INFO 4000
Informatics III

Data Science Specialization - Advanced

Week8 Audio analytics and SVD

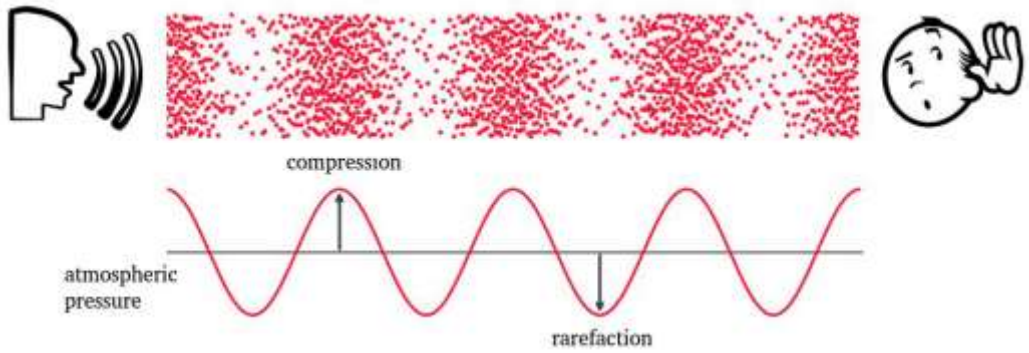
Course Instructor

Jagannath Rao

raoj@uga.edu

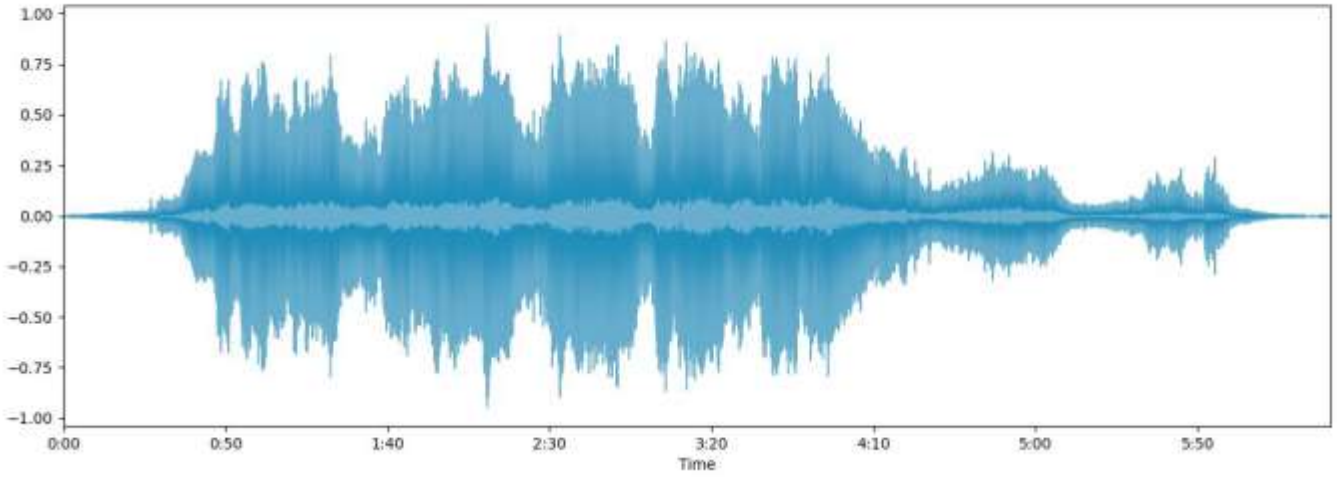
Recap

Sound and Waveforms



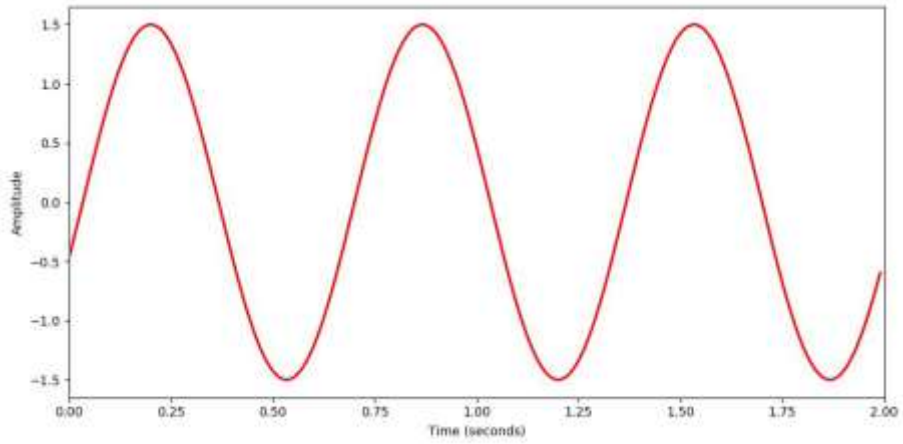
Waveform is the fundamental form of data that we work with in ML as we get info like:

