POLITECNICO DI MILANO

# Linear predictive coding

Antonio Canclini
Augusto Sarti

# Idea and motivations



- Basic idea: a sample of a discrete-time signal can be approximated (predicted) as a linear combination of its past samples

- Motivations: why linear predictive coding?
  - LPC provides a parsimonious source-filter model for the human voice and other signals
  - LPC is good for low-bit-rate coding of speech, as in “Codebook-Excited” LP (CELP)
  - LPC provides a spectral envelope in the form of an all-pole digital filter
  - LPC spectral envelopes are well suited for audio work (estimation of vocal formants)
  - The LPC voice model has a (loose) physical interpretation
  - LPC is analytically tractable: mathematically precise, simple, and easy to implement
  - Variations on LPC show up in other kinds of audio signal analysis

# Data model



- A signal sample $s(n)$ at time $n$ can be approximated by a linear combination of its own $p$ past samples:

$$s(n) \approx a_1 s(n-1) + a_2 s(n-2) + \ldots + a_p s(n-p)$$
$$= \sum_{k=1}^{p} a_k s(n-k)$$

- The coefficients $a_k$ are assumed to be constant over the duration of the analysis window

- If we assume that the signal can be modelled as an autoregressive (AR) stochastic process, then $s(n)$ can be expressed as

$$s(n) = \sum_{k=1}^{p} a_k s(n-k) + Gu(n)$$

$G$ is a gain parameter

$u(n)$ is a white noise (excitation signal)

- Example: voice production can be modelled as above with $u(n)$ being the source excitation at the glottis, and $s(n)$ being the output voice signal

# Source-filter interpretation

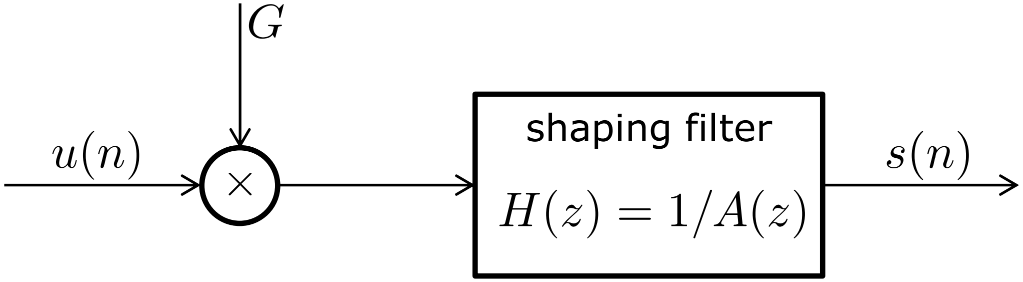
- Taking the z-transform of the previous equation we obtain

$$S(z) = \sum_{k=1}^{p} a_k z^{-k} S(z) + GU(z)$$

which lead to the transfer function

$$H(z) \triangleq \frac{S(z)}{GU(z)} = \frac{1}{1 - \sum_{k=1}^{p} a_k z^{-k}} \triangleq \frac{1}{A(z)} \quad \text{with} \quad A(z) \triangleq 1 - \sum_{k=1}^{p} a_k z^{-k}$$

- The source-filter interpretation of the above equation is provided in the figure below, showing the excitation source $u(n)$ being scaled by the gain $G$ , and fed to the all-pole system $H(z) = 1/A(z)$ to produce the voice signal $s(n)$



- Now, let’s look at LPC from the viewpoint of estimating a signal sample based on its past
- We consider the linear combination of past samples as the linearly predicted estimate $\hat{s}(n)$, defined by

