Analog to Digital Conversion: Quantization

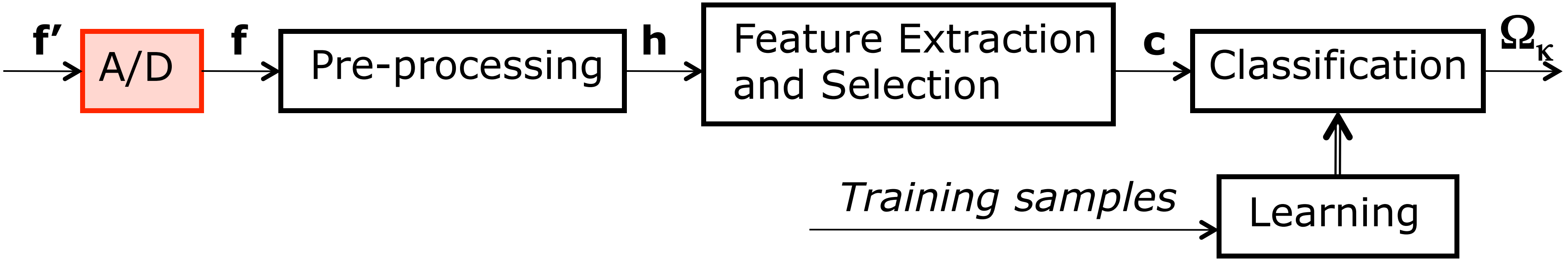
from the Perspective of Pattern Recognition

Dr. Elli Angelopoulou

Lehrstuhl für Mustererkennung (Informatik 5)

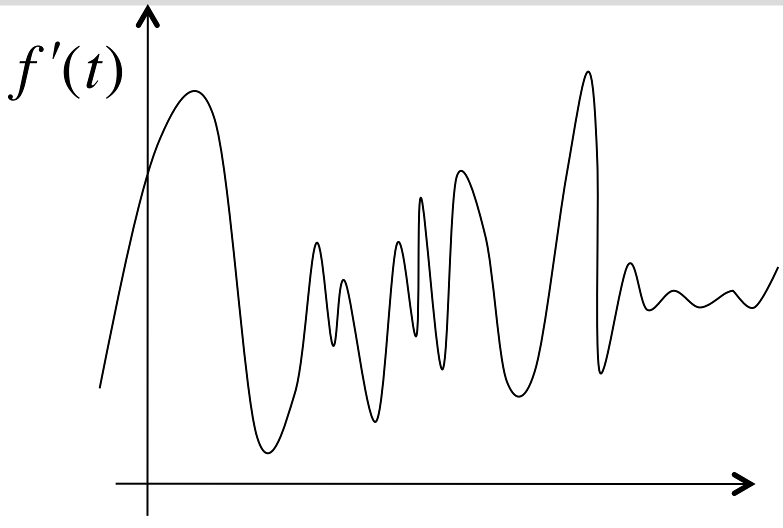
Friedrich-Alexander-Universität Erlangen-Nürnberg

Pattern Recognition Pipeline



- The goal of analog to digital conversion is to gather sensed data f' and change it to a representation that is amenable to further digital processing.