1. Frequency
2. Intensity
3. Timbre
4. and much more out of it

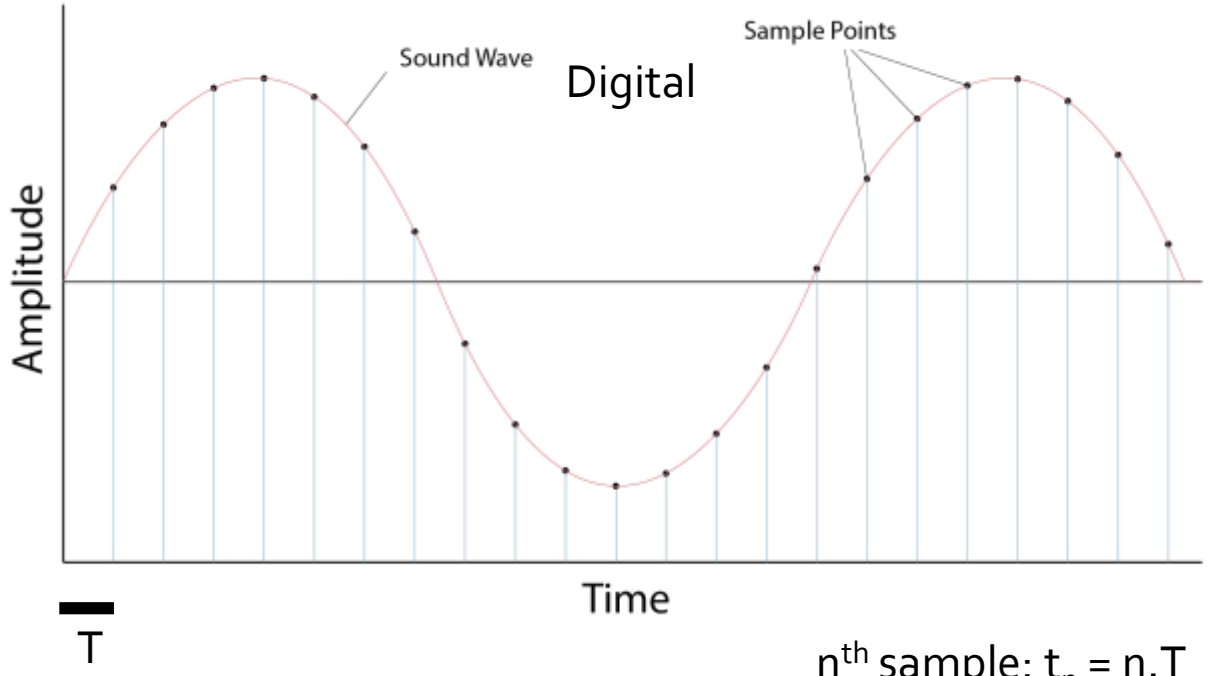


Analog and Digital sound signals

Analog

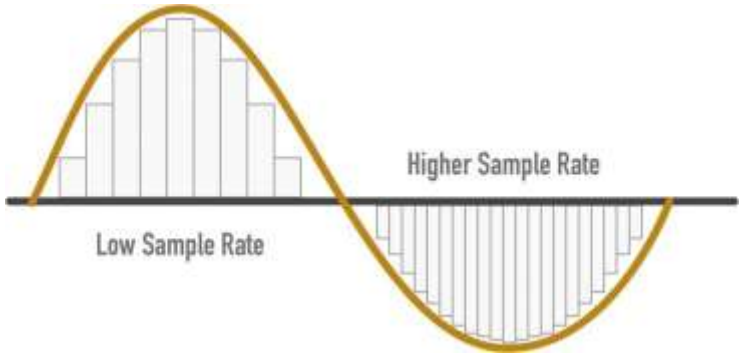


Digital

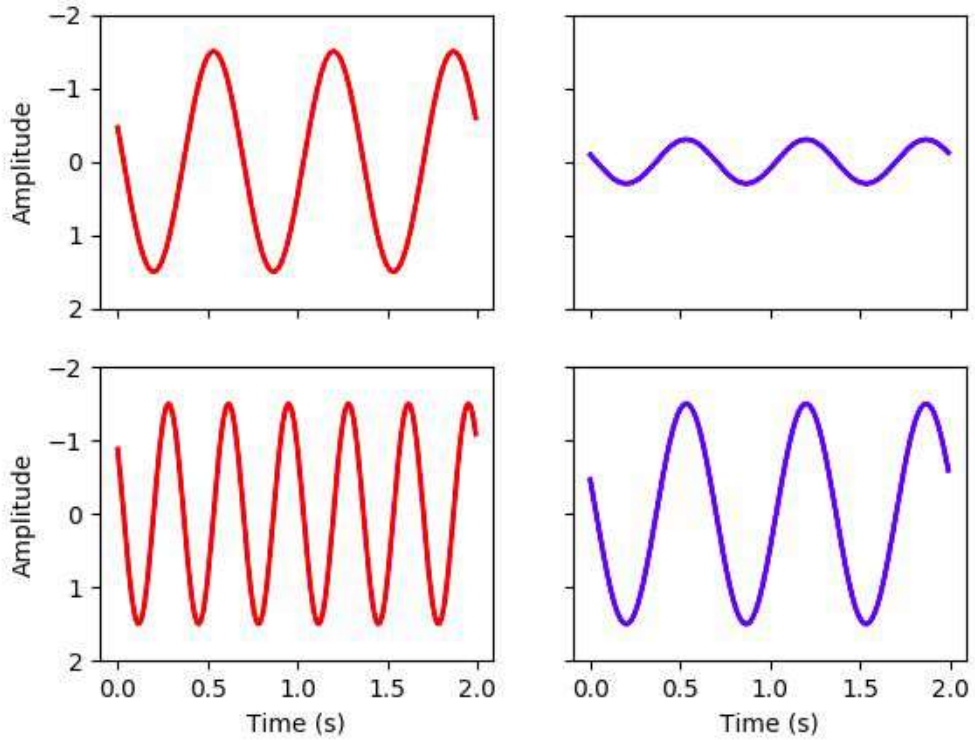


n^{th} sample: $t_n = n.T$

Sample rate: $s_r = 1/T$



Frequency and amplitude



higher frequency -> higher pitch sound

larger amplitude -> louder

Sound Power and Intensity

Power

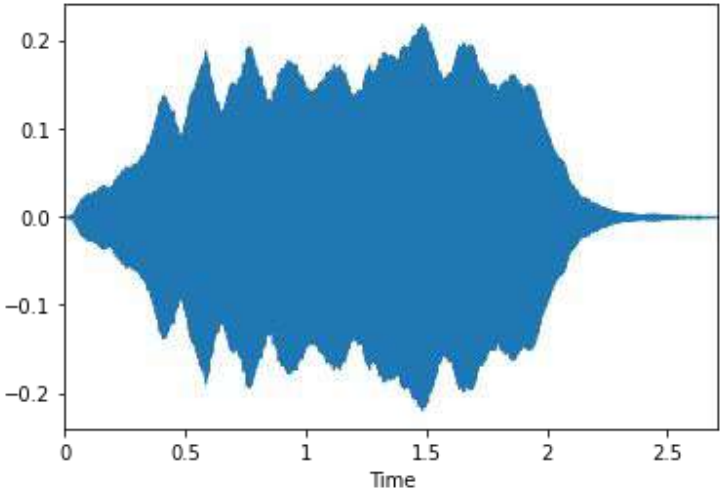
- *Rate at which energy is transferred*
- *Energy per unit of time emitted by a sound source in all directions*
- *Measured in watt (W)*

