To know if it's bit 0 or 1, let's look at the echo delay in other audio tracks
  - $\rightarrow$  there will be 2 delay values  $\delta_0$  (bit 0) and  $\delta_1$  (bit 1), but don't know where is  $\delta_0$  or  $\delta_1$
  - $\rightarrow$  Alice and Bob choose that

$$\delta_0 < \delta_1$$

# How to extract?

---

- Given a piece of stego audio, how can the echo delay be determined?
- Calculate autocorrelation
  - See demo

# Correlation

$$Var(X) = \sigma^2 = \sum (X - \mu_x)^2 / N]$$

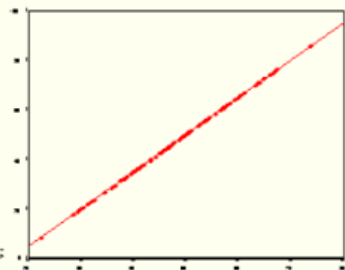
$$Cov(x,y) = \frac{\sum (x_i - \bar{x}) * (y_i - \bar{y})}{N}$$

- Correlation

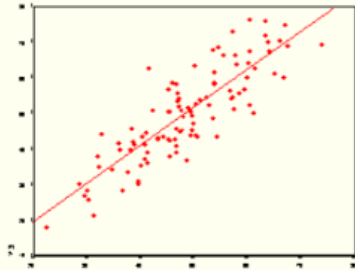
$$\rho_{x,y} = \frac{Cov(X, Y)}{\sqrt{Var(X)Var(Y)}}$$

- Range from -1 to 1

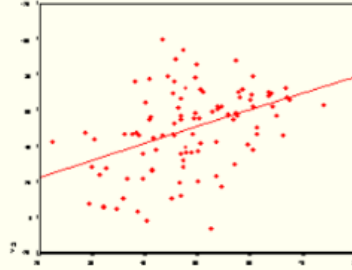
r = 1.00



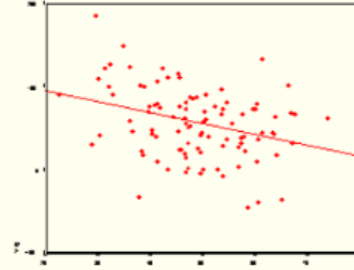
r = .85



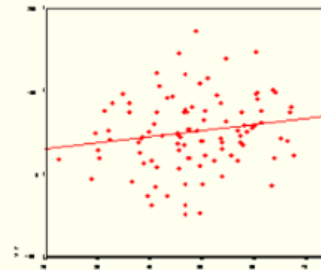
r = .42



r = -.33



r = .17



# Autocorrelation

---

$$r_k = \frac{\sum_{t=k+1}^n (y_t - \bar{y})(y_{t-k} - \bar{y})}{\sum_{t=1}^n (y_t - \bar{y})^2}$$