Need for A/D Conversion



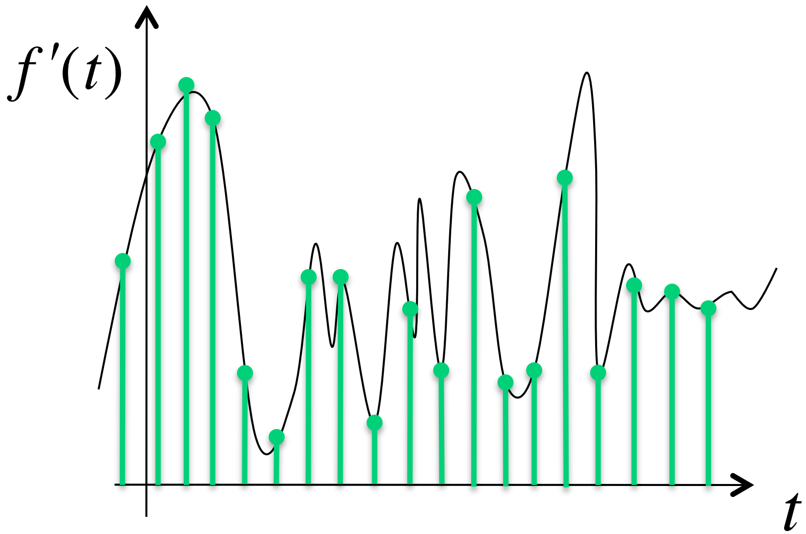
- Continuous range of t values
- Continuous range of amplitude $f'(t)$ values.
- We can only store a finite amount of values
- in a finite number of bits (discrete values).
- Goal: Find a discrete representation such that the original analog signal can be accurately reconstructed.

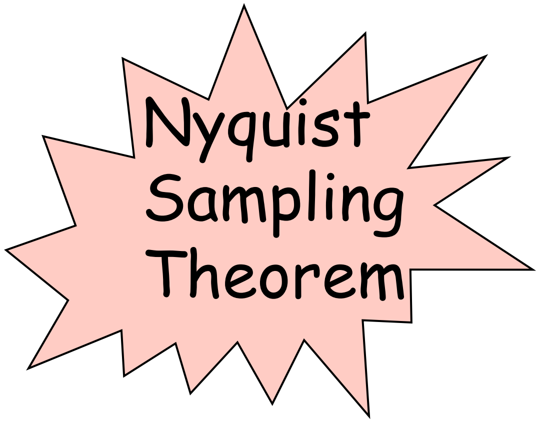
A/D Conversion Steps

- The A/D conversion (coding) involves:

1. measuring the amplitude values (or function values) at a finite number of positions:

   sampling,

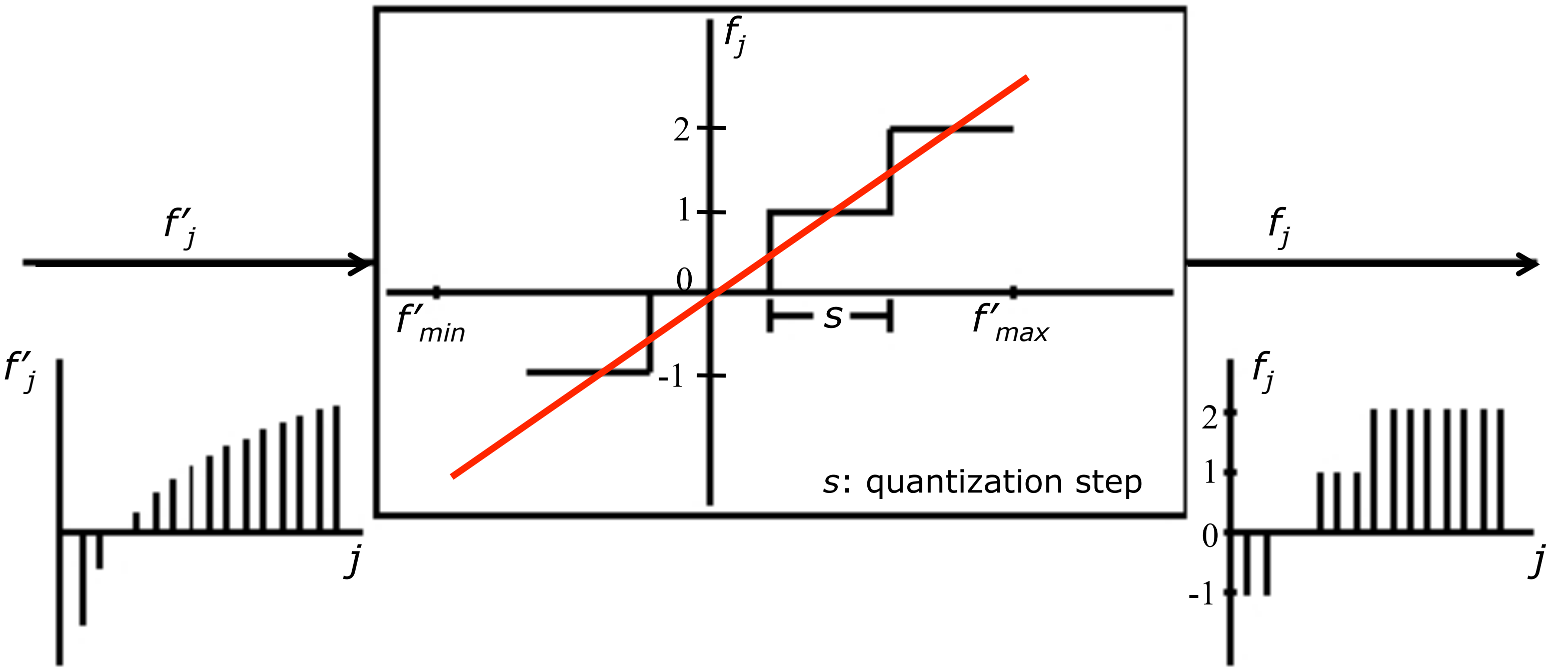




2. representing the amplitude values by a finite number of natural numbers:

   quantization

Quantization



- The number of quantization steps is defined by the number of bits we use to represent the value of the function.

Bits

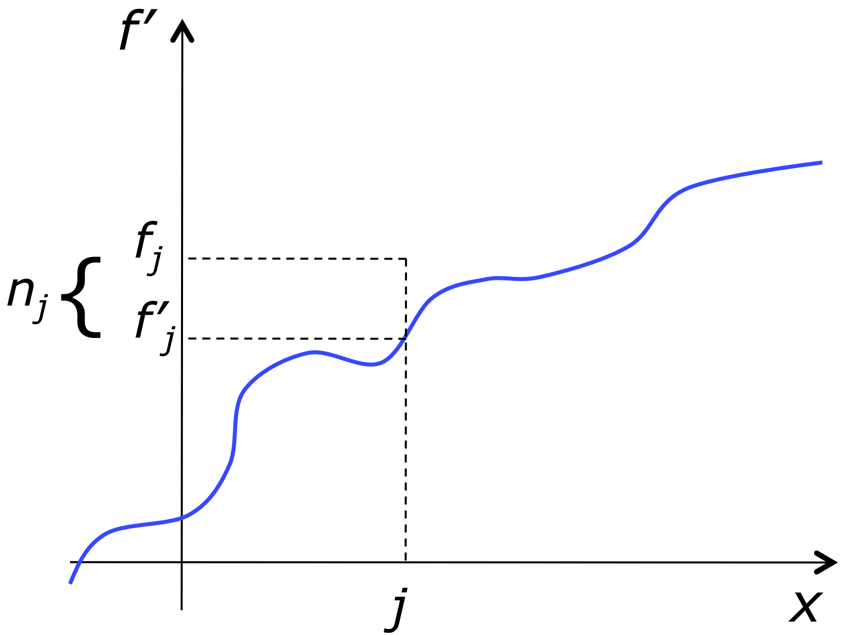
- Two key questions:
  1. How many bits?
  2. How do we use these bits?
- When we use B bits, we get 2^B quantized levels.
- Examples:
  - most intensity images: B = 8-12, 256 – 4,096 different gray values.
  - medical images: B = 10 - 16, 1024 – 65,536 different gray values.
  - most color images: B = 24-36, 8-12 for each color channel, at least 16 million colors.
- Typical data sizes for a 1024 x 1024 (1 MP) image:
  - at 8 bits => 1MB/img => a movie at 30fps creates 30MB/sec
  - at 12 bits => almost 1.6 MB/img => at 30 fps we get 47MB/sec
  - at 24 bits => 3.1 MB/img => at 30 fps we get 93MB/sec

  => a 5 minute movie needs 27GB.