Intensity

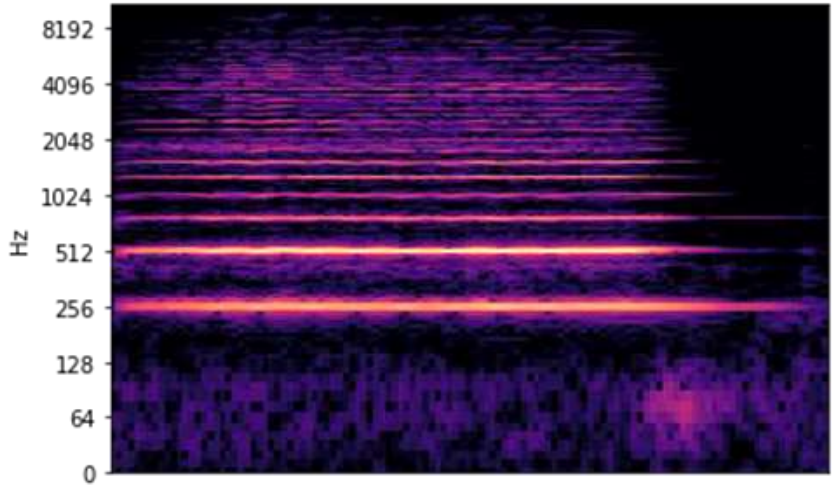
- *Sound power per unit area*
- *Unit – W/m^2*
- *Threshold of hearing = 10^{-12}* W/m^2.
- *Threshold of pain = 10 W/m^2*
- *Logarithmic scale*
- *Measured in decibels (dB)*