$$\hat{s}(n) \triangleq \sum_{k=1}^{p} a_k s(n-k)$$

- We define the **prediction error** as

$$e(n) \triangleq s(n) - \hat{s}(n) = s(n) - \sum_{k=1}^{p} a_k s(n-k)$$

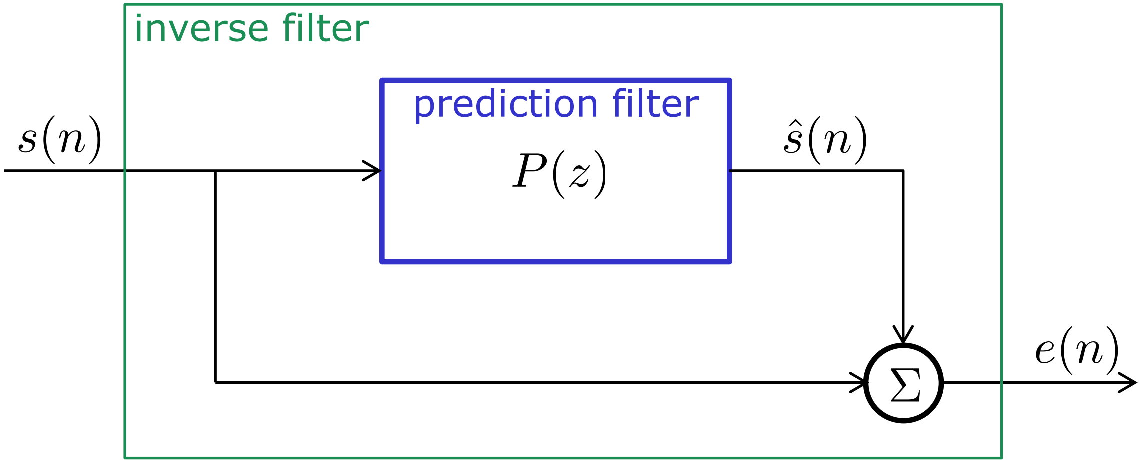
- In the z-domain, we obtain

$$E(z) = \left(1 - \sum_{k=1}^{p} a_k z^{-k}\right) S(z) = A(z)S(z) = \frac{S(z)}{H(z)}$$

- We can deduce that the prediction error $e(n)$ equals $Gu(n)$, i.e. the scaled white noise process

# Fundamentals of LPC

- Definitions:
  - $A(z)$ is called "**inverse filter**" or "**whitening filter**", as $E(z) = A(z)S(z)$
  - $H(z)$ is the "**forward filter**" or "**shaping filter**", as $S(z) = H(z)E(z)$
  - $P(z) \triangleq \sum_{k=1}^{p} a_k z^{-k}$ is the "**prediction filter**", as $\hat{S}(z) = P(z)S(z)$

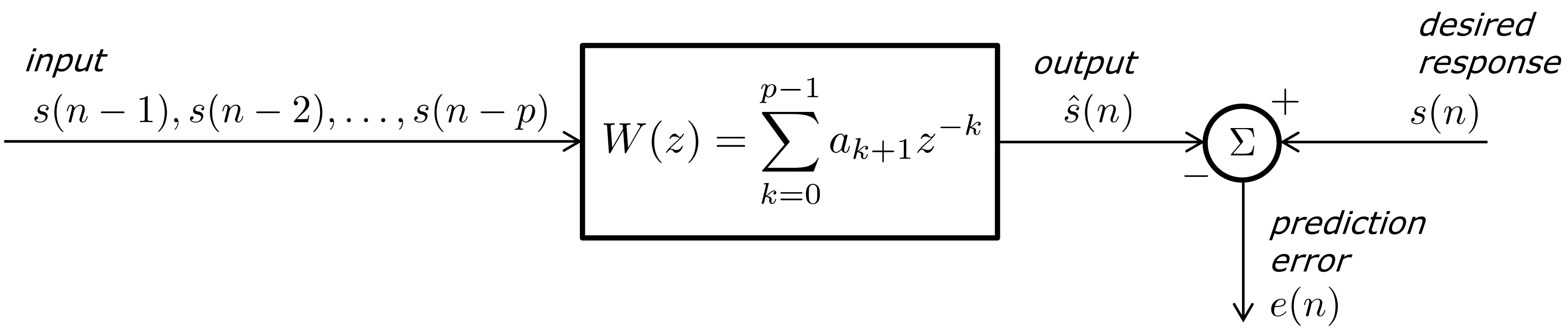


- Goal: find the set of predictor coefficients $\{a_k\}|_{k=1}^{p}$ that minimizes the mean-squared prediciton error, i.e.

$$\min_{a_k} \; E\left\{|s(n) - \hat{s}(n)|^2\right\}$$

- The problem can be set up as a Wiener Filtering problem, where the voice signal $s(n)$ constitutes both the filter input and the desired response

- The Wiener filter to be identified corresponds to the coefficients of the prediction filter $\{a_k\}|_{k=1}^{p}$ :



- Let $r(i) = E\{s(n)s(n-i)\}$ be the auto-correlation function of the input signal. The Wiener-Hopf equations for the LPC problem are thus given by

$$\boxed{\sum_{k=1}^{p} a_k r(i-k) = r(i)\ , \quad i = 1, 2, \ldots, p}$$

- Note that the cross-correlation of the input and the impulse response coincides with the auto-correlation function $r(i)$, as the desired response corresponds to the input signal

