Neural Networks – Pre-Processing and Feature Extraction

Mohamed Krini

Christian-Albrechts-Universität zu Kiel
Faculty of Engineering
Institute of Electrical and Information Engineering
Digital Signal Processing and System Theory

Entire Semester

- Introduction
- ***Pre-Processing and Feature Extraction***
- Threshold Logic Units - Single Perceptrons
- Multilayer Perceptrons
- Training Multilayer Perceptrons
- Radial Basis Function Networks
- Learning Vector Quantization
- Kohonen Self-Organizing Maps
- Hopfield and Recurrent Networks

Contents of this Part

Pre-Processing and Feature Extraction

- Introduction
- Input Normalization
- Feature Extraction for Speech Recognition
  - Analysis Filterbank
  - Mel Filterbank
  - Mel-filtered Cepstral Coefficients
  - Temporal Features
- Dimension Reduction

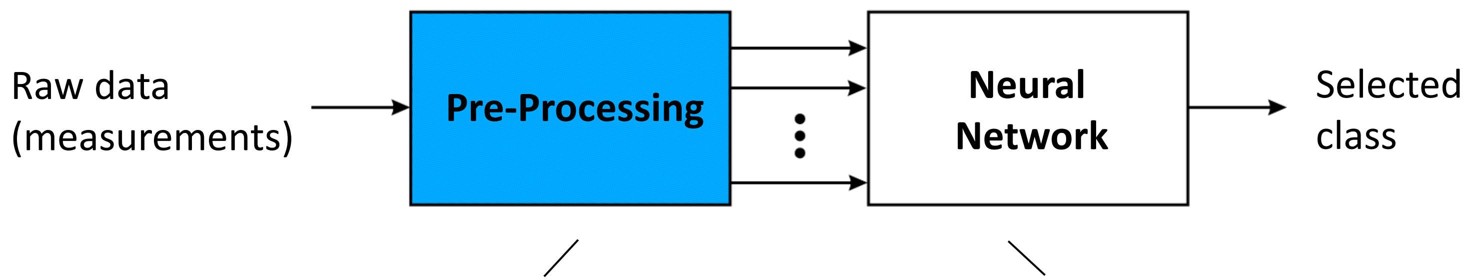
Introduction (1/3)

Classical Model of Pattern Recognition:

Task: Assign one of several classes to a set of measurements

Input: A vector of measurements (patterns)

Output: Number of the best class



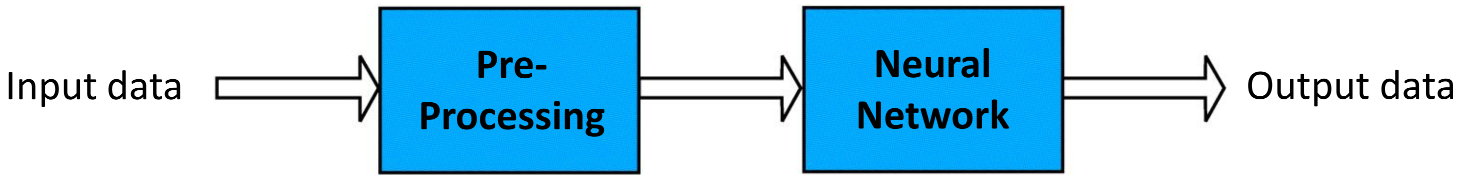
Feature Extraction:
Extracts features from raw data (e.g. speech, image, weather data) that should ease the task of classification.

Classification:
The input features are assigned to one of K classes (e.g. speech signal is classified into phonemes and/or words).

Introduction (2/3)

Definitions and Properties:

- For most applications it is necessary first to ***transform*** the ***data*** into some new representation before presented to a network:



- The choice of ***pre-processing*** will be one of the most ***significant factors*** for determining the performance of the final system.

- In the simplest case the pre-processing performs a linear transformation of the input data (e.g. ***input normalization***).

Introduction (3/3)

Definitions and Properties (continued):

- One of the most important forms of pre-processing involves a ***reduction*** in the ***dimensionality*** of the ***input data.***

- In most situations a reduction in the dimensionality will result in ***loss*** of ***information.***

  ➡ ***Goal***: Preserve as much of relevant information as possible.

- A network with fewer inputs has fewer adaptive parameters to be determined, leading to a network with better ***generalization*** properties.

- In addition a network with fewer weights may be ***faster*** to train.

- The transformed inputs are often called ***features*** and the process of generating them is called ***feature extraction*****.**

Input Normalization

Input Normalization

Input Normalization (1/3)

Properties:

- Simple linear rescaling of the input variables is one of the ***most common forms of pre-processing***.
- Often useful if different variables have typical values which differ significantly.
- Input normalization ***avoids scaling problems***.
- The rescaling is useful for radial basis function networks and multilayer networks:
  - ➡ If variation in one parameter is small with respect to the others it will contribute very little to distance measures.
- Each of the input variables is treated independently for linear rescaling.

Input Normalization (2/3)

Procedure for Input Normalization:

- We calculate the ***mean*** and the ***variance*** with respect to the training set:

Training set consists of N-dimensional input vectors:

Number of training patterns

$$\boldsymbol{x}^{(m)} = \left[x_0^{(m)}, x_1^{(m)}, ..., x_{N-1}^{(m)}\right]^{\mathrm{T}}, \quad \text{with} \quad m \in \{0, ..., M-1\}.$$

Mean:

$$\mu_i = \frac{1}{M} \sum_{m=0}^{M-1} x_i^{(m)}.$$

Variance:

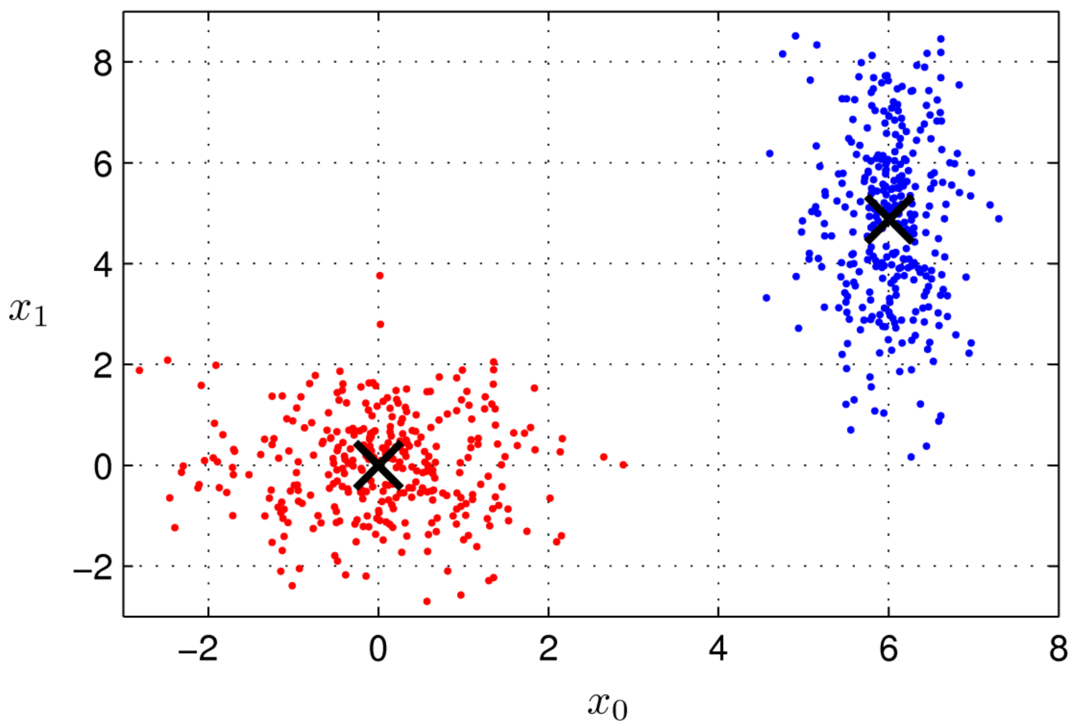
$$\sigma_i^2 = \frac{1}{M} \sum_{m=0}^{M-1} \left(x_i^{(m)} - \mu_i\right)^2.$$