Audio data inherently has features in: time, frequency and Cepstral domains

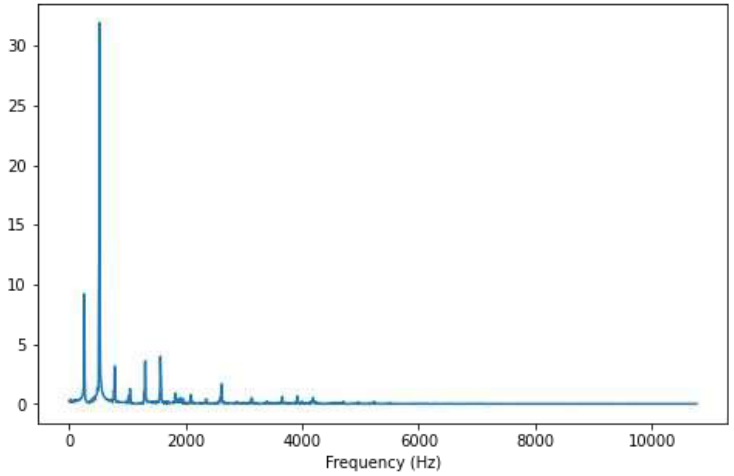
Time Domain



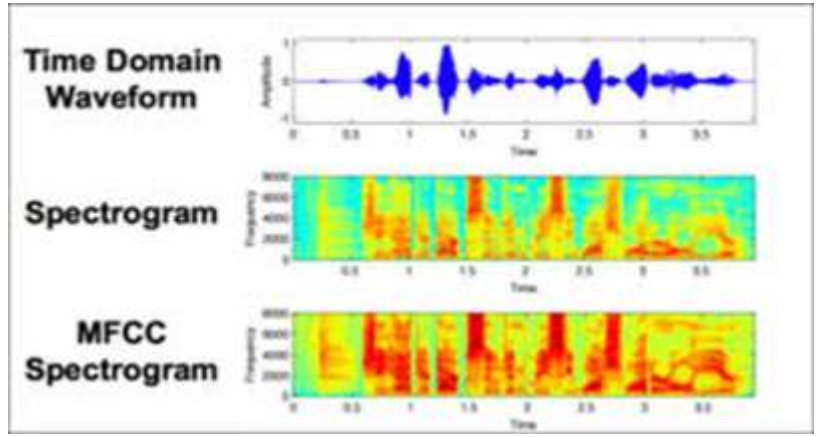
Spectrogram



Frequency Domain

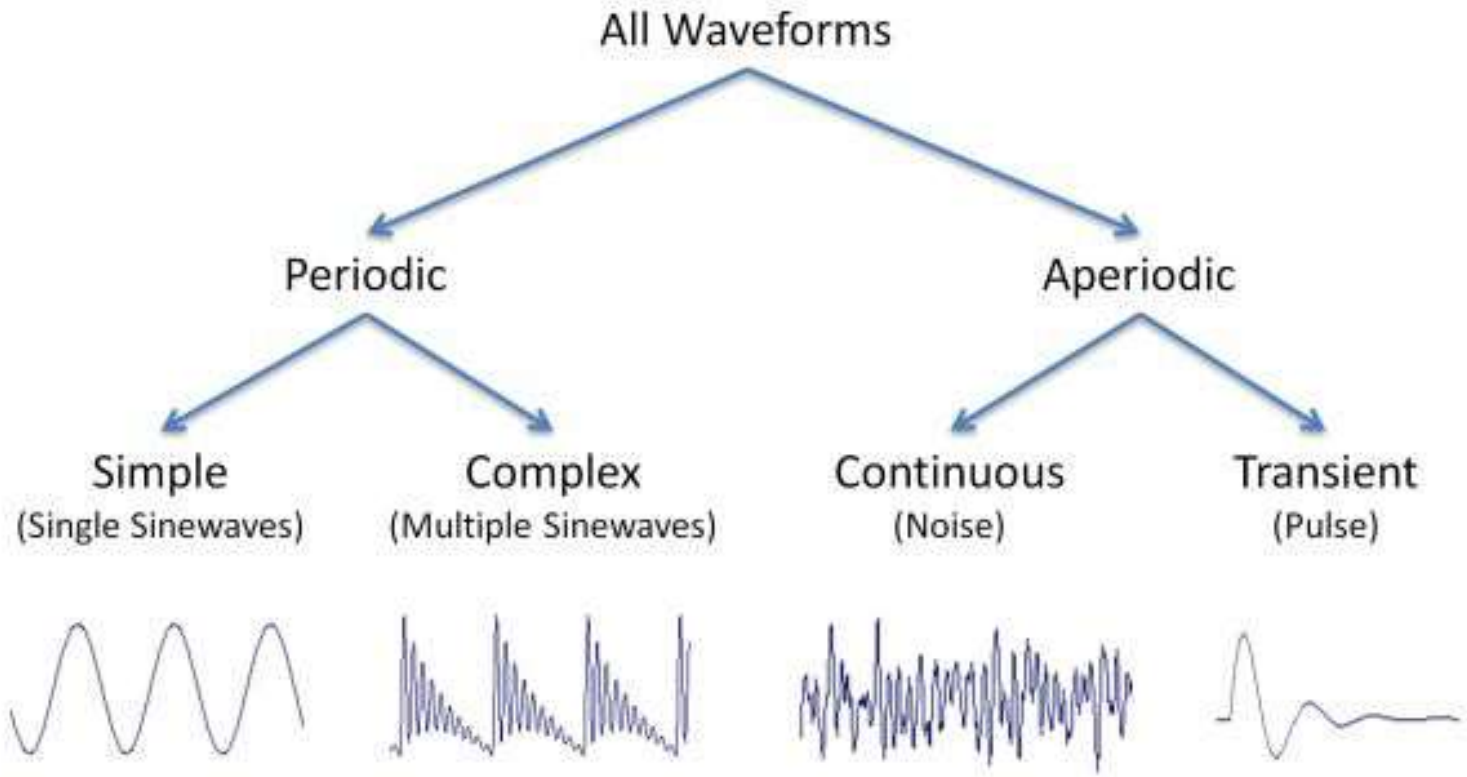


Cepstral Domain

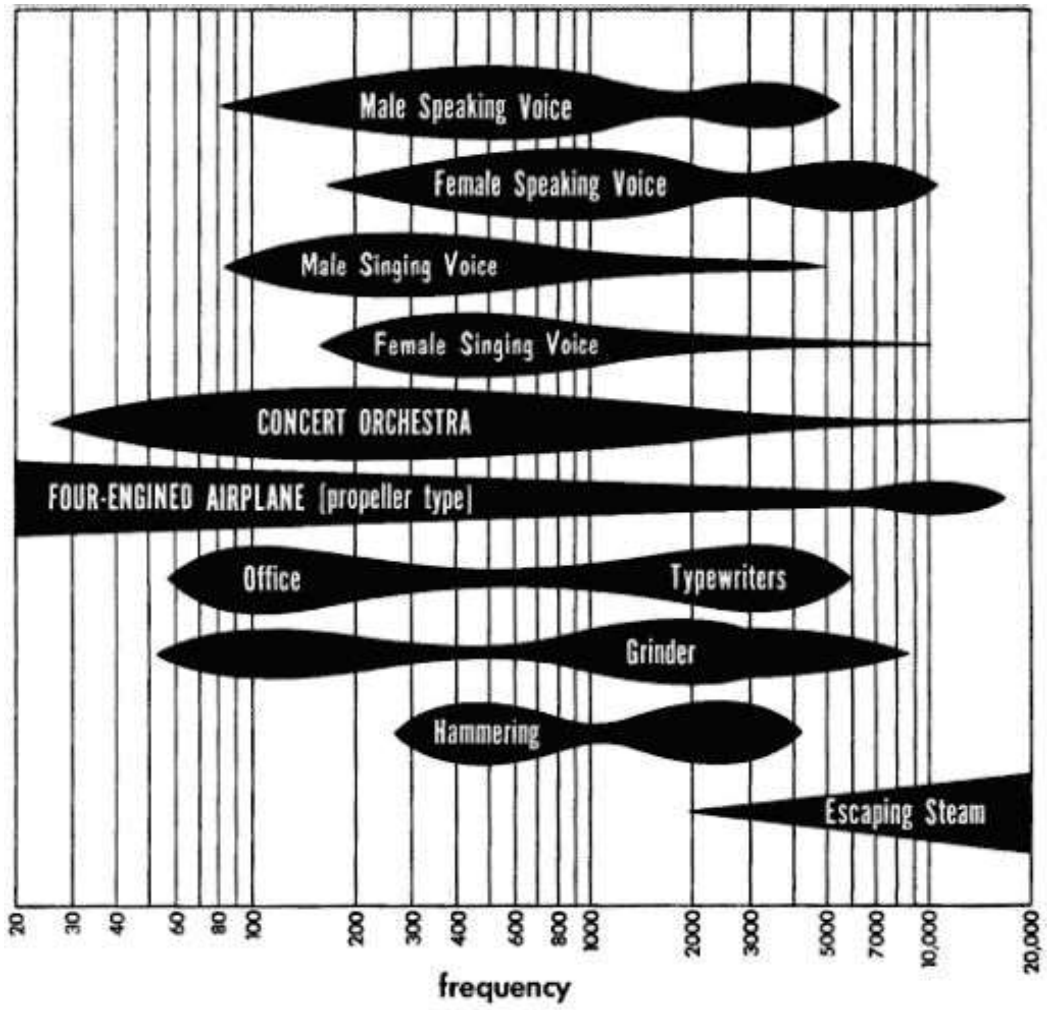


Audio signals - Types

Types of waveforms



Human hearing range



What is a Mel frequency and a Spectrogram?

The Mel scale

- Studies have shown that humans do not perceive frequencies on a linear scale.
- We are better at detecting differences in lower frequencies than higher frequencies.
- For example,
  - we can easily tell the difference between 500 and 1000 Hz,
  - but we will hardly be able to tell a difference between 10,000 and 10,500 Hz,
  - even though the distance between the two pairs are the same.
- In 1937, Stevens, Volkmann, and Newmann proposed a unit of pitch such that
  - equal distances in pitch sounded equally distant to the listener.
  - This is called the mel scale.
  - So, a mathematical operation on frequencies to convert them to the mel scale is performed.

Mel frequency

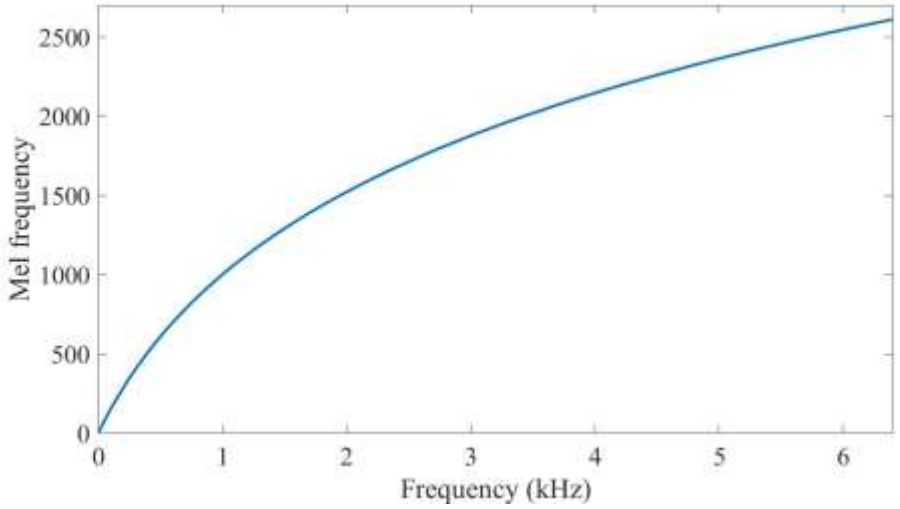
- Humans perceive frequency logarithmically

Ideally, we want features like:

- *Perceptually relevant amplitude representation*
- *Perceptually relevant frequency representation*

This is what Mel Spectrograms provide us:

- *Logarithmic scale*
- *Equal distances on the scale have same "perceptual" distance*
- *1000 Hz = 1000 Mel*