- The optimum LPC coefficients are found as the solution of the Wiener-Hopf equations. In matrix form:

$$\mathbf{a} = \mathbf{R}^{-1}\mathbf{r}\ ,$$

$$\text{where} \quad \mathbf{R} = \begin{bmatrix} r(0) & r(1) & \cdots & r(p-1) \\ r(1) & r(0) & \cdots & r(p-2) \\ \vdots & \vdots & \ddots & \vdots \\ r(p-1) & r(p-2) & \cdots & r(0) \end{bmatrix}\ , \ \mathbf{r} = \begin{bmatrix} r(1) \\ r(2) \\ \vdots \\ r(p) \end{bmatrix}\ , \ \mathbf{a} = \begin{bmatrix} a_1 \\ a_2 \\ \vdots \\ a_p \end{bmatrix}$$

# Minimum MSE and Prediction Gain

- Using the Wiener theory, it is easy to compute the minimum MSE (i.e., the minimum value of the cost function $J \triangleq E\left\{|s(n)-\hat{s}(n)|^2\right\}$ ). Denoting the variance of the input signal as $\sigma_s^2 = r(0)$ , we have:

$$
\begin{aligned}
D_p \triangleq J_{\min} &= \sigma_s^2 - \mathbf{r}^T\mathbf{R}^{-1}\mathbf{r} \\
&= r(0) - \mathbf{r}^T\mathbf{a} \\
&= r(0) - \sum_{k=1}^{p} a_k r(k)
\end{aligned}
$$

- Definition: **prediction gain**

$$
\boxed{G_p = \frac{\sigma_s^2}{D_p}}
$$

$G_p \to \infty$ when the input signal $s(n)$ is highly predictable

$G_p \to 1$ when $s(n)$ is unpredictable (e.g., white noise)

# Infinite memory linear prediction (statistical interpretation)



- Let's assume that we are predicting $s(n)$ using the entire set of past samples:

$$\hat{s}(n) = \sum_{k=1}^{\infty} a_k s(n-k)$$

- For the orthogonality principle, we have that

$$E\left\{e_o(n)s(n-i)\right\} = 0 \ , \quad i = 1, 2, \ldots, \infty$$

- Computing the auto-correlation of the optimum error $e_o(n)$ gives us

$$\begin{aligned}
r_{e_o}(i) &\triangleq E\left\{e_o(n)e_o(n-i)\right\} \\
&= E\left\{e_o(n)[s(n-i) - \hat{s}_o(n-i)]\right\} \\
&= E\left\{e_o(n)\left[s(n-i) - \sum_{k=1}^{\infty} a_k s(n-k-i)\right]\right\} \\
&= E\left\{e_o(n)s(n-i)\right\} - \sum_{k=1}^{\infty} a_k E\left\{e_o(n)s(n-k-i)\right\} \\
&= 0 \ , \quad \forall i > 0
\end{aligned}$$

# Infinite memory linear prediction (statistical interpretation)



- Since the auto-correlation function must be even, then the previous equations also holds for $i < 0$
- It turns out that $r_{e_o}(i)$ is non-zero only when $i = 0$ , where $r_{e_o}(0) = D_p$
- Therefore, we have that:

$$\boxed{r_{e_o}(i) = D_p \delta(i)}$$

The auto-correlation of the optimum noise is a Dirac delta: **the optimum prediction error is a white noise process**

- Since $e_o(n)$ is purely random, the infinite-memory LP "extracts" all the information of $s(n)$ into the inverse filter $A(z)$
- The residual of the prediction process $e_o(n)$ is thus left with no sample-spanning information about the signal
- Infinite memory LP can be characterized as a whitening process of the input signal $s(n)$

# Objective of prediction



- After obtaining the prediction error from prediction, we can recover the original signal back from $e_o(n)$ and $A(z)$. Indeed, we can get $s(n)$ by feeding $e_o(n)$ into the shaping filter $H(z) = 1/A(z)$

- Since all the correlation information of $s(n)$ is contained in the inverse filter $A(z)$, **we can use any white noise with the same variance in the reconstruction process**

- Let's use any arbitrary white noise $e'(n)$ with the sample variance of $e_o(n)$, and let $s'(n)$ be the output of the filter $H(z)$ with $e'(n)$ as the input
  - $e_o(n)$ and $e'(n)$ have the same power spectrum, i.e.