Audio vs. Video Data Rates

Type	Specifications	Data Rate
Audio, understandable	1 channel, 8kHz @ 8 bits	64 kbit/sec
Audio, MPEG encoded	CD equivalence	384 kbit/sec
Audio, CD quality	2 channels, 44.1kHz @16 bits	1.4 Mbit/sec
Video, MPEG-2	640 × 480, 24 bits/pixel	0.42 MB/sec
Video, NTSC	640 × 480, 24 bits/pixel	27 MB/sec
Video, HDTV	1280 × 720, 24 bits/pixel	81 MB/sec

Quantization Error



- Quantization Error: The error we make when we approximate a real value f'_j by a discrete value f_j:

$$n_j = f'_j - f_j$$

Signal-to-Noise Ratio (SNR)

- There exists a standardized way of expressing the noise in a system or sensor that is associated with quantization. It is called the *Signal-to-Noise Ratio*.
- SNR is a general measure that is used for different types (sources) of noise.
- In Engineering SNR is a power ratio: $SNR = \dfrac{P_{signal}}{P_{noise}}$
- Within the context of pattern recognition, because of the uncertainty involved in the input signal, SNR is the ratio of the expected signal over the expected quantization noise.

$$SNR = \frac{E\{f'^2\}}{E\{n^2\}}$$

Signal-to-Noise Ratio (SNR) - continued

- The Signal-to-Noise Ratio is defined as:

$$SNR = r' = \frac{E\{f'^2\}}{E\{n^2\}}$$

where the quantization noise n is $n_j = f_j' - f_j$.

- The expected value $E\{\}$ is defined as:

$$E\{x\} = \int_{-\infty}^{\infty} xp(x)dx$$

where x is a random variable, and $p(x)$ is the probability density function (pdf) of x, which tells us how often different values of x occur.

- So, similar information on f' can guide us on how many bits to use.

SNR and logarithmic scale

- Because input signals can have a wide dynamic range, SNR is usually expressed in terms of the logarithmic decibel scale:

$$SNR_{dB} = r = 10\log_{10}\frac{E\{f'^2\}}{E\{n^2\}} = 10\log_{10}(r')$$

- Do we want a small or a large SNR? Why?

  Large is better.

  We want over 30dB SNR.

  42dB is considered “impressive” for SLR cameras.

  In computer vision we often use sensors with 60dB.

Does One Bit Make a Difference?

- Important question: How many bits should one use when quantizing a particular family of functions/ signals (i.e. medical images, or remote sensing data etc.)?
- Does one additional bit make a difference?
- Under certain assumptions (see next slide), the SNR is directly proportional to the number of bits used for quantization:

$$SNR_{db} = r = 6B - 7.2$$

- This means that 1 extra bit can increase the SNR by 6dB.

Assumptions

1. On average we have white noise.

$$E\{\vec{n}\} = 0 \text{ and } E\{\vec{n}\vec{n}^T\} = \sigma^2 I$$

2. We have a signal with $E\{f'\} = 0$.
3. The error (noise) is uniformly distributed.
4. The signal values lie in a limited range:

$$-4\sigma_{f'} \leq f' < 4\sigma_{f'}$$

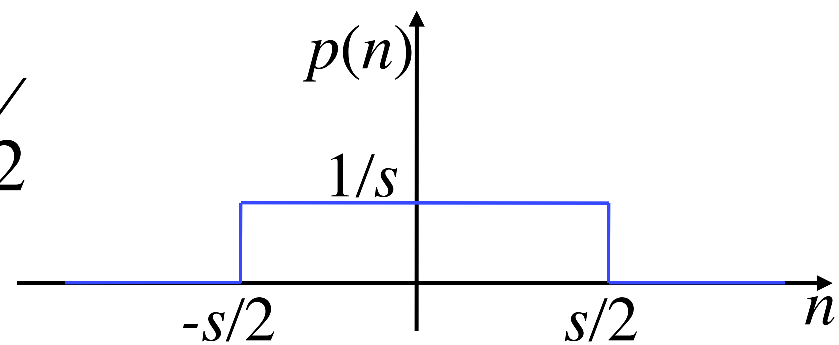
If we have a normal distribution, then
about 68% of the values lie within 1 σ of the mean,
about 95% of the values lie within 2 σ of the mean,
about 99.7% of the values lie within 3 σ of the mean,
about 99.99% of the values lie within 4 σ of the mean.

So **if** the values of f' follow a normal distribution, assumption 4 is reasonable.

Assumptions 1 and 3

- We have uniformly distributed white noise, $E\{n\}=0$. Let s be the quantization step (quantization interval). Then $p(n)$ will be of the form:

$$p(n)=\begin{cases} 1/s & \text{for } -\left(s/2\right)\le n\le s/2 \\ 0 & \text{otherwise} \end{cases}$$



The width of the pdf has to be s and centered around the value 0 (since $E\{n\}=0$), and the integral of the pdf has to sum up to 1 by definition.

SNR Denominator

- Recall that $SNR = r' = \dfrac{E\{f'^2\}}{E\{n^2\}}$
- What is $E\{n^2\}$?
- The definition of expected value is $E\{x\} = \int_{-\infty}^{\infty} xp(x)dx$.
- Thus, $E\{n^2\} = \int_{-\infty}^{\infty} n^2 p(n)dn$

$$= \int_{-s/2}^{s/2} n^2 p(n)dn = \frac{1}{s}\int_{-s/2}^{s/2} n^2 dn$$

$$= \frac{1}{s}\frac{1}{3}\left[n^3\right]_{-s/2}^{s/2} = \frac{1}{s}\frac{1}{3}\left(\frac{s^3}{8} - \left(-\frac{s^3}{8}\right)\right) = \frac{s^2}{12}$$

$$\boxed{E\{n^2\} = s^2/12} \qquad (1)$$

Assumption 2

- We have a signal with $E\{f'\} = 0$.
- According to the definition of standard deviation:

$$\sigma_{f'} = \sqrt{E\{f'^2\} - \left(E\{f'\}\right)^2}$$

- However, by assumption 2, we get

$$\sigma_{f'} = \sqrt{E\{f'^2\}}$$

$$\boxed{\sigma_{f'}^2 = E\{f'^2\}} \qquad (2)$$

Assumption 4

- The signal values lie in the range: $-4\sigma_{f'} \leq f' < 4\sigma_{f'}$
- So the length of the interval of the f' values is $8\sigma_{f'}$
- When we use B bits to store these $8\sigma_{f'}$ values, we have 2^B quantization levels.
- Assuming equidistant quantization, each quantization step, s, is

$$\boxed{s = \frac{8\sigma_{f'}}{2^B}} \qquad (3)$$

Assumption Combination

- So far, by exploiting the 4 assumptions we have shown:

$$\boxed{E\{n^2\} = s^2 / 12} \quad (1)$$

$$\boxed{\sigma_{f'}^2 = E\{f'^2\}} \quad (2)$$

$$\boxed{s = \frac{8\sigma_{f'}}{2^B}} \quad (3)$$

- From (1) and (3):

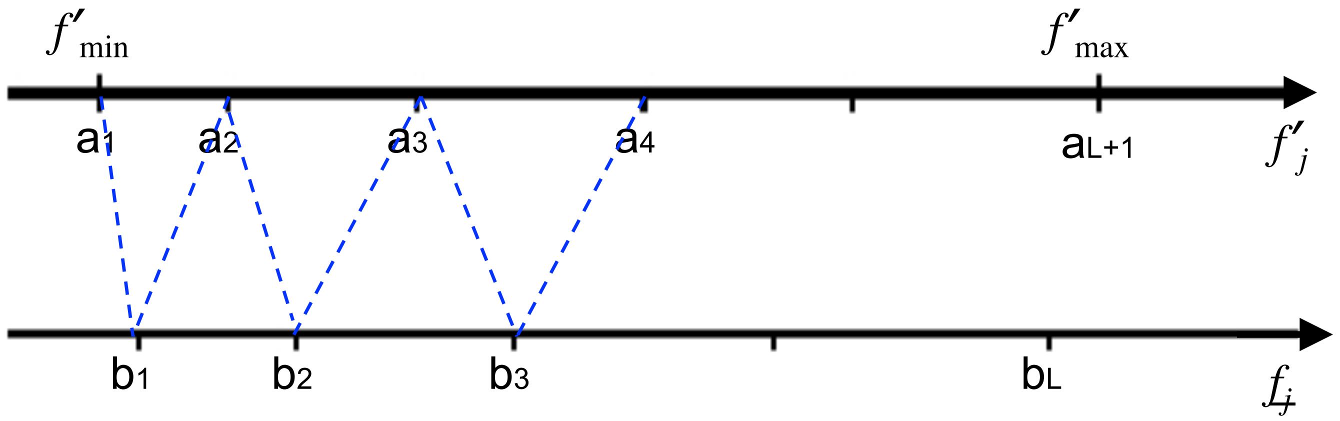
$$\boxed{E\{n^2\} = \frac{2^6 \sigma_{f'}^2}{12 \cdot 2^{2B}}} \quad (4)$$

- Recall that SNR is defined as $r' = \dfrac{E\{f'^2\}}{E\{n^2\}}$

$$r' = \sigma_{f'}^2 \bigg/ \left(\frac{2^6 \sigma_{f'}^2}{12 \cdot 2^{2B}} \right) = \frac{12 \cdot 2^{2B}}{2^6} = 12 \cdot 2^{2B-6} \quad \boxed{r = 10 \log_{10} r' = 6B - 7.27}$$

Mapping

- Using SNR as a criterion, we know how many bits to use, but how do we use them?
- To which discrete value do we map a continuous interval?



Good Mapping

- How can I tell whether my mapping is good?
- What is a possible objective function, a criterion to judge the quality of the mapping?
- Error measure (error that occurs when mapping f' to b_v)

$$\varepsilon = \sum_{v=1}^{L} \int_{a_v}^{a_{v+1}} \left(f' - b_v\right)^2 p(f')\,df'$$

- By weighing the error by the probability density of f', values that have a higher probability of occurring have a higher impact on the error term.
- The optimal quantization characteristics are defined by the values a_v , b_v which minimize the error ε.

Optimal Quantization Characteristics

- Optimal discrete value:

$$\partial\varepsilon / \partial b_v = \sum_{v=1}^{L} \int_{a_v}^{a_{v+1}} 2\left(f' - b_v\right) p(f')\,df' = 0 \qquad v = 1,2,\ldots,L$$

$$\int_{a_v}^{a_{v+1}} f' p(f')\,df' = b_v \int_{a_v}^{a_{v+1}} p(f')\,df' \Leftrightarrow b_v = \frac{\int_{a_v}^{a_{v+1}} f' p(f')\,df'}{\int_{a_v}^{a_{v+1}} p(f')\,df'}$$

- Optimal threshold level:

$$\partial\varepsilon / \partial a_v = \partial\left[\sum_{v=1}^{L} \int_{a_v}^{a_{v+1}} \left(f' - b_v\right)^2 p(f')\,df'\right] \Big/ \partial a_v = 0$$

Use the middle value as a threshold

$$\left(a_v - b_{v-1}\right)^2 p(a_v) - \left(a_v - b_v\right)^2 p(a_v) = 0 \Leftrightarrow a_v = \frac{b_v + b_{v-1}}{2}$$

Pulse Code Modulation

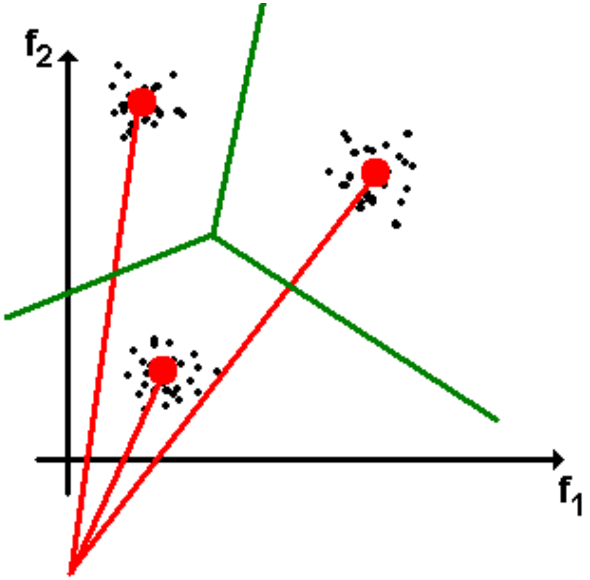
- A linear quantization characteristic function (with equally spaced quantization levels) is an optimal quantization if and only if the signal amplitudes are equally distributed.
- Coding using the methods introduced so far is called Pulse Code Modulation.
- Other coding methods, depending on the application are:
  - Coding with a minimal number of bits
  - Error detection and correction
  - Run-length encoding
  - Chain code

Vector Quantization

- So far, we have considered the quantization of real valued functions, i.e. $f \in R$.
- There exist signals where we have to deal with vector valued functions, $\vec{f} \in R^N$ (e.g. color images with RGB values).
- The quantization of vectors to discrete vectors is called *vector quantization*.
- **Vector quantization** is the process of mapping N-dimensional vectors in the vector space R^N into a finite set of vectors $Y = \{\vec{y}_i | i = 1 \cdots k\}$, where $k < N$.
- Each vector $\vec{y}_i$ is called a **code vector** or a **codeword**.
- The set of all the codewords, Y, is called a **codebook**.

Codebook Design

- There exist many vector quantization methods.
- We are just going to present one method which is based on mean values.
- Another one is based on computing nearest neighbor regions, aka Voronoi regions.



Using the Mean Vectors

- For each cluster in the training data compute the mean vector $\vec{\mu}_i$.
- Each mean vector $\vec{\mu}_i$ becomes the code vector or codeword, $\vec{y}_i$.
- All the mean vectors define the so-called code book, Y.
- Given an arbitrary input vector $\vec{f}'_j$ find the nearest code vector $\vec{y}_u$, s.t. $u = \min_i d(\vec{f}'_j, \vec{y}_i)$.
- Store the offset to the closest mean $\vec{y}_u$. There is a finite number of bits that can be used for the offset.
- Use your favorite distance metric, e.g. Euclidean, Manhattan, etc. We often use the Euclidean distance.

Computing the Codebook

- k-means algorithm
- k:# of code vectors
- Input: M data vectors $\vec{f}_1, \vec{f}_2, \ldots, \vec{f}_M, \vec{f}_i \in R^N$

1. Randomly assign the vectors $\vec{f}_1, \vec{f}_2, \ldots, \vec{f}_M$ to k clusters.
2. Compute the mean vector $\vec{\mu}_i$ for each cluster.
3. Reassign each vector $\vec{f}_1, \vec{f}_2, \ldots, \vec{f}_M$ to the cluster with the nearest mean vector $\vec{\mu}_i$.
4. Repeat 2. and 3. until no further changes occur

- Output: code book

Linde-Buzo-Gray Algorithm

- The Linde-Buzo-Gray (LBG) algorithm is a widely-used vector quantization algorithm which is very similar to the k-means algorithm.
- Main idea. Start with a single code vector. At each iteration, each code vector is split into two new vectors.

1. Initial state: compute the mean of the training data.
2. Initial estimation #1: code book of size 2.
3. Final estimation for code book of size 2, after training data reassignment.
4. Initial estimation #2: code book of size 4.
5. Final estimation for code book of size 4, after training data reassignment. …