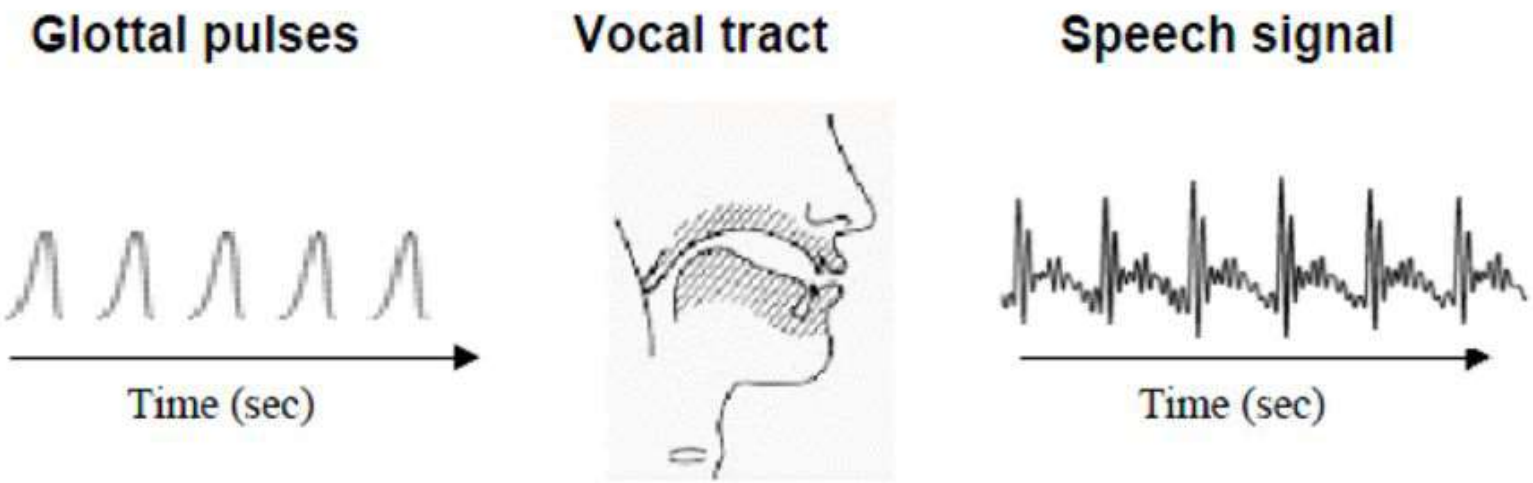


$$m = 2595 \log_{10}\left(1 + \frac{f}{700}\right)$$

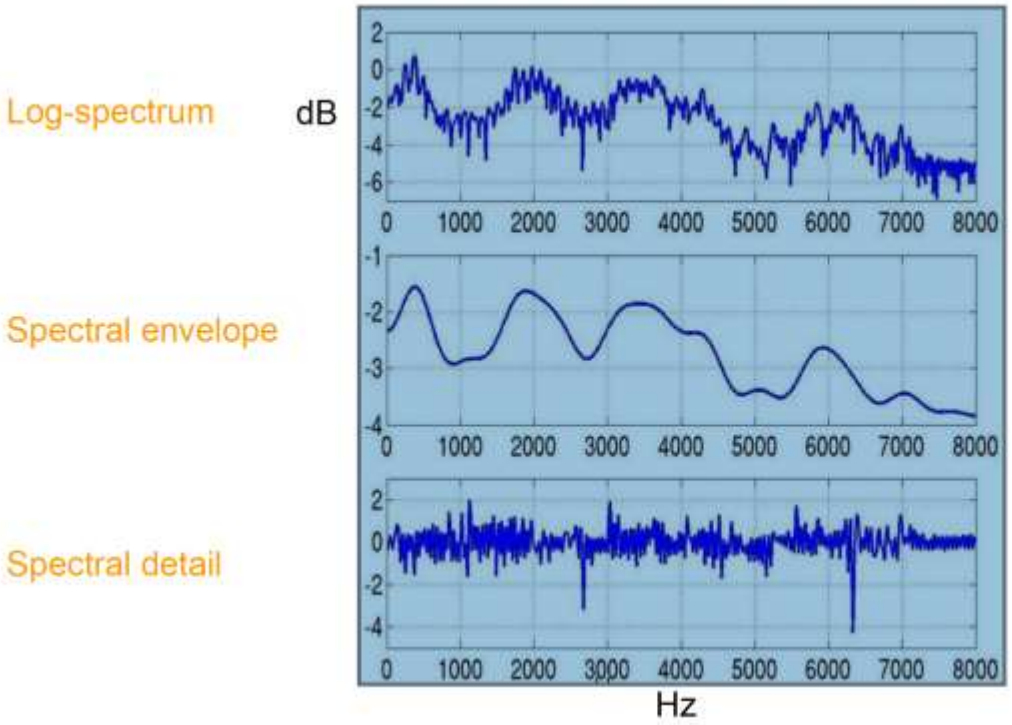
Mel-Frequency Cepstral Coefficients (MFCC)

- Developed while studying echoes in seismic signals (1960s)
- Audio feature of choice for speech recognition / identification (1970s)
- Music processing (2000s)

How speech is generated in humans

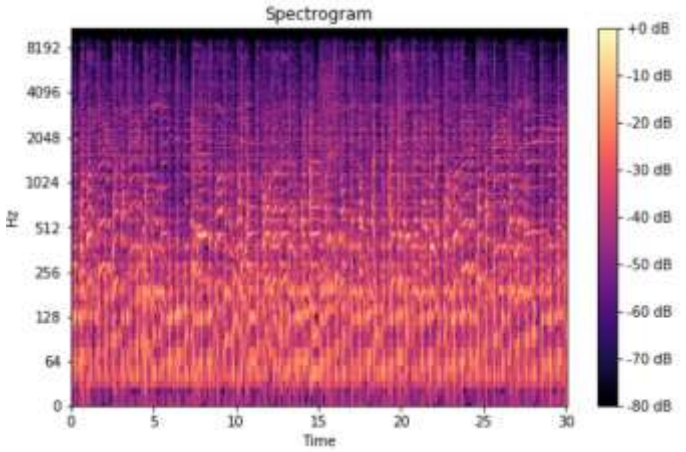


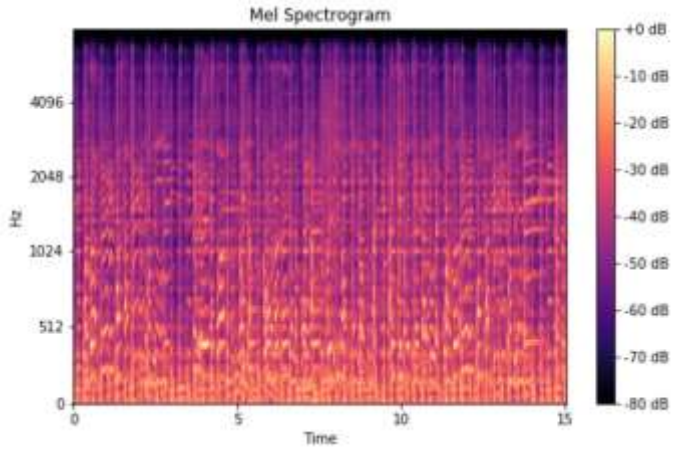
Getting phonemes out of the speech is what MFCCs do