$$|E'(e^{j\omega})|^2 = |E_o(e^{j\omega})|^2 = D_p \quad (\text{constant})$$

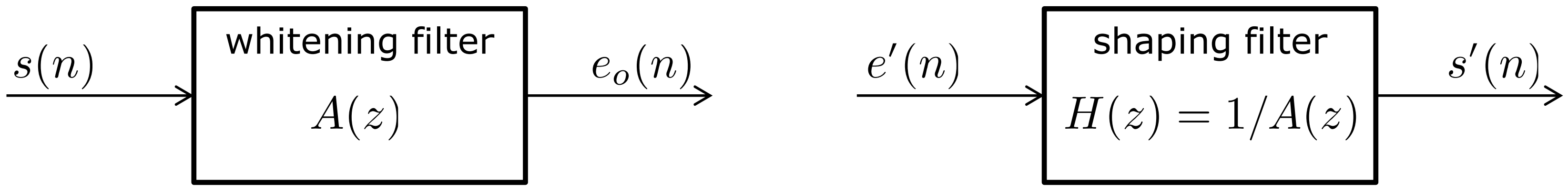
  - Although $s(n) \neq s'(n)$, they have the same power spectrum:

$$|S(e^{j\omega})|^2 = |H(e^{j\omega})|^2 |E_o(e^{j\omega})|^2 = |H(e^{j\omega})|^2 D_p$$
$$|S'(e^{j\omega})|^2 = |H(e^{j\omega})|^2 |E'(e^{j\omega})|^2 = |H(e^{j\omega})|^2 D_p$$

$$\Rightarrow \quad \boxed{|S(e^{j\omega})|^2 = |S'(e^{j\omega})|^2}$$

# Whitening and shaping filters

- Note that, in general, only the infinite-memory LP results in a true whitening filter $A(z)$. Anyway, by convention we call $A(z)$ the **whitening filter** even if the prediction order is finite
- In the finite case, the spectrum of the prediction error is flattened, but not white
- By analogy, $H(z) = 1/A(z)$ is called the **shaping filter**



# Implementation: dealing with non-stationary signals

- So far we assumed the signal as stationary, as required by the Wiener filtering theory

- To overcome this limitation, we perform the LPC analysis over short segments of the signal, over which the signal is assumed as stationary. For segments with length $M$ samples, the short-time voice and error signals are defined as:

$$s_n(m) \triangleq s(n+m)\ , \qquad m = 0, 1, 2, \ldots, M-1$$

$$e_n(m) \triangleq e(n+m)\ , \qquad m = 0, 1, 2, \ldots, M+p-1$$

- For each segment, we seek the LPC parameters (time-varying) that minimize the short-time mean-squared error:

$$\min_{\mathbf{a}_n} \varepsilon_n\ , \quad \text{where}$$

$$\varepsilon_n \triangleq \sum_{m=0}^{M+p-1} e_n^2(m) = \sum_{m=0}^{M+p-1} \left[ s_n(m) - \sum_{k=1}^{p} a_k(n) s_n(m-k) \right]^2$$

# Implementation: computation of the LPC parameters

- To solve the Wiener-Hopf equations, we need to estimate the samples of the auto-correlation function; in particular, we are interested in the short-time auto-correlation:

$$r_n(|i-k|) \triangleq \sum_{m=0}^{M-1-(i-k)} s_n(m)s_n(m+i-k) \qquad \begin{array}{l} 1 \le i \le p\,, \\ 0 \le k \le p\,, \\ i \ge k \end{array}$$

- The Wiener-Hopf equations therefore are specialized as

$$\sum_{k=1}^{p} a_k(n) r_n(|i-k|) = r_n(i)\ , \qquad i = 1, 2, \ldots, p$$

or, in matrix form as

$$\mathbf{a}_n = \mathbf{R}_n^{-1}\mathbf{r}_n\ ,$$

$$\text{where} \quad \mathbf{R} = \begin{bmatrix} r_n(0) & r_n(1) & \cdots & r_n(p-1) \\ r_n(1) & r_n(0) & \cdots & r_n(p-2) \\ \vdots & \vdots & \ddots & \vdots \\ r_n(p-1) & r_n(p-2) & \cdots & r_n(0) \end{bmatrix}, \ \mathbf{r}_n = \begin{bmatrix} r_n(1) \\ r_n(2) \\ \vdots \\ r_n(p) \end{bmatrix}, \ \mathbf{a}_n = \begin{bmatrix} a_1(n) \\ a_2(n) \\ \vdots \\ a_p(n) \end{bmatrix}$$

# Implementation: computation of the LPC parameters



- The estimate of the auto-correlation function provided by the function $r_n(|i-k|)$ leads to a minimum-phase shaping filter $A_n(z)$, i.e. its zeros are inside the unit circle

- This guarantees the stability of the shaping filter $H_n(z) = 1/A_n(z)$

- Remarks:

  - To efficiently compute the LPC coefficients, use the **Levinson-Durbin recursion** (fast algorithm exploiting the Toeplitz structure of the auto-correlation matrix $\mathbf{R}_n$ )

  - Apply a tapered window to the extracted frames (possibly with overlap) to attenuate edge effects

# Power spectrum envelope matching



- We saw earlier that in the case of infinite memory LP the power spectrum of the signal can be exactly reconstructed from the shaping filter and the error variance