- Normalize the input vectors to ***expected value 0*** and ***standard deviation 1***:

$$\tilde{x}_i^{(m)} = \frac{x_i^{(m)} - \mu_i}{\sigma_i}.$$

Input Normalization (3/3)

Experiment:



- Large data set consists of two dimensional feature vectors.
- Blue contour line represents feature vectors without normalization.
- Red contour line shows the input vectors after normalization.
- Mean value: ×

Feature Extraction

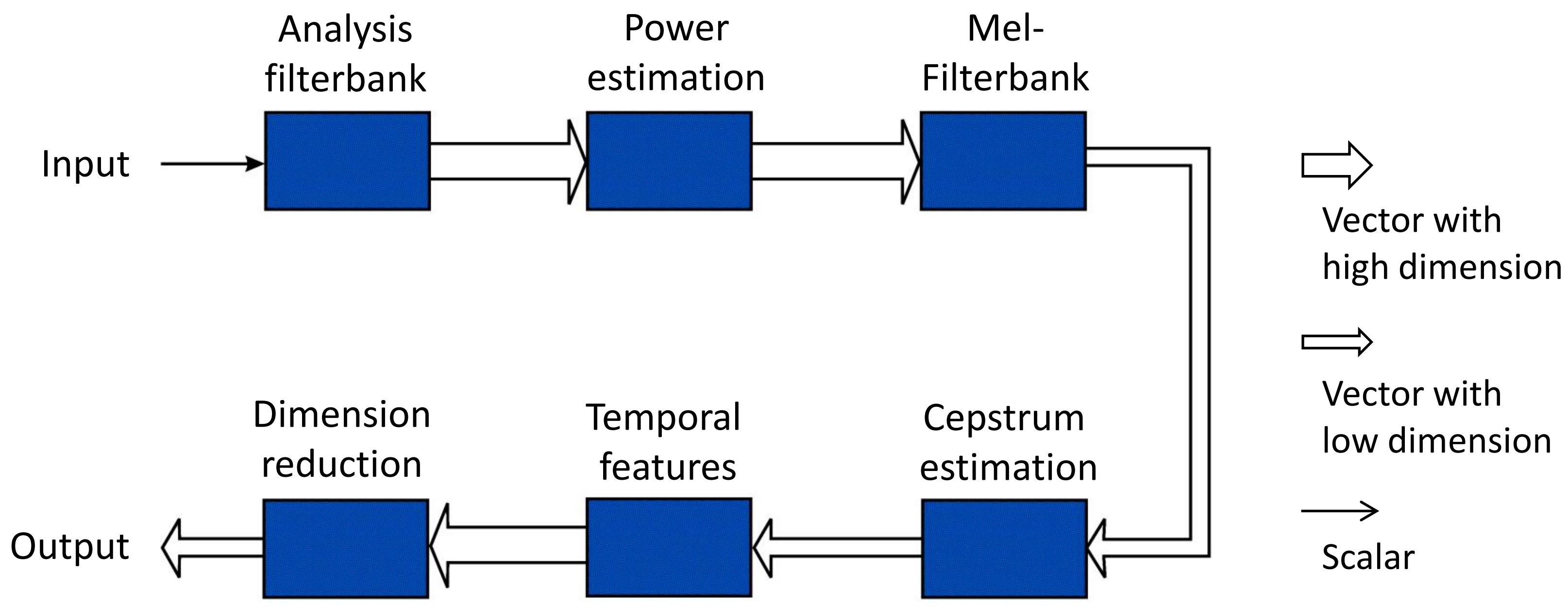
Feature Extraction:

- *Application for Speech Recognition*

Feature Extraction for Speech Recognition

Overview:

- As an application example we consider the pre-processing and feature extraction for ***speech recognition.***



Feature Extraction – Analysis Filterbank (1/3)

Analysis Filterbank:

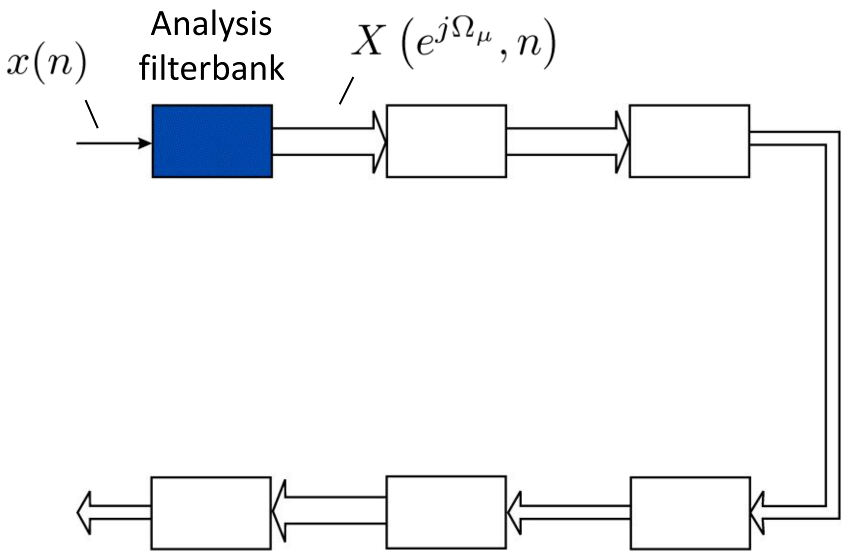
- Input signal is first ***segmented*** into blocks of appropriate size:

$$\tilde{\boldsymbol{x}}(n) = \big[x(n),\, x(n-1),\, \ldots,\, x(n-N+1)\big]^{\mathrm{T}}.$$

- Usually adjacent input segments are ***over-lapped*** (modeled by a ***subsampling*** factor R):

$$\boldsymbol{x}(nR) \;=\; \tilde{\boldsymbol{x}}(nR)\,.$$

- By applying a window function h_k and computing the DFT, the ***short-term spectrum*** results:



$$X\left(e^{j\Omega_\mu}, n\right) \;=\; \sum_{k=0}^{N-1} x(nR-k)\, h_k\, e^{-j\Omega_\mu k}, \quad \text{with } \mu \in \{0,\, \ldots,\, N-1\}\,.$$

Feature Extraction – Analysis Filterbank (2/3)

Principle:

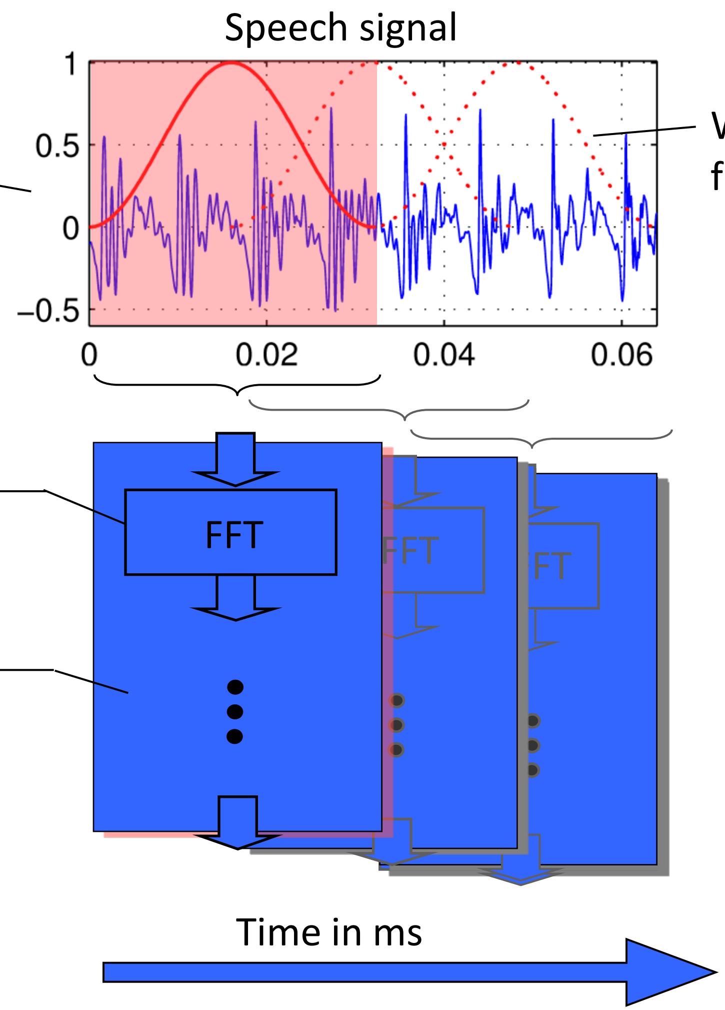
Segmentation, overlapping and windowing

Short-term spectrum computation

Further processing

Result:

Feature vector each 16 ms



Used parameters:

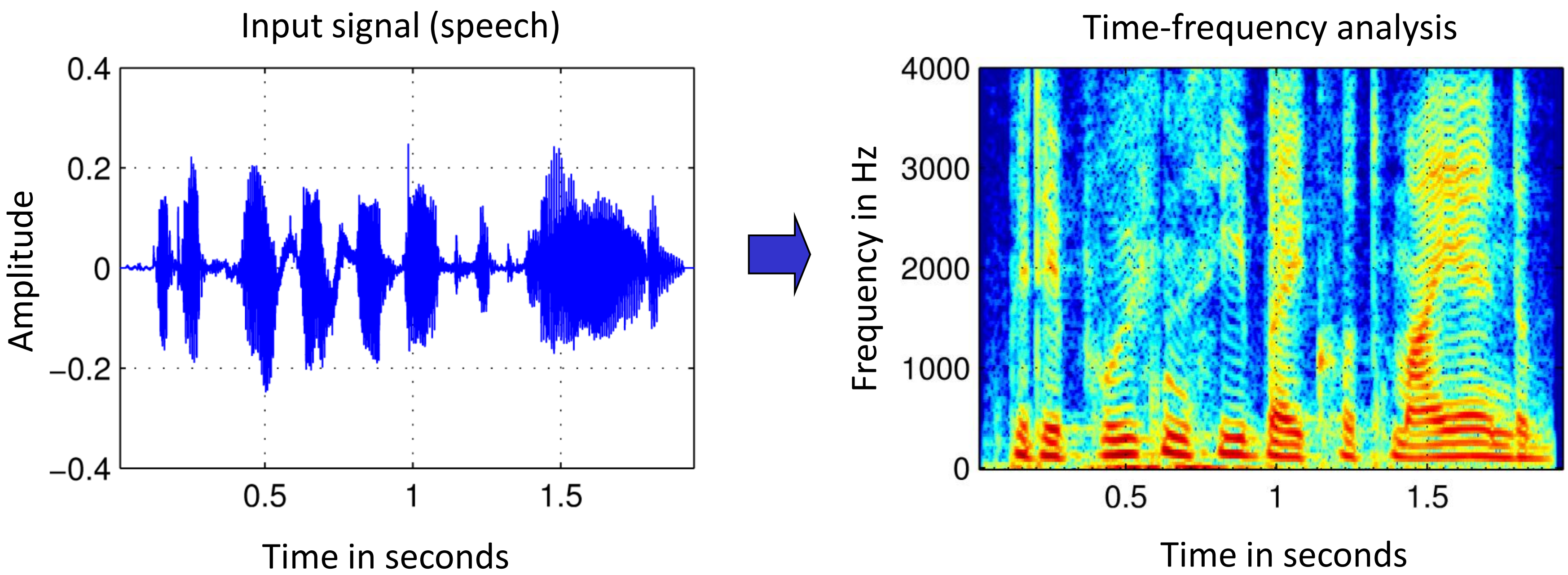
Block length:
$$N = 512$$

Subsampling rate:
$$R = 256$$

Sampling rate:
$$f_s = 16 \text{ kHz}$$

Feature Extraction – Analysis Filterbank (3/3)

Example of Time-Frequency Representation:



Feature Extraction – Power Estimation

Power Estimation:

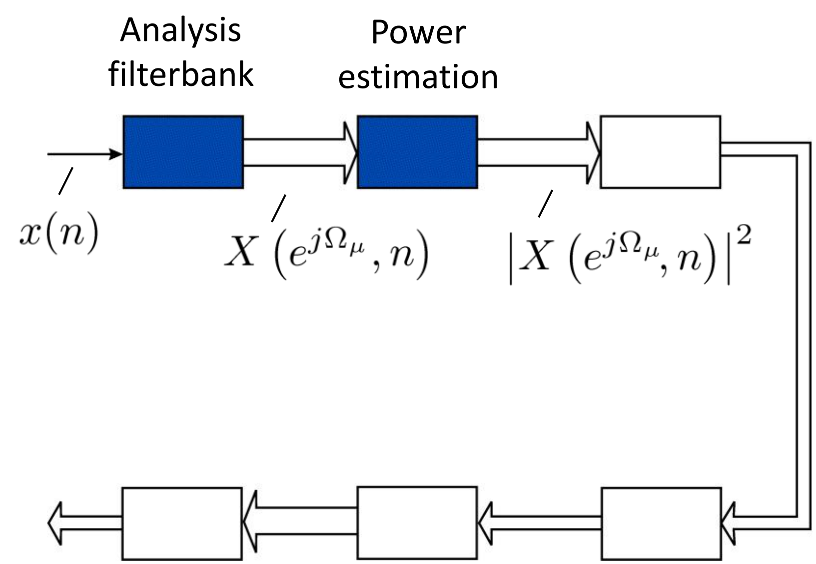
- Absolute square of the input spectrum:

$$\left|X\left(e^{j\Omega_\mu}, n\right)\right|^2 = \Re^2\left\{X\left(e^{j\Omega_\mu}, n\right)\right\} + \Im^2\left\{X\left(e^{j\Omega_\mu}, n\right)\right\} .$$

- ***Approximation*** of the absolute value (low computational cost needed, low dynamic):

$$\left|X\left(e^{j\Omega_\mu}, n\right)\right| \approx K\left|\Re\left\{X\left(e^{j\Omega_\mu}, n\right)\right\}\right| + K\left|\Im\left\{X\left(e^{j\Omega_\mu}, n\right)\right\}\right| .$$

- ***Amplitude*** is much ***more important than*** the ***phase*** (phase is discarded), the results are real numbers.



Feature Extraction – Mel Filterbank (1/5)

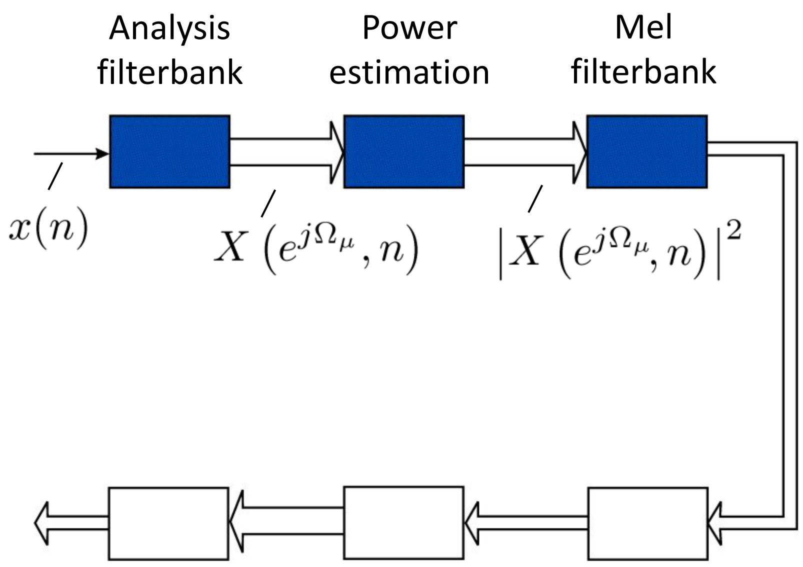
Mel Filterbank:

- A set of ***triangular filters*** is used to approximate the frequency resolution of the human ear.