- MFCCs represent
  - the overall shape or envelope of the sound spectrum,
  - effectively providing a compact and perceptually relevant feature set.
  - MFCCs are built on Mel frequencies
- By incorporating the Mel scale into the calculation of MFCCs,
  - the features become more relevant to human perception of sound,
  - making MFCCs a standard and effective choice for tasks like speech recognition & music

Mel Spectrogram – How it works!





- A **mel spectrogram** is a spectrogram where the frequencies are converted to the mel scale.
  - The audio signal is first broken into short, overlapping frames, and the Fourier Transform is applied to each frame to convert it into its frequency-domain representation.
  - The amplitude (loudness) of each frequency component is often converted to a logarithmic scale, such as decibels (dB).
  - Frequencies are then converted to the mel scale using a logarithmic formula.
- The result is a 2D graph where:
  - the x-axis represents time, the y-axis represents frequency on the mel scale, and
    - the color or brightness of the image indicates the amplitude (loudness) at that time and frequency.

So, what does all this mean for ML?

- By mapping frequencies to the mel scale, the model prioritizes features in the low-frequency range and groups high-frequency details into wider bands.
- Grouping frequencies into fewer mel bands significantly decreases the number of input data points for the machine learning algorithm.
- For tasks like speech recognition, the mel scale can enhance important speech features and increase the signal-to-noise ratio
- This enhances important speech characteristics for tasks like speech recognition and audio classification.
- It helps the model distinguish key characteristics from background noise.
- The mel spectrogram provides a feature representation that is more effective for audio-related machine learning tasks, often leading to better model performance compared to linear frequency representations.

MFCC based applications

Speech processing

- *Speech recognition*
- *Speaker recognition*

Music processing

- *Music genre classification*
- *Mood classification*
- *Automatic tagging*

Feature extraction in Audio

Approach to ML

Traditional ML OR Deep Learning

For traditional ML we can extract some low-level features like:

- *Amplitude envelope*
- *Root-mean square energy*
- *Zero crossing rate*
- *Band energy ratio*
- *Spectral centroid*
- *Spectral flux*
- *Spectral spread*
- *Spectral roll-off*

We could be building a model to recognize typical sounds – gunshot, toaster popping, jet engine, etc and some of these features may be enough:

- *Amplitude envelope*
- *Zero crossing rate*
- *Spectral flux*

Model building approaches

Amplitude envelope
Zero crossing rate
Spectral flux

Traditional
ML algorithm

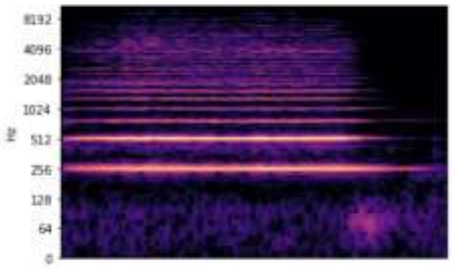
"car engine"

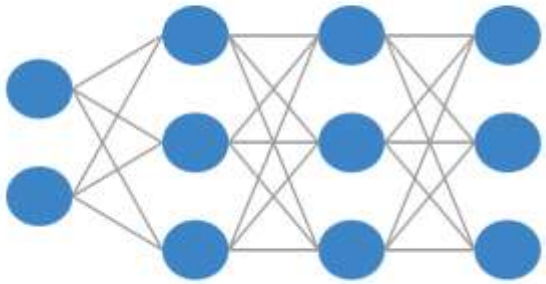
Traditional ML

feature engineering

Deep Learning

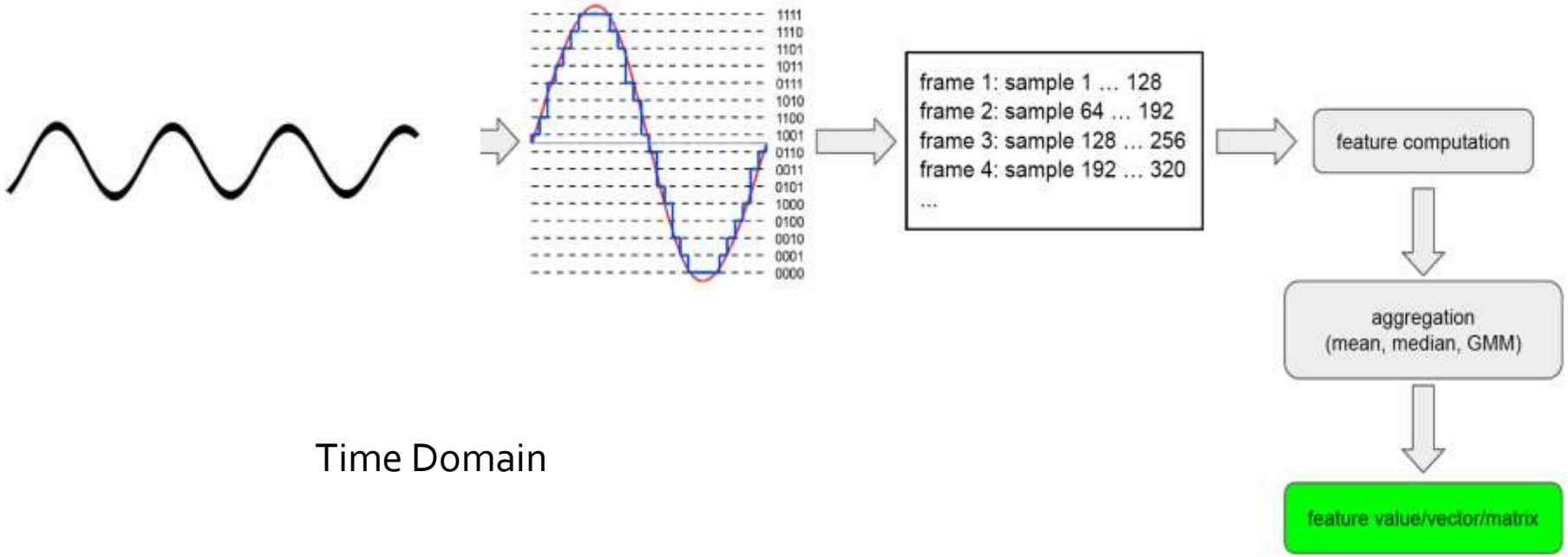
automatic feature extraction





"car engine"