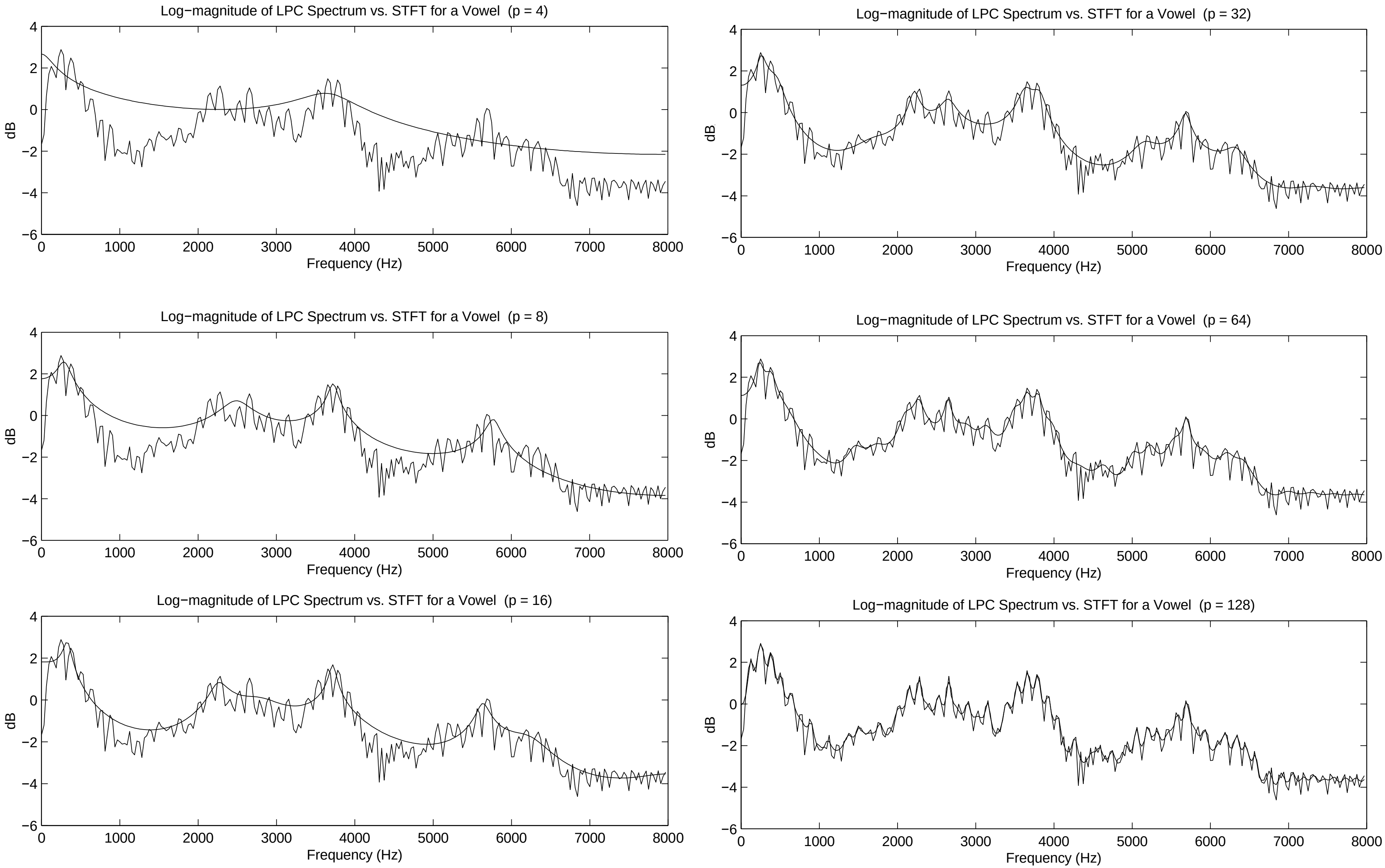
- This implies that, as $p \rightarrow \infty$ , we can approximate the power spectrum of the signal with arbitrarily small error using the all-pole shaping filter $H_n(z)$

$$\lim_{p \rightarrow \infty} |\hat{S}_n(e^{j\omega})|^2 = |S_n(e^{j\omega})|^2$$

- As the prediction order $p$ increases, the resulting mean-squared error $\varepsilon_n$ monotonically decreases. This implies that as we increase the prediction order, the LPC power spectrum $|\hat{S}_n(e^{j\omega})|^2$ will try to match the signal power spectrum $|S_n(e^{j\omega})|^2$ more closely