- Approximated formula to compute the frequency (or pitch) in Mel for a given frequency *f* in Hz:

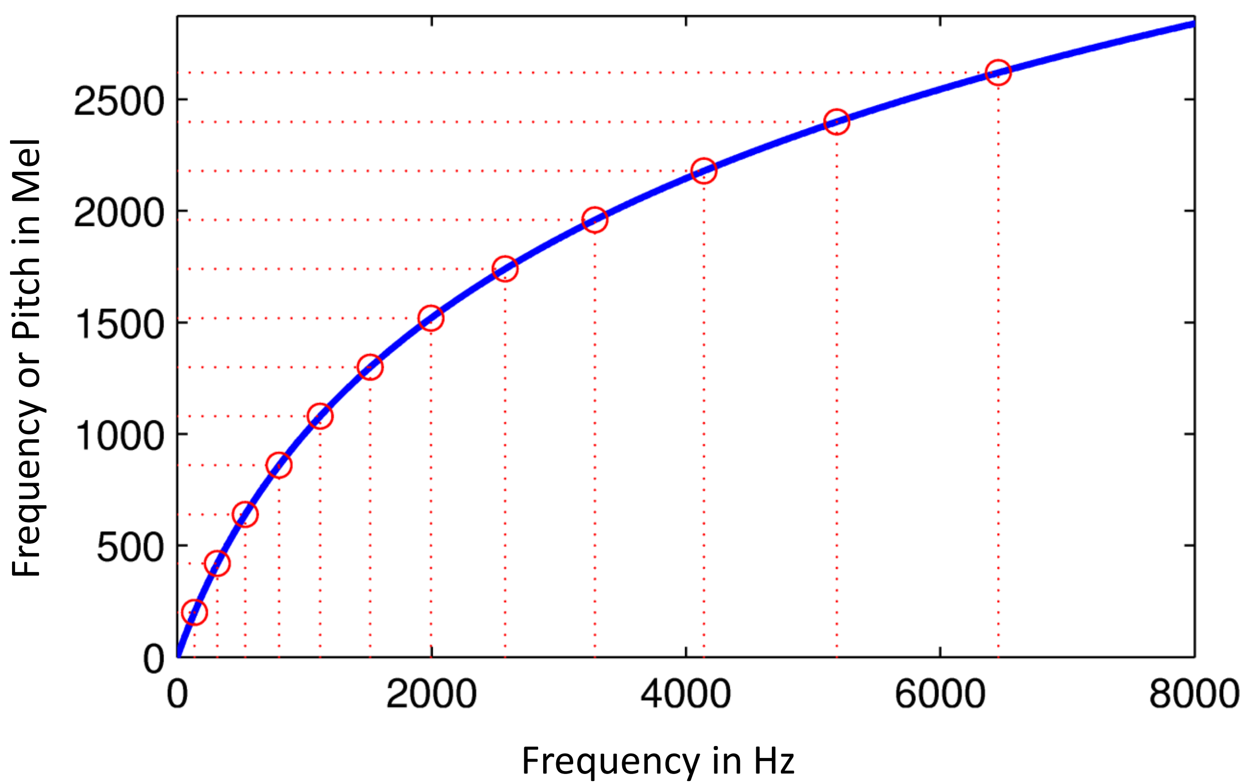
$$m = 2595\,\mathrm{Mel}\,\log_{10}\left(1 + \frac{f}{700\,\mathrm{Hz}}\right).$$

- The Mel-frequency scale is linear up to 1 kHz and logarithmic thereafter.



Feature Extraction – Mel Filterbank (2/5)

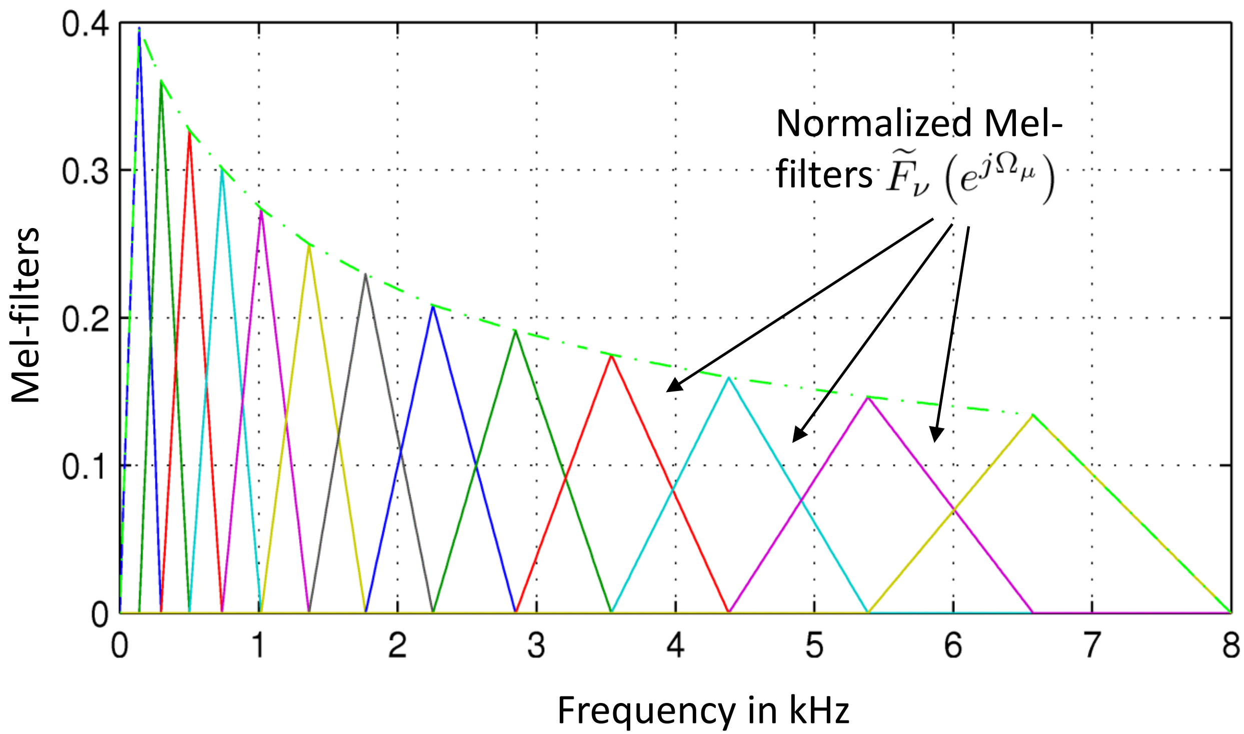
Mel-Scale Characteristic:



- The Mel-frequency scale shows a ***logarithmic*** behavior.
- Mel-frequency axis (Mel scale) divided into 13 ***equispaced*** bands in this example.
- A set of ***overlapping Mel filters*** is designed such that the center frequencies of the filters are equidistant on the Mel scale.

Feature Extraction – Mel Filterbank (3/5)

Overlapping Mel-Filters:



- A set of ***triangular*** filter banks is used to approximate the frequency resolution of the human ear.
- Typically 15 up to 30 Mel filters are applied for a sampling rate range from 8 to 16 kHz.
- The power of the spectrum is mapped onto the Mel scale using triangular ***overlapping*** windows.