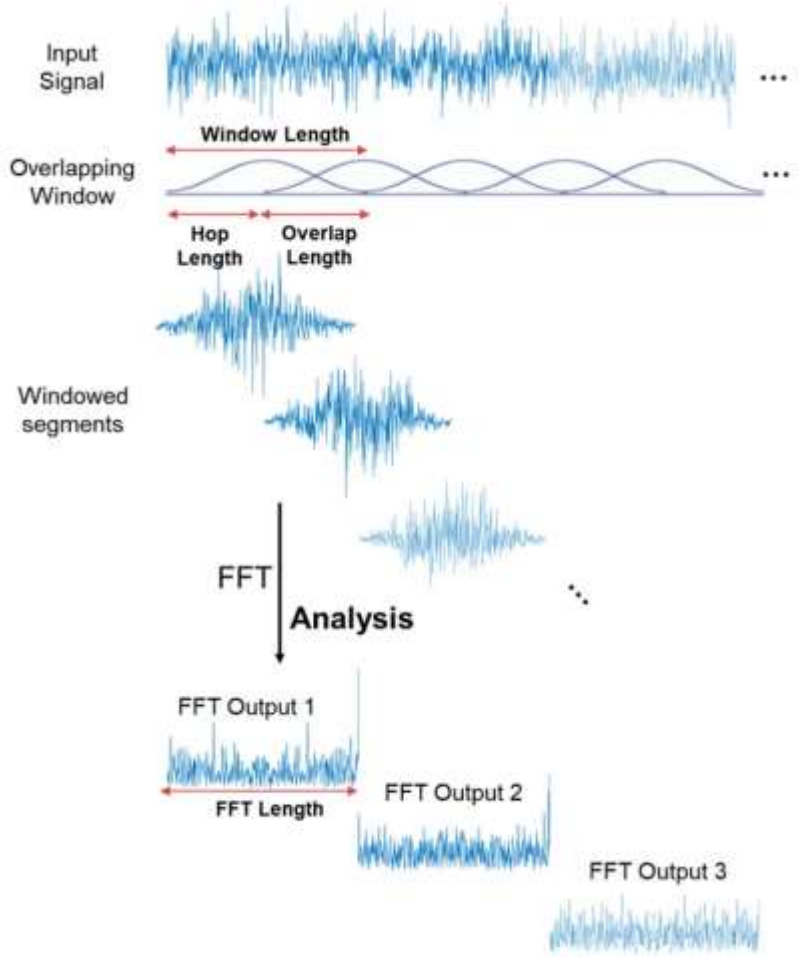
Time Domain features extraction



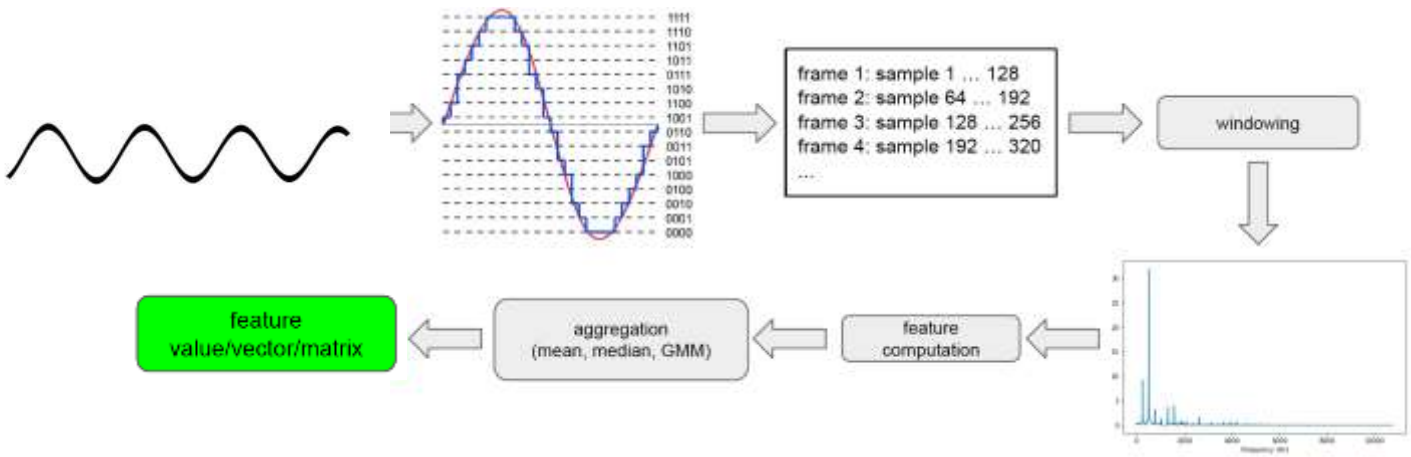
Time Domain

Frequency Domain features extraction

Short-time Fourier transform (STFT)



Frequency Domain



Feature extraction in Code using “librosa”

PyTorch’s Torchaudio package for Deep Learning

Torchaudio

- Deep learning technologies have boosted audio processing capabilities significantly in recent years.
- Torchaudio is a library for audio and signal processing with PyTorch.
- It provides audio processing functions like loading, pre-processing, and saving audio files.
- Selection of datasets and pre-trained models for audio classification, segmentation, and separation tasks, and tools for loading, manipulating, and enhancing audio data.
- Torchaudio's simple interface and high extensibility allow programmers to create unique audio processing pipelines with little code.
- As we have already seen, a spectrogram is a visual representation of the frequency content of an audio signal over time.
- torchaudio.transforms module has the functions required for various audio transformations like - "**torchaudio.transform.Spectrogram** "
- To generate a mel spectrogram, you can use the **MelSpectrogram** transformation from the torchaudio.transforms module.

Torchaudio basics in code

Example: Speech command classification algorithm.

ML for audio

Workflow for Audio-based Machine Learning:

- Data Loading and Preprocessing: Convert raw audio to a suitable format (e.g., spectrograms or MFCCs).
- Feature Engineering: Extract useful features for model input.
- Build models to learn patterns from the audio features. You can build from scratch or use pretrained models (recommended).
- Training and Evaluation: Train the model on the labeled audio data and evaluate its performance.

Audio projects you can do using Deep Learning ..1

Speech Recognition:
Implement a system that converts spoken language into text using models like DeepSpeech or Wav2Vec 2.0
Develop an application that can transcribe meetings or lectures in real-time.

Speaker Identification:
Create a system that identifies the speaker from a given piece of audio using speaker recognition models.
Develop a security system that uses voice biometrics for authentication.

Emotion Recognition:
Build an application that detects the emotion of the speaker from their voice using emotion recognition models.
This can be used in customer service to understand and improve customer experiences.

Music Genre Classification:
Develop a model that classifies music tracks into genres.
Create a recommendation system that suggests new music based on the user's listening habits.

Sound Event Detection:
Implement a system that recognizes different sound events (e.g., glass breaking, sirens, etc.) for security or surveillance applications.
Develop an app that provides notifications or insights for the hearing impaired.

Audio projects using Deep Learning ..2

Audio Denoising:

Build a tool that removes noise from audio clips using deep learning models, improving the clarity of recordings in noisy environments.

Speech Synthesis (Text-to-Speech):

Use models like Tacotron or WaveNet to convert text into natural-sounding speech.
Create an audiobook generator from text sources.

Language Translation:

Combine speech recognition with machine translation and text-to-speech models to develop a real-time audio translator.

Audio Tagging and Classification:

Build a system that tags and categorizes audio files (e.g., podcasts, lectures) based on their content for easier searching and sorting.

Voice Conversion:

Create a system that alters a speaker's voice to sound like a different person while maintaining the original speech content.

Music Separation and Instrument Recognition:

Develop an application that can separate individual instruments from a music track or identify the instruments being played.

Audio projects using Deep Learning ..3

Automatic Lyrics Alignment:
Implement a system that aligns lyrics with the sung audio in songs, useful for karaoke applications or music learning.

Bioacoustic Monitoring:
Use audio deep learning for ecological studies, such as identifying bird calls or monitoring rainforest sounds to assess biodiversity.

Enhancing Astrophysical Research:
Process and analyze data from radio telescopes or gravitational wave detectors using deep learning models.

Audio-based Health Diagnostics:
Develop tools for diagnosing health conditions through sound, such as analyzing cough sounds to detect respiratory diseases.

When embarking on these projects, you can leverage the wealth of pretrained models available in the PyTorch / HF ecosystem and fine-tune them on your specific dataset to achieve your project goals. This approach can save significant time and resources compared to training models from scratch.

Singular Value Decomposition(SVD):
Get a deeper understanding of the data

What are eigen vectors and eigen values?

- By definition a vector is a projection along a unit vector in the same direction to some magnitude.

$$A\vec{x} = \lambda\vec{x}$$

- A is a matrix of row or column vectors, the vector 'x' is called the eigen vector and the 'λ' is the eigen value.
- We can compute the eigen values of any matrix A from the characteristic equation :

$Ax = \lambda x \rightarrow Ax - \lambda x = o \rightarrow (A - I\lambda)x = o$

and therefore

"$\det(A - I\lambda) = o$"

What does SVD do?

- SVD is a linear algebra operation which allows us to decompose any matrix as a combination of matrices.
- This decomposition is represented as : **$A=U\Sigma V^T$**

  A is the original matrix (which can represent different types of data such as images, text, etc.).
  U is the left singular matrix.
  Σ is the diagonal matrix of singular values.
  V^T is the transpose of the right singular matrix.

- **Left Singular Matrix U:** Captures patterns across the rows of the matrix A. Each column of U corresponds to a direction in the row space of A
- For instance, in image data, these vectors might correspond to dominant spatial patterns, while in text data (like a term-document matrix), they might describe common themes or topics present across documents.
- **Right Singular Matrix V^T:** Captures patterns and structures related to the **columns** of A. Each column of V^T corresponds to a direction in the column space of A.
- In the case of images, these could represent different frequency patterns in the columns, while in a term-document matrix, they might describe how individual terms relate to the topics represented by U

Example 1 of SVD operation and what it tells us

- In this example we will use a simple 2 x 2 matrix
- We will see how to decompose it using SVD operator from the NumPy library.
- We will separate the 3 resulting matrices.
- We will then take 2 columns of the U matrix and project row 1 of A onto each of those columns
- We will then see what insight it gives us.

SVD’s role in Image analytics

- Singular Value Decomposition (SVD) plays an important role in image analytics, where it is used for tasks such as:
  - image compression, by decomposing the image into its key components and discarding less significant information
  - denoising, helps by decomposing an image into components and allowing you to filter out noisy components
  - feature extraction, used to decompose an image into singular vectors and values, with the left singular vectors capturing spatial patterns, and the right singular vectors capturing frequency components. These extracted features can be used in tasks like object recognition or scene classification.
  - dimensionality reduction of an image data by identifying the most important singular values, which capture the dominant structure in an image.
- In conjunction with Deep Learning, SVD can offer complementary benefits in specific cases, often providing insights into the underlying structure of an image or enabling efficient computation where neural networks might be overkill.

SVD in the context of audio

- A common approach is to transform the audio signal into the frequency domain using a spectrogram.
- This results in a matrix where rows represent time frames, and columns represent different frequencies.
- SVD decomposes this matrix into components that capture the essential information of the signal.
  - The left singular vectors (U) capture temporal patterns,
  - the singular values (Σ) represent the strength of those patterns,
  - and the right singular vectors (V^T) describe frequency components.
- In audio analytics, SVD can be used to reduce noise by discarding less important components (those with smaller singular values).
- For example, after performing SVD, you can reconstruct the matrix using only the largest singular values, thus retaining the dominant structures in the audio (such as voice or melody) while filtering out background noise.
- SVD is often used to extract features from audio data for further analysis, such as in speech recognition.

Let us look at some examples of SVD applications in image and audio

Exercise 8 – Audio classification

Exercise 8 instructions (Due by 15th midnight)

- You are provided a dataset which has 2 kinds of information:
    i. A folder full of audio files of 3 different sounds – dog bark, street music and drilling.
    ii. A csv file which has different kinds of details but the one's relevant for this exercise are filename and class with class ID.
- Your task is as follows:
    i. Build a classical ML classification model by extracting features out of the audio files.
    ii. Use SVD to pick significant features and therefore, perhaps, removing noise. (Numpy has a "truncated_svd" operator)
    iii. Use any ML classification algorithm.
- Save the model and build a separate function (in a separate notebook) to load the model and predict on the three audio samples provided.
- Answer the following questions:
    1. What features did you extract?
    2. What was your choice of truncation factor using SVD?
    3. What other preprocessing did you do on the data?
    4. How good were your predictions?

End of Lesson