# Power spectrum envelope matching (example)





- The short-time error is time limited to the interval $[0, M+p-1]$, thus it can be expressed as

$$\varepsilon_n = \sum_{m=0}^{M+p-1} e_n^2(m) = \sum_{m=-\infty}^{\infty} e_n^2(m)$$

- We can express the above in the frequency domain using Parseval's Theorem:

$$\begin{aligned}\varepsilon_n &= \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|E_n(e^{j\omega})\right|^2 d\omega \\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi} \left|S_n(e^{j\omega})\right|^2 \left|A_n(e^{j\omega})\right|^2 d\omega \\ &= \frac{1}{2\pi}\int_{-\pi}^{\pi} \frac{\left|S_n(e^{j\omega})\right|^2}{\left|H_n(e^{j\omega})\right|^2} d\omega\end{aligned}$$

- Since the integrand is positive, we conclude the following:

$$\min_{\mathbf{a}_n} \ \varepsilon_n \quad \Longleftrightarrow \quad \min_{\mathbf{a}_n} \ \frac{\left|S_n(e^{j\omega})\right|^2}{\left|H_n(e^{j\omega})\right|^2}, \quad \forall \omega$$

# Power spectrum envelope matching



- As LPC attempts to minimize the ratio $|S_n(e^{j\omega})|^2/|H_n(e^{j\omega})|^2$ for $-\pi \leq \omega \leq \pi$ in the integral, there exists an interesting discrepancy in carrying out the minimization. We can identify the following regions:
  - **Region 1:** $|S_n(e^{j\omega})| > |H_n(e^{j\omega})|$ , corresponding to the region where the magnitude of the signal spectrum is large; here the integrand contributing to the error integral is relatively large (greater than 1)
  - **Region 2:** $|S_n(e^{j\omega})| < |H_n(e^{j\omega})|$ , where the magnitude of the signal spectrum is small, and the integrand contributing to the error integral is relatively small (less than 1)
- Integrands in region 1 contribute more to the total error than those in region 2
- From the above argument, it is clear that the LPC spectrum matches the signal spectrum much more closely in region 1 (near the spectrum peaks) than in region 2 (near spectral valleys)

# Power spectrum envelope matching

- Summarizing, the LPC spectrum can be considered to be a good spectral envelope estimator since it puts more emphasis on tracking peaks than tracking valleys

- As we increase the order $p$, the approximation to the valleys is going to improve as well as for the peaks since the total error becomes smaller

- Thus, the prediction order $p$ can serve as a control parameter for determining the smoothness of the LPC spectrum
  - if our goal is to capture the spectral envelope and not the fine structure, then it is essential to choose an appropriate value of $p$
  - **rule of thumb for speech**, at sample frequency $f_s$:

$$\boxed{\frac{f_s}{1000} \leq p \leq \frac{f_s}{1000} + 4}$$

  - E.g., if $f_s = 16\,\mathrm{kHz}$ then using $16 \leq p \leq 20$ would be approriate

# Frequency selective LPC



- We saw earlier that the prediction order can be adjusted to control the accuracy and the smoothness of the LPC spectrum

- In some cases, it would be nice to perform separate LPC analysis for a selected partition of the spectrum

- <u>Example</u>: for voiced speech, such as vowels, we are generally interested in the region from 0 to 4 kHz; for unvoiced sounds, such as fricatives, the region from 4 to 8 kHz is important

- <u>Motivation</u>: using frequency selective linear prediction, the spectrum from 0 to 4 kHz can be modeled by a predictor of order $p_1$ ; while the region from 4 to 8 kHz can be modeled by a different predictor of order $p_2$ . In most of the cases, we want a smoother fit (smaller order $p$) in the higher octaves

# Frequency selective LPC



- To model only the frequency range $f \in [f_A, f_B]$ we perform the following:

1. Map the interval to a normalized frequency

$$\omega \in [2\pi f_A, 2\pi f_B] \qquad \Longrightarrow \qquad \omega' \in [0, 2\pi]$$