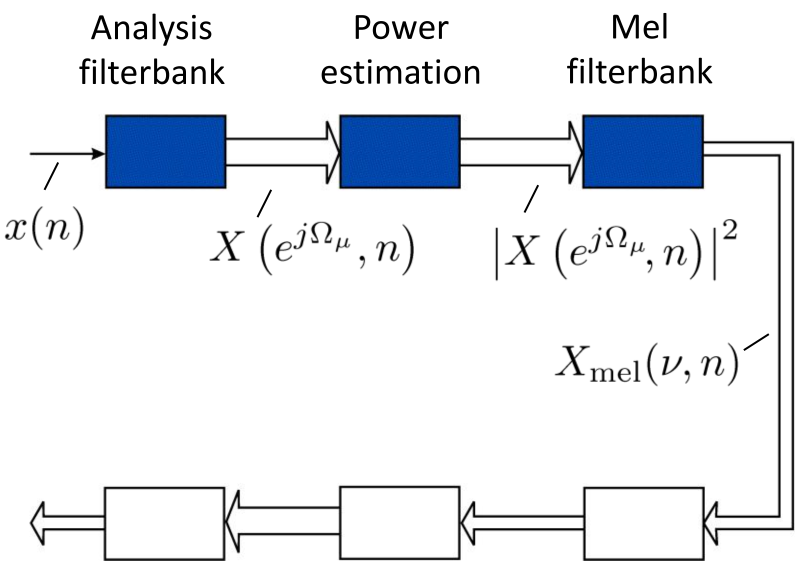
Feature Extraction – Mel Filterbank (4/5)

Power Estimation in Mel Domain:

- The ***Mel power spectrum*** is determined by:

$$X_{\text{mel}}(\nu, n) = \frac{\sum\limits_{\mu=0}^{N-1} F_\nu\left(e^{j\Omega_\mu}\right)\left|X\left(e^{j\Omega_\mu}, n\right)\right|^2}{\sum\limits_{\mu=0}^{N-1} F_\nu\left(e^{j\Omega_\mu}\right)},$$

with $\nu \in \{0, \ldots, M-1\}$.

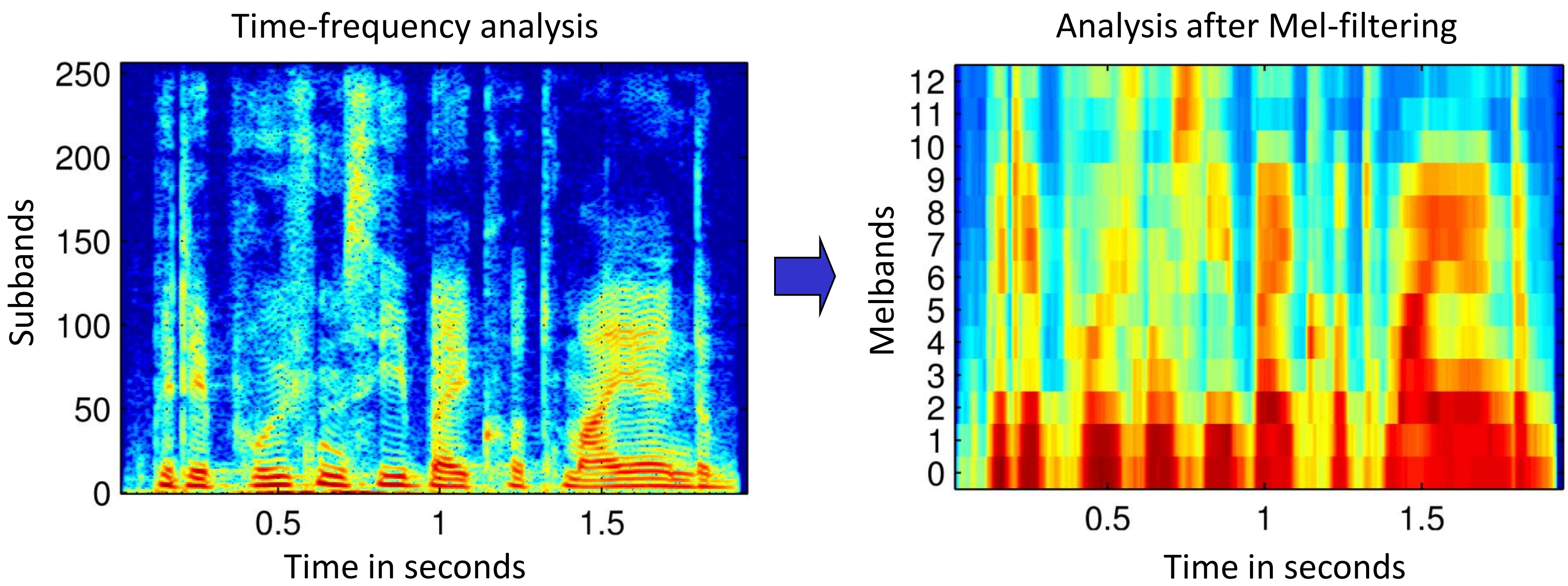


- Mel filters are ***normalized*** to guarantee the same power at the output in case of white noise excitation.

- The outputs are ***real-valued*** and ***reduced in dimension*** compared to the input.

Feature Extraction – Mel Filterbank (5/5)

Example:



- The dimension of the input features is reduced after transformation into the Mel-domain.

C | A | U
Christian-Albrechts-Universität zu Kiel

Feature Extraction – Cepstrum (1/3)

Cepstrum:

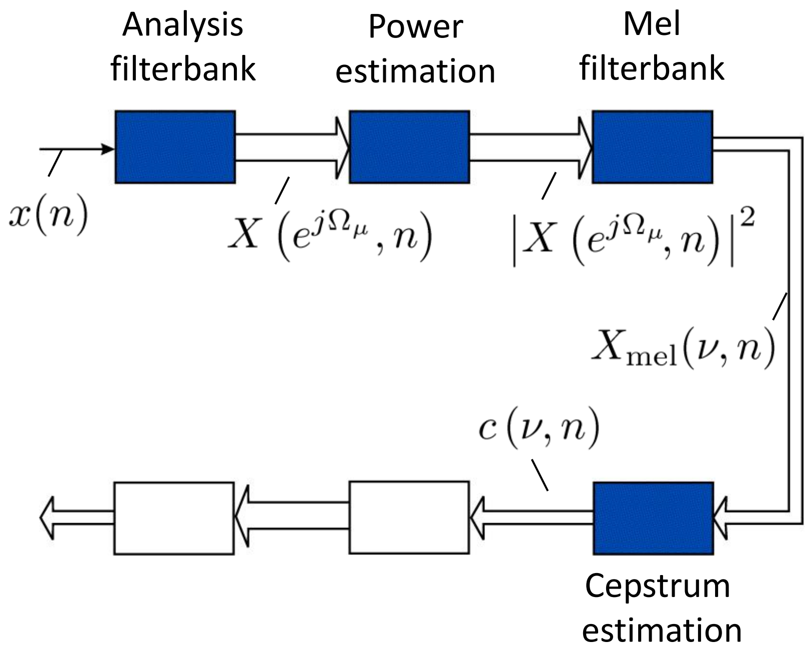
- The ***cepstrum*** is defined as the IDFT of the logarithm of the power spectrum:

$$c(\nu, n) = \frac{1}{M} \sum_{k=0}^{M-1} \log\left\{X_{\mathrm{mel}}(k,n)\right\} e^{j\frac{2\pi}{M} k\nu},$$

$$\boldsymbol{c}(n) = \left[c(0,n),\, ...,\, c(M-1,n)\right]^{\mathrm{T}}.$$

- Useful transformation for decorrelating and removing speaker dependent information from the input features.

- The resulting cepstrum is symmetric, it‘s therefore sufficient to use only the first half of the cepstral coefficients.



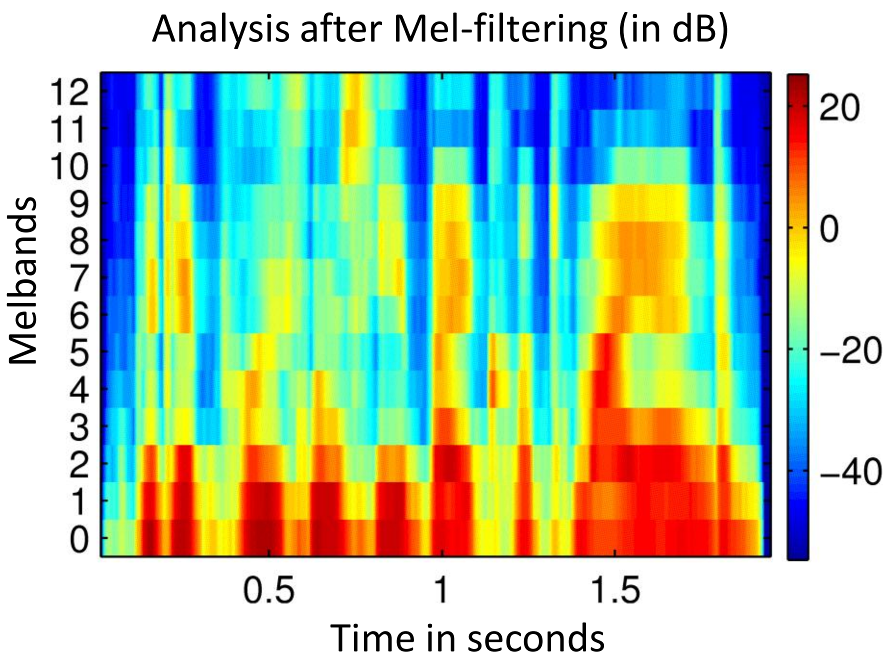
Feature Extraction – Cepstrum (2/3)

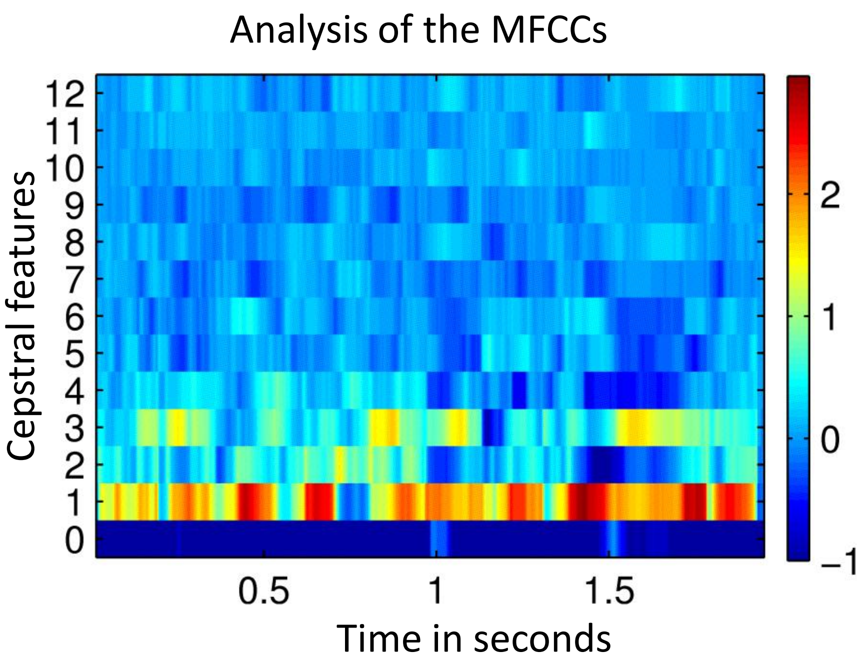
Properties:

- The purpose of the transformation is to ***decorrelate*** the logarithmic input features:
  - ➡ Good for classification.
- The outcome features are called ***Mel-Frequency Cepstral Coefficients*** (MFCCs).
- Since the input vector is real-valued the IDFT can be replaced by IDCT for efficient implementation.
- The cepstrum contains largely ***vocal tract*** information (concentrates at lower bands).
- Usually the ***dimension*** of the ***feature vector*** is ***reduced*** by minimizing the behavior of the ***pitch frequency*** (concentrates at higher bands). Often the last third of the feature elements is discarded.
- Cepstral coefficients are ***intensive to loudness*** (only energy changes, cepstrum unaffected).

Feature Extraction – Cepstrum (3/3)

Example:





- The computed ***MFCCs*** of this example have mainly high energy at lower quefrencies.

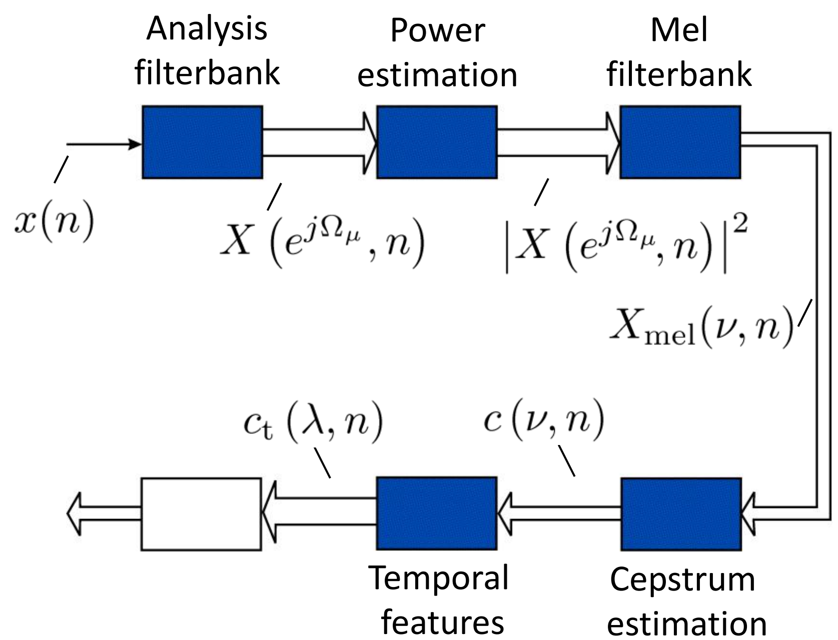
Feature Extraction – Temporal Features (1/2)

Temporal Features:

- After feature extraction often some number of successive ***feature vectors*** are ***combined.***

- In some cases the difference of adjacent feature vectors is computed (called ***delta features***) or the difference of two adjacent differences (called ***delta-delta features***).

- In the first case, successive vectors are ***stacked*** to a ***multi-feature vector*** as follows:

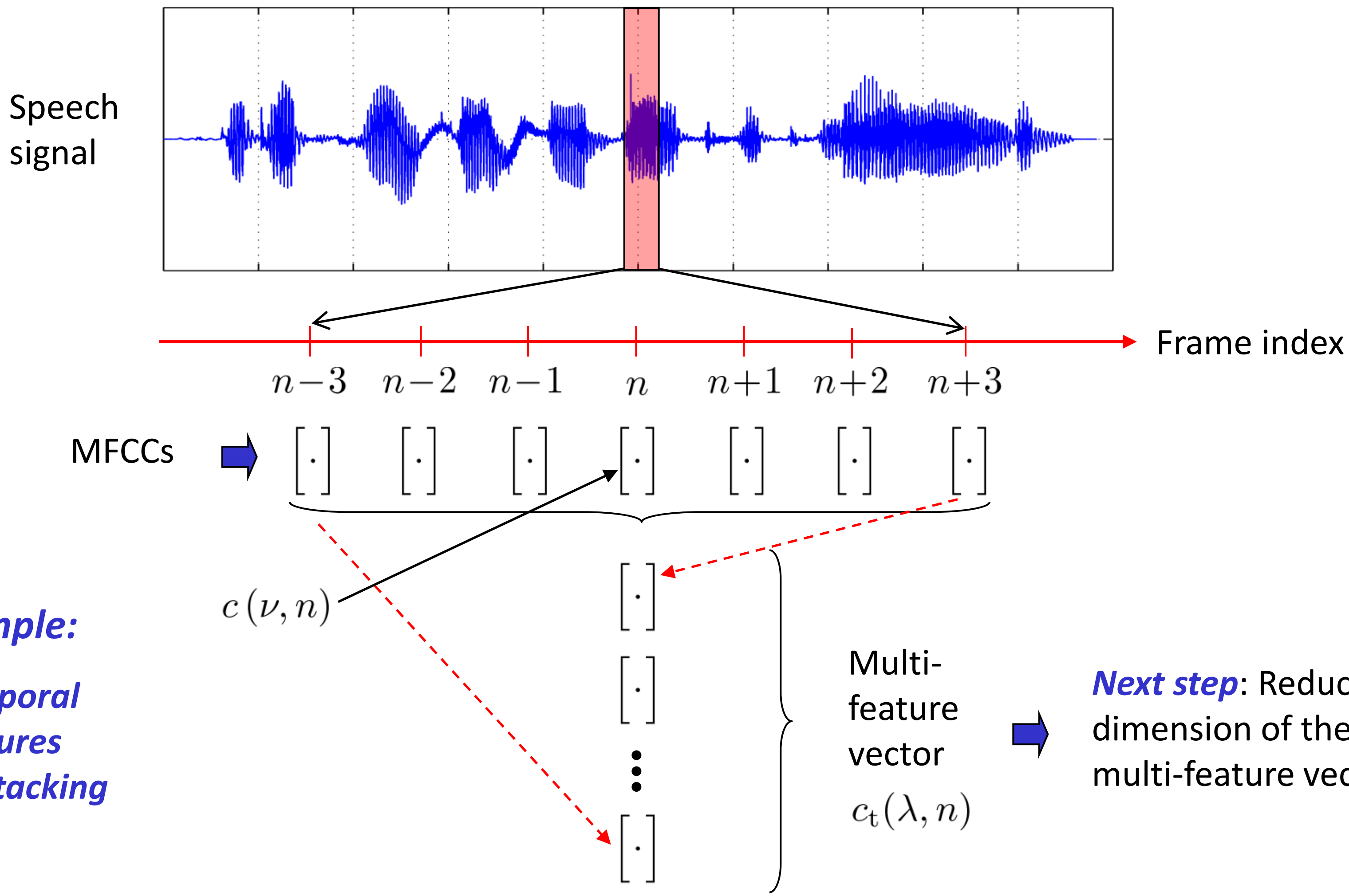


$$\boldsymbol{c}_{\mathrm{t}}(n) = \left[\boldsymbol{c}^{\mathrm{T}}(n - P_{\mathrm{b}}),\, \dots,\, \boldsymbol{c}^{\mathrm{T}}(n),\, \dots,\, \boldsymbol{c}^{\mathrm{T}}(n + P_{\mathrm{f}})\right]^{\mathrm{T}},$$

Number of feature vectors from the past and the future

$$\boldsymbol{c}_{\mathrm{t}}(n) = \left[c_{\mathrm{t}}(0, n),\, \dots,\, c_{\mathrm{t}}(M_{\mathrm{t}} - 1, n)\right]^{\mathrm{T}}, \quad M_{\mathrm{t}} = M + M\left(P_{\mathrm{f}} + P_{\mathrm{b}}\right).$$

Feature Extraction – Temporal Features (2/2)



Example:

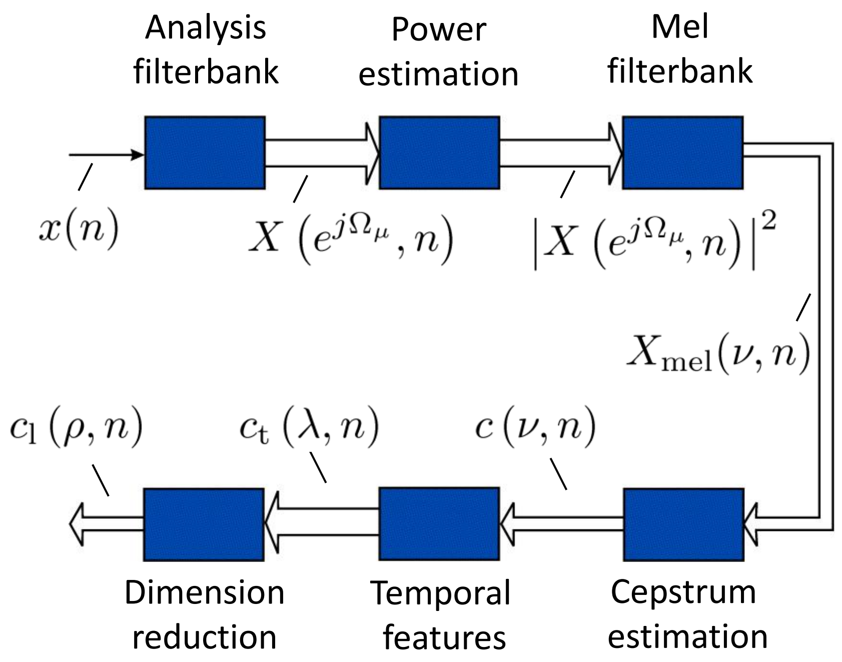
Temporal features by stacking

Next step: Reduce dimension of the multi-feature vector.

Feature Extraction – Feature Space Transformation (1/6)

Dimension Reduction:

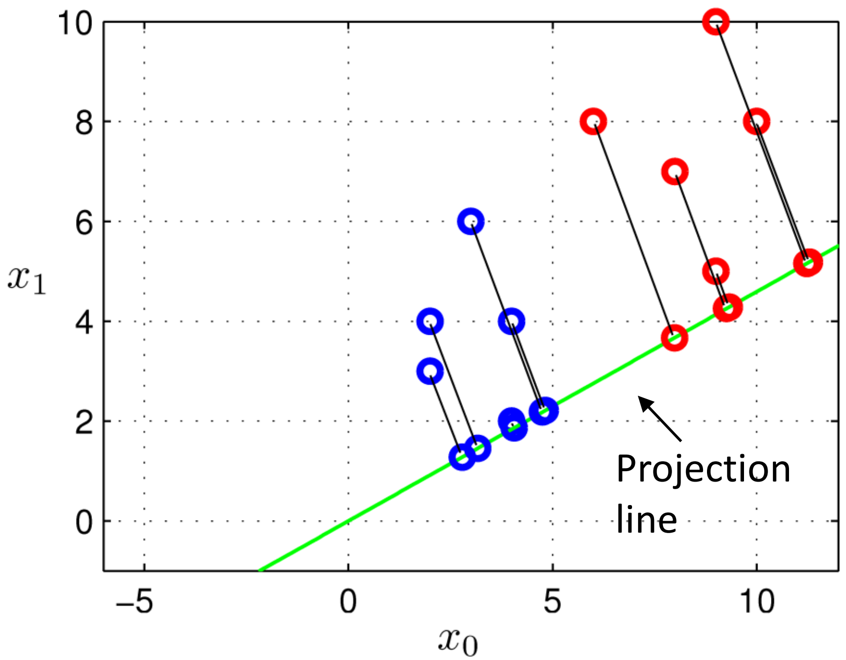
- A ***feature space transformation*** is performed to reduce the dimension of the input features $\boldsymbol{c}_{\mathrm{t}}(n)$.
- The so-called ***linear discriminant analysis*** (LDA) can be applied.
- The objective of LDA is to perform dimensionality reduction while ***preserving*** as much of the ***class discriminatory*** information as possible.
- The ***variance*** of features which corresponds to a feature class is ***minimized*** while the ***distance*** between feature ***classes*** is ***maximized.***



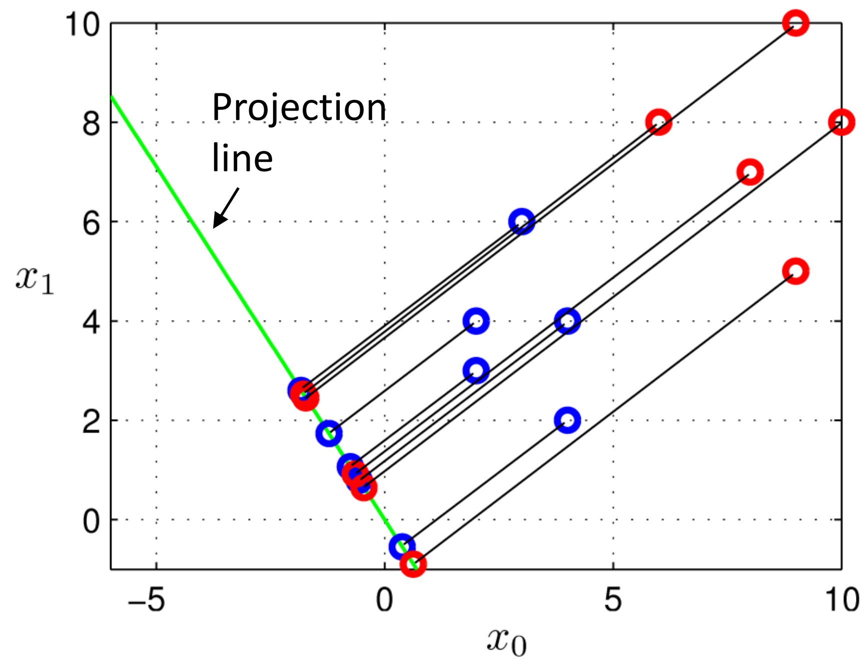
Feature Extraction – Feature Space Transformation (2/6)

LDA Example:

Good separability between two classes



Bad separability between two classes



- Select the line that ***maximizes*** the ***separability*** of the ***projected features.***

Feature Extraction – Feature Space Transformation (3/6)

LDA Derivation – Two Classes:

- Suppose we have a data set T consisting of M observations of a N- dimensional Euclidian variable $\boldsymbol{x}$:

$$T = \left\{\boldsymbol{x}^{(0)}, \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(M-1)}\right\}, \quad \text{with} \quad \boldsymbol{x}^{(m)} = \left[x_0^{(m)}, x_1^{(m)}, \ldots, x_{N-1}^{(m)}\right]^{\mathrm{T}}.$$

- We take the N-dimensional input vector $\boldsymbol{x}$ and ***project*** it down to one dimension using a projection vector $\boldsymbol{w}$:

$$y^{(m)} = \boldsymbol{w}^{\mathrm{T}} \boldsymbol{x}^{(m)}, \quad \text{with} \quad \boldsymbol{w} = [w_0, w_1, \ldots, w_{N-1}]^{\mathrm{T}}.$$

- In general the projection onto one dimension leads to a considerable loss of information:

➡ ***Select a projection that maximizes the class separation.***

Feature Extraction – Feature Space Transformation (4/6)

LDA Derivation (continued):

- The simplest measure of the separation of the classes is the separation of the ***projected*** class ***means***. For a ***two class problem***, chose $\boldsymbol{w}$ so as to maximize:

$$\tilde{\mu}_1 - \tilde{\mu}_0 \;=\; \boldsymbol{w}^{\mathrm{T}}\big(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0\big) \rightarrow \max\,,$$

Mean vectors of the two classes C_0 and C_1 with M_0 and M_1 points.

$$\text{with}\quad \boldsymbol{\mu}_0 \;=\; \frac{1}{M_0}\sum_{l\in C_0}\boldsymbol{x}^{(l)}\,,\quad \boldsymbol{\mu}_1 \;=\; \frac{1}{M_1}\sum_{l\in C_1}\boldsymbol{x}^{(l)}\,.$$

- The within ***class variances*** of the transformed data from classes C_0 and C_1 are given by:

$$s_0^2 \;=\; \sum_{l\in C_0}\Big(y^{(l)} - \tilde{\mu}_0\Big)^2\,,\qquad s_1^2 \;=\; \sum_{l\in C_1}\Big(y^{(l)} - \tilde{\mu}_1\Big)^2\,.$$

Transformed feature

- The ***total*** within ***class variance*** is simply: $s_0^2 + s_1^2\,.$

Feature Extraction – Feature Space Transformation (5/6)

LDA Derivation (continued):

- ***Fisher's idea***: Maximize a function that will give a large separation between the projected class ***means*** while also giving a small ***variance*** within each class, thereby minimizing the class overlap. The Fisher criterion is defined as:

$$J(\boldsymbol{w}) \;=\; \frac{(\tilde{\mu}_1 - \tilde{\mu}_0)^2}{s_0^2 + s_1^2} \;=\; \frac{\boldsymbol{w}^{\mathrm{T}}\,\boldsymbol{S}_{\mathrm{b}}\,\boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}}\,\boldsymbol{S}_{\mathrm{w}}\,\boldsymbol{w}}\,.$$

- The matrix $\boldsymbol{S}_{\mathrm{b}}$ is called ***between class covariance matrix***:

$$\boldsymbol{S}_{\mathrm{b}} \;=\; (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)\,(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)^{\mathrm{T}}.$$

- The matrix $\boldsymbol{S}_{\mathrm{w}}$ is called ***within class covariance matrix***:

$$\boldsymbol{S}_{\mathrm{w}} \;=\; \sum_{l \in C_0} \left(\boldsymbol{x}^{(l)} - \boldsymbol{\mu}_0\right)\left(\boldsymbol{x}^{(l)} - \boldsymbol{\mu}_0\right)^{\mathrm{T}} + \sum_{l \in C_1} \left(\boldsymbol{x}^{(l)} - \boldsymbol{\mu}_1\right)\left(\boldsymbol{x}^{(l)} - \boldsymbol{\mu}_1\right)^{\mathrm{T}}.$$

Feature Extraction – Feature Space Transformation (6/6)

LDA Derivation (continued):

- The Fishers ***criterion*** is ***maximized*** to find the optimal weights:

$$\boldsymbol{w}_{\mathrm{opt}} = \underset{\boldsymbol{w}}{\operatorname{argmax}} \left(\frac{\boldsymbol{w}^{\mathrm{T}}\, \boldsymbol{S}_{\mathrm{b}}\, \boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}}\, \boldsymbol{S}_{\mathrm{w}}\, \boldsymbol{w}} \right) .$$

- Differentiating $J(\boldsymbol{w})$ with respect to $\boldsymbol{w}$ and setting the equation to zero we obtain:

$$\frac{\partial J(\boldsymbol{w})}{\partial \boldsymbol{w}} = \frac{\partial}{\partial \boldsymbol{w}} \left(\frac{\boldsymbol{w}^{\mathrm{T}}\, \boldsymbol{S}_{\mathrm{b}}\, \boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}}\, \boldsymbol{S}_{\mathrm{w}}\, \boldsymbol{w}} \right) = \mathbf{0}\, .$$

- The result is known as ***Fishers linear discriminant***:

$$\boxed{\boldsymbol{w}_{\mathrm{opt}} = \underset{\boldsymbol{w}}{\operatorname{argmax}} \left(\frac{\boldsymbol{w}^{\mathrm{T}}\, \boldsymbol{S}_{\mathrm{b}}\, \boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}}\, \boldsymbol{S}_{\mathrm{w}}\, \boldsymbol{w}} \right) = \boldsymbol{S}_{\mathrm{w}}^{-1} \left(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0 \right) .}$$

Literature

Further details can be found in:

- C. Bishop: ***Pattern Recognition and Machine Learning***, Springer, Berlin, Germany, 2006.
- C. Bishop: ***Neural Networks for Pattern Recognition***, Oxford University Press, UK, 1996.
- E Schukat-Talamanzzini: ***Automatische Spracherkennung – Grundlagen, Statistische Modelle und effiziente Algorithmen***, Vieweg, 1995.
- L. Rabiner, B.-H. Juang: ***Fundamentals of Speech Recognition***, Prentice-Hall, 1993.