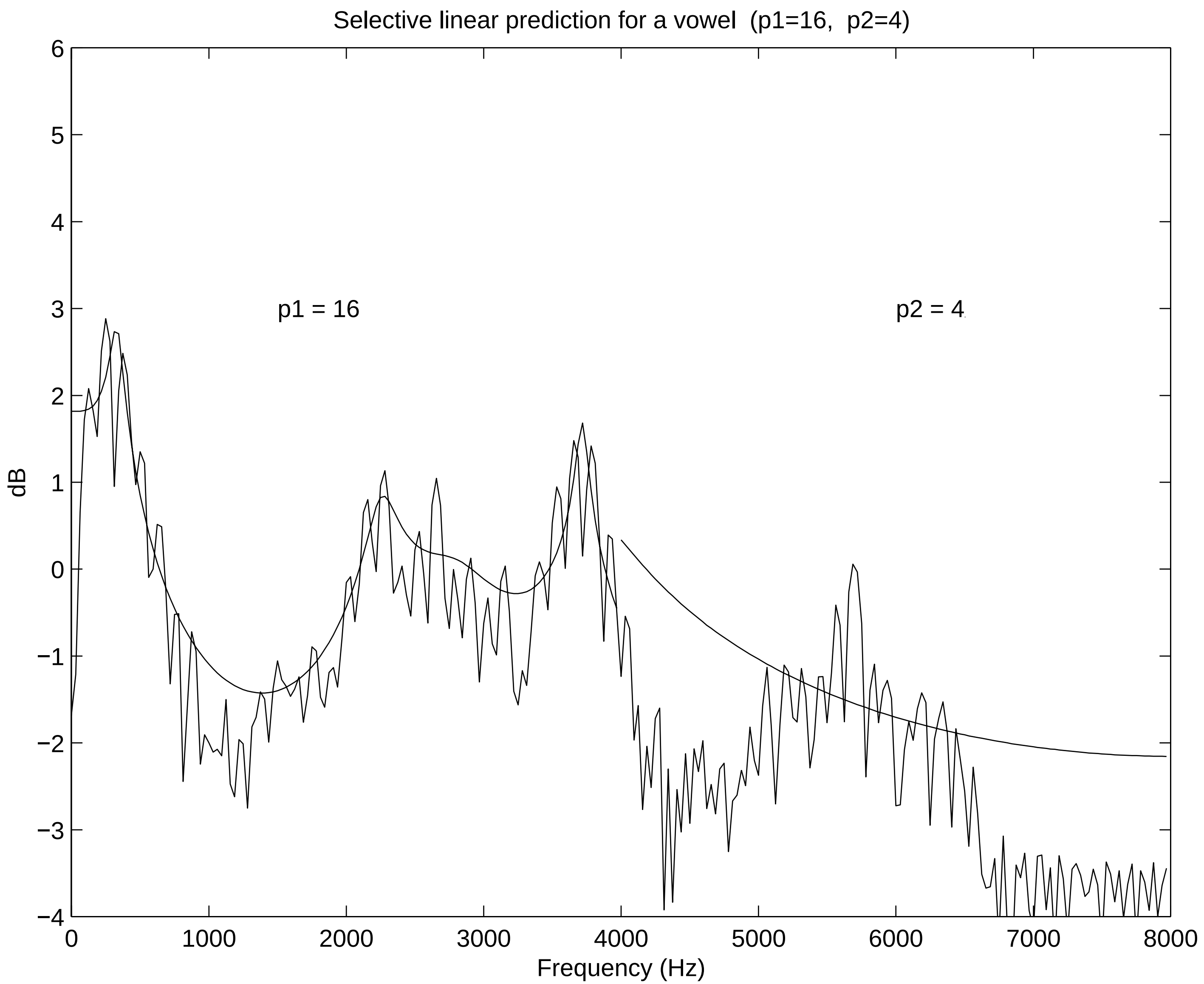
2. Obtain the new auto-correlation coefficients by Inverse Discrete-Time Fourier Transform (IDTFT):

$$r'_n(k) = \frac{1}{2\pi} \int_{-\pi}^{\pi} |S_n(e^{j\omega'})|^2 e^{j\omega' k} d\omega'$$

3. Solve the new set of Wiener-Hopf equations using the samples of the auto-correlation $\{r'_n(k)\}$ to get the predictor coefficients for that particular spectral region

# Frequency selective LPC - example

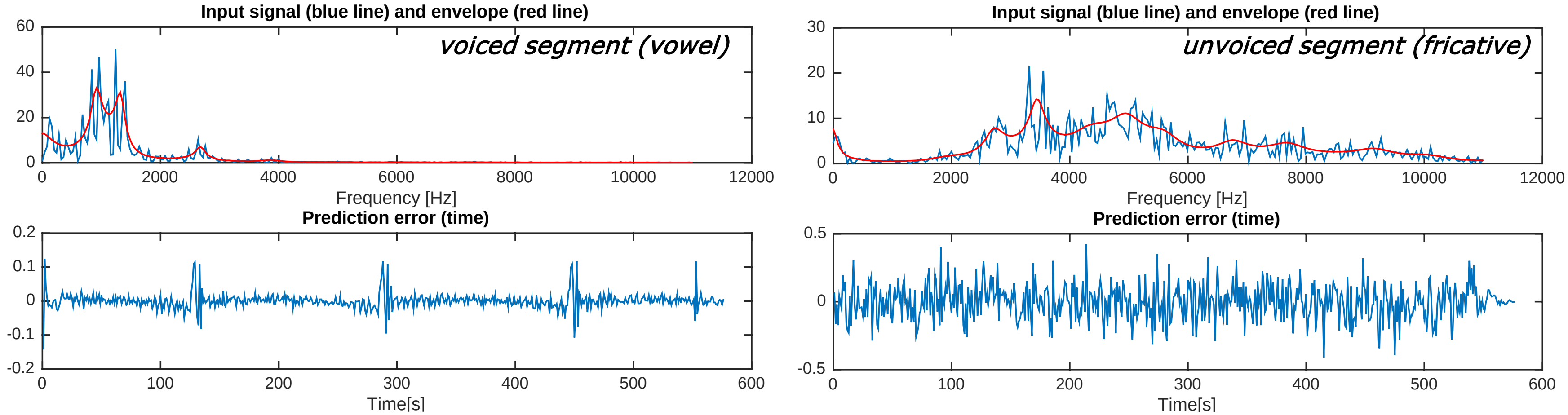




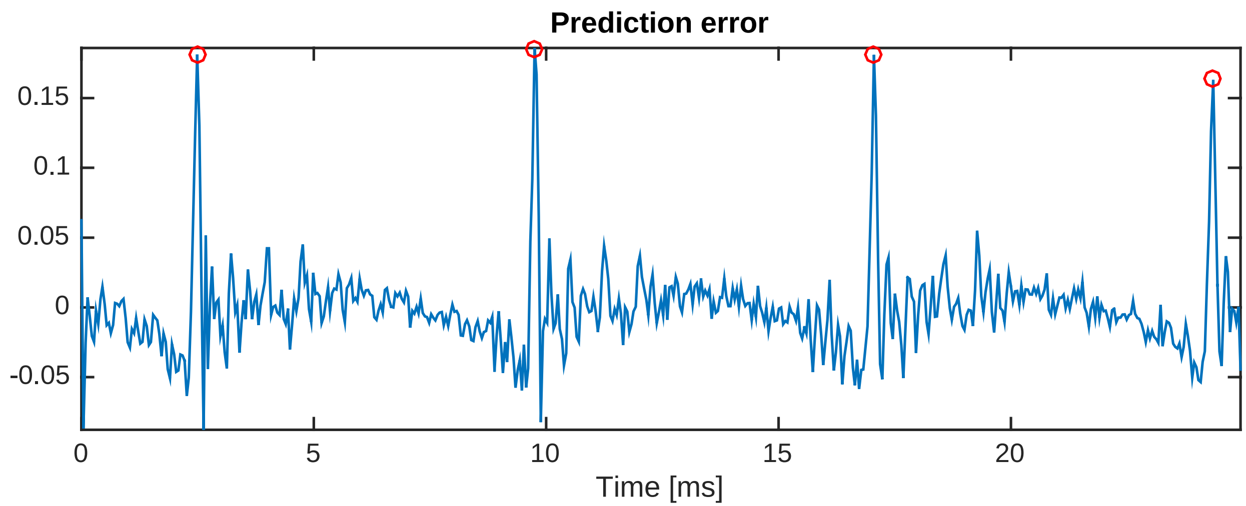
# Applications of LPC

## ❑ Speech coding/synthesis

- model of speech production: $e_n(m)$ is the excitation source at the glottis; and $H_n(z)$ represents the transfer function of the vocal tract
- we can encode the LPC parameters to achieve data compression
- we distinguish between voiced/unvoiced segments, leading to different prediction errors (see figure below)
- idea:
  - use a train of pulses as excitation signal $e'_n(m)$ for synthesizing voiced segments
  - use a white noise as $e'_n(m)$ for synthesizing unvoiced segments



- **Robust pitch prediction:**
  - extract peaks from the prediction error of voiced segments
  - compute the average distance between peaks to obtain an estimate of the pitch




  - a more robust estimation can be obtained considering a long-term predictor, i.e. predicting the current sample from the past values one pitch period earlier

## Cross Synthesis in computer music: talking instruments

- We may feed any sound (typically that of a musical instrument) into the shaping filter $H_n(z)$ obtained from LPC analysis of a speech segment
- The musical signal input acts as a periodic excitation source $e_n(m)$ to the shaping filter $H_n(z)$, and thus the output spectrum will possess vocal formant structure as well as the harmonic and textural qualities of the musical sound

- Methodology:
  - for each frame, perform LPC analysis on the musical signal $x^{\mathrm{M}}(n)$ and the speech signal $x^{\mathrm{S}}(n)$
  - use the prediction error $e^{\mathrm{M}}(n)$ of the musical segment to feed the shaping filter $H^{\mathrm{S}}(z)$ of the speech segment: