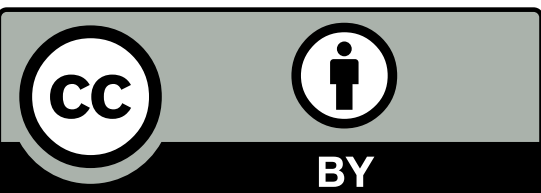


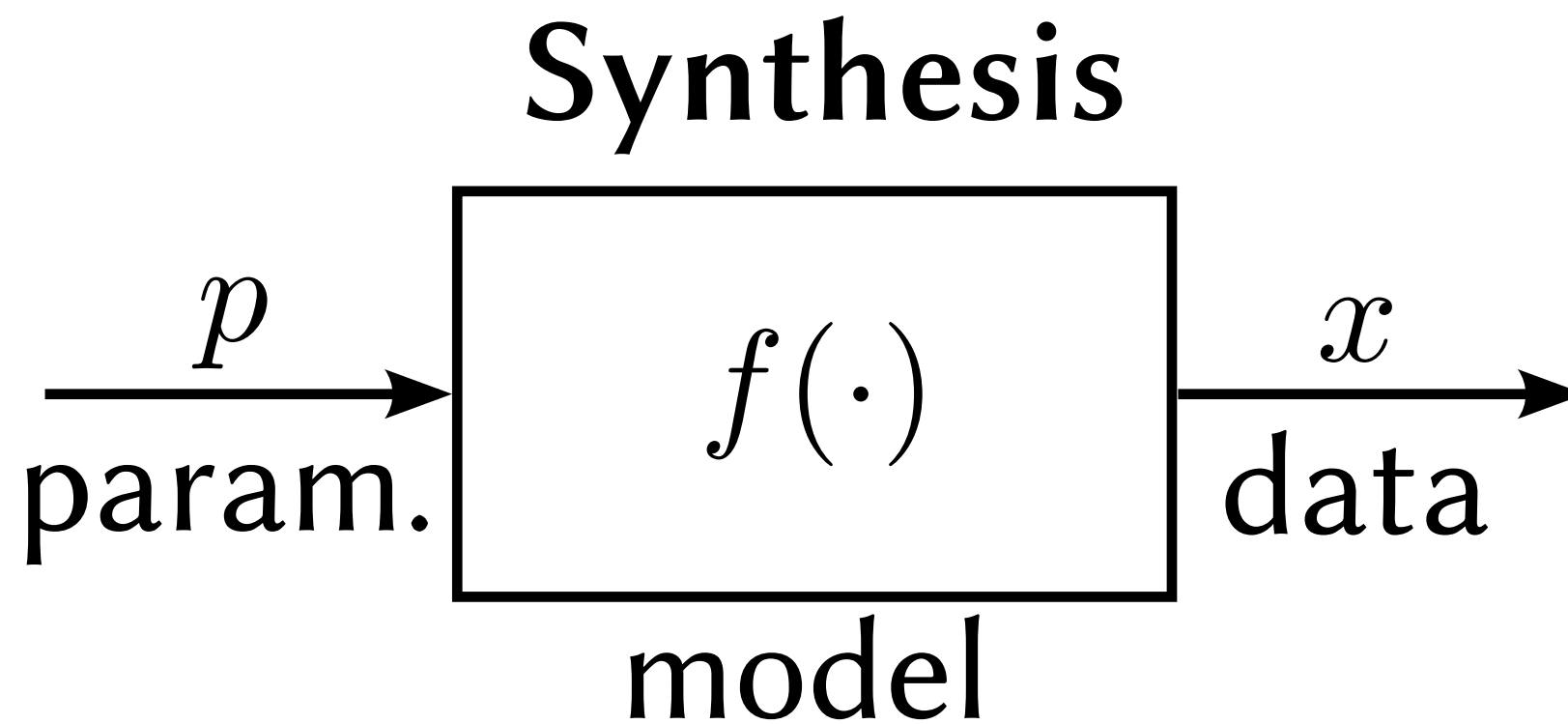
Linear Prediction

Digital Audio Coding

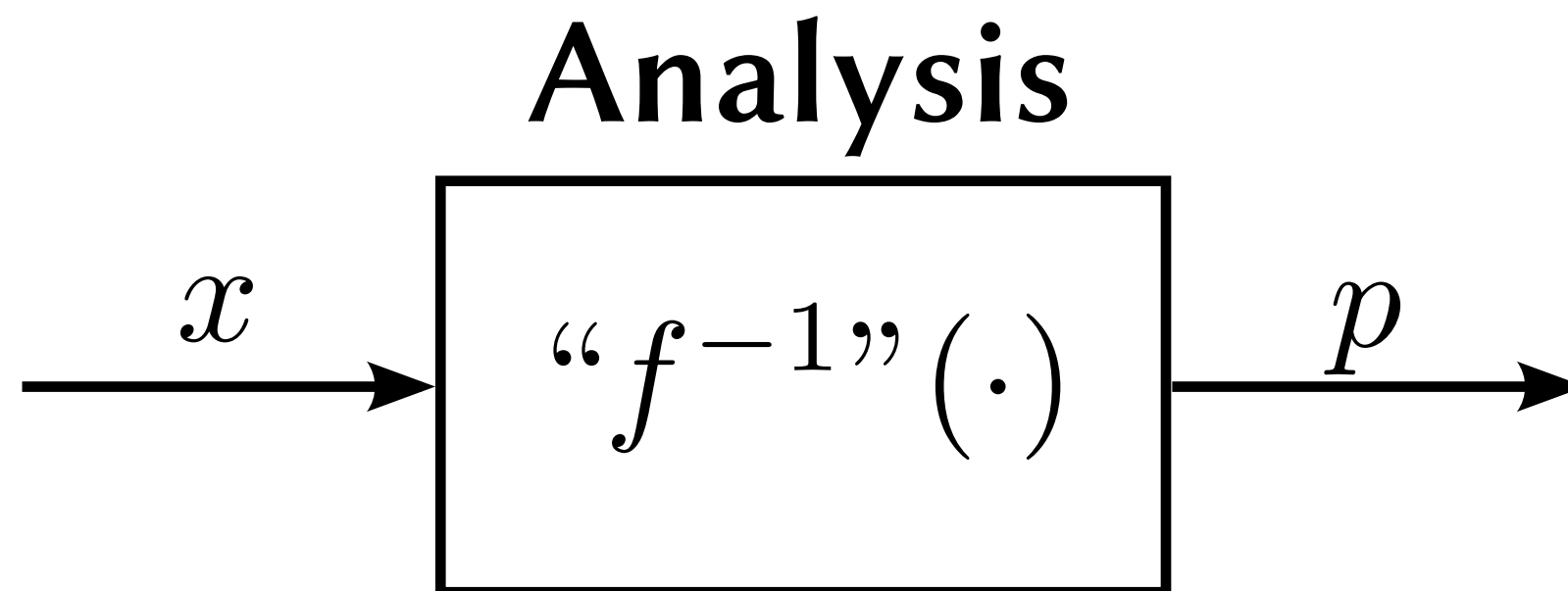


SEBASTIEN.BOISGERAULT@MINES-PARISTECH.FR

Parametric Models



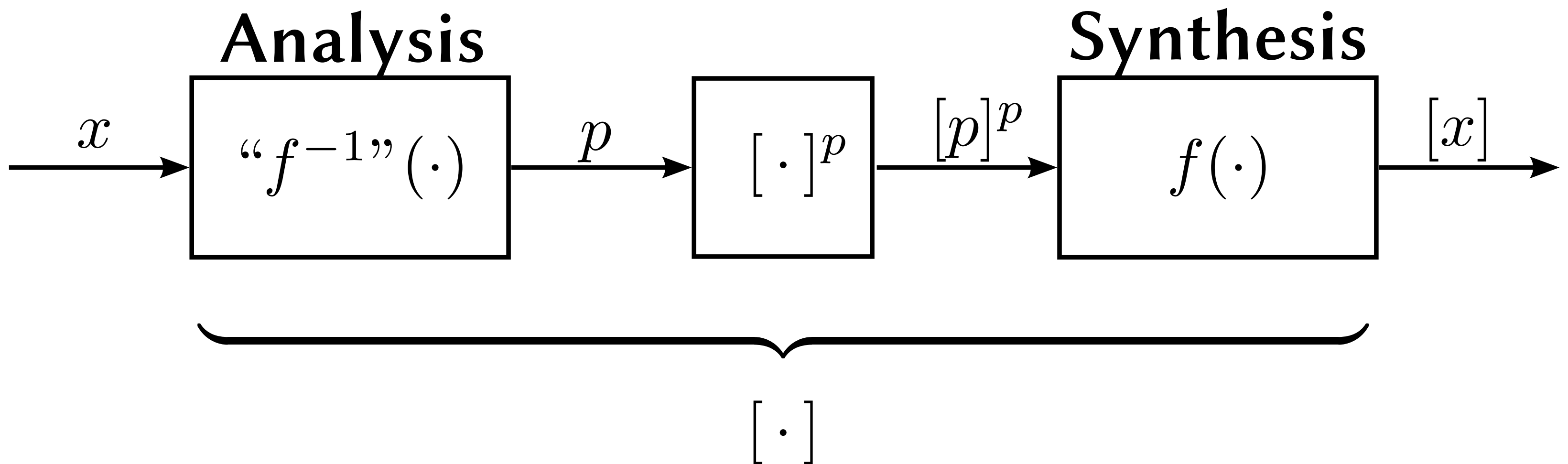
Assume that f is onto ; invert with



$$p = \arg \min_{p'} \{j(p') \mid f(p') = x\}$$

Parametric Models

Quantization



Linear Prediction Principles

Given a sequence of values x_0, x_1, \dots, x_{n-1} ,
find the best approximation \hat{x}_i of x_i such that:

$$\hat{x}_i = a_1 x_{i-1} + a_2 x_{i-2} + \dots + a_m x_{i-m}$$

The prediction is **linear** and (strictly) **causal**.

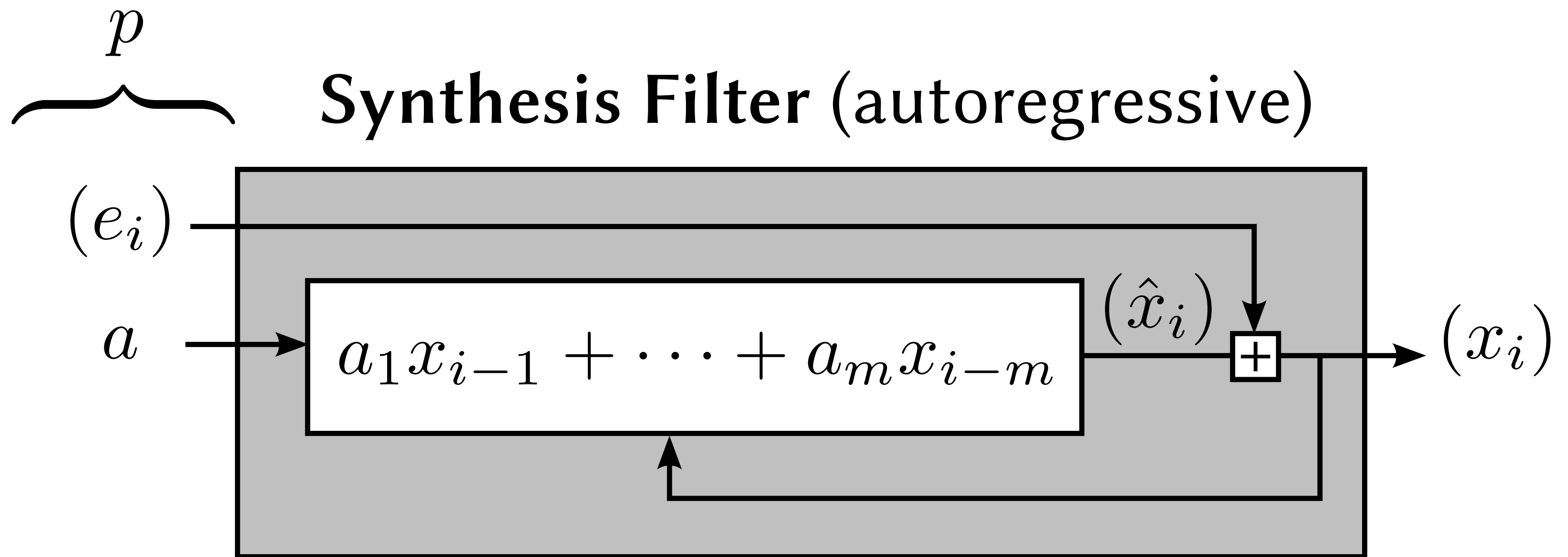
The number m is the **prediction order**.

The **prediction error/residual** is defined by:

$$e_i = x_i - \hat{x}_i$$

$$(x_i = \hat{x}_i + e_i)$$

Linear Prediction



Linear Prediction

Covariance Method

Solve

$$a^{\star} = \arg \min_{a \in \mathbb{R}^m} j(a)$$

with

$$j(a) = \sum_{i=m}^{n-1} (x_i - a_1 x_{i-1} - \cdots - a_m x_{i-m})^2$$

Linear Prediction

$$a^{\star} = \arg \min_{a \in \mathbb{R}^m} \|Aa - b\|^2$$

Covariance Method:

$$A = \begin{bmatrix} x_{m-1} & x_{m-2} & \dots & x_0 \\ x_m & x_{m-1} & \dots & x_1 \\ \vdots & \vdots & \vdots & \vdots \\ x_{n-2} & x_{n-3} & \dots & x_{n-m-1} \end{bmatrix}, \quad b = \begin{bmatrix} x_m \\ x_{m+1} \\ \vdots \\ x_{n-1} \end{bmatrix}$$

Linear Prediction

$$a^{\star} = \arg \min_{a \in \mathbb{R}^m} \|Aa - b\|^2$$

Unique solution if A is into:

$$a^{\star} = [A^t A]^{-1} A^t b$$

Otherwise, one solution of:

$$j(a^{\star}) = \min_{a \in \mathbb{R}^n} j(a)$$

provided by:

$$a^{\star} = A^{\#} b \quad \text{where} \quad A^{\#} = \lim_{\epsilon \rightarrow 0} [A^t A + \epsilon I]^{-1} A^t$$

Linear Prediction

With `numpy.linalg` least-square solution to $Ax = b$:

```
def lp(x, m):
```

```
    "Linear predictor coefficients -- covariance method"
```

```
    n = len(x)
```

```
    A = array([x[m - 1 - arange(0, m) + i] for i in range(n-m)])
```

```
    b = x[m:n]
```

```
    a = lstsq(A, b)[0]
```

```
    return a
```

AutoCorrelation

Variant: treat the data as an infinite signal.

Set $x_i = 0$ if $i \neq 0, \dots, n-1$ and minimize:

$$j(a) = \sum_{i=-\infty}^{+\infty} (x_i - a_1 x_{i-1} - \dots - a_m x_{i-m})^2$$

The solutions are the same if we minimize:

$$j(a) = \sum_{i=0}^{n+m-1} (x_i - a_1 x_{i-1} - \dots - a_m x_{i-m})^2$$

AutoCorrelation

```
def lp(x, m, zero_padding=False):
```

```
    "Linear predictor coefficients"
```

```
    if zero_padding:
```

```
        x = r_[zeros(m), x, zeros(m)]
```

```
    n = len(x)
```

```
    A = array([x[m - 1 - arange(0, m) + i] for i in range(n-m)])
```

```
    b = x[m:n]
```

```
    a = lstsq(A, b)[0]
```

```
    return a
```

Method Selection

Covariance:

- Fast computation,
- Accurate solution.

Autocorrelation:

- Even faster computation,
- Stable synthesis filter.

Infinite Order

Define $e_i = x_i - \sum_{j=1}^{+\infty} a_j x_{i-j}$ for $a \in L^2(\mathbb{N}^*)$

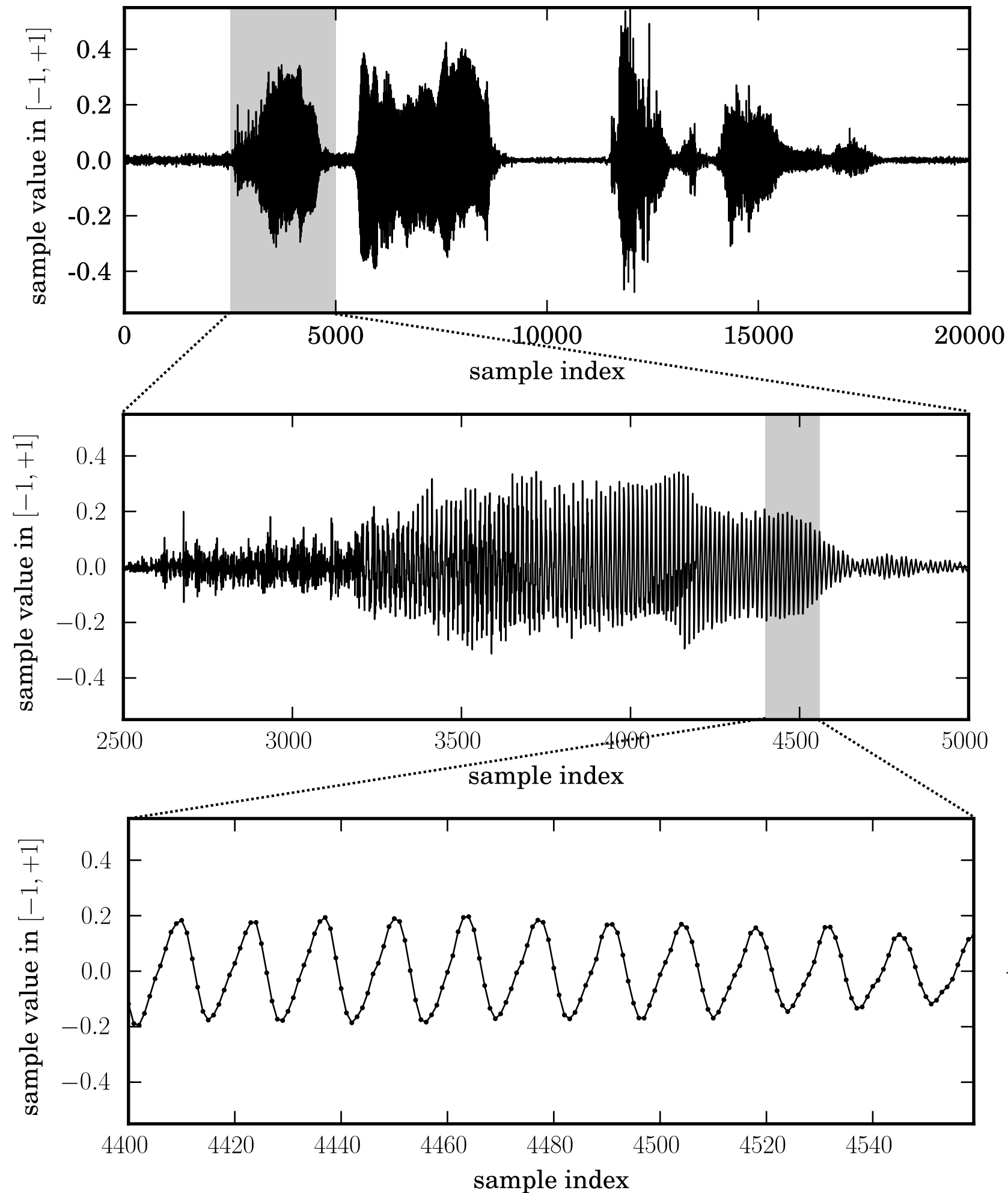
If a is a minimum of $j(a) = \sum_{i=-\infty}^{+\infty} e_i^2$ then

$$\forall j \in \mathbb{Z}^*, \sum_{i \in \mathbb{Z}} e_i e_{i+j} = 0 \iff |e(f)| = \text{const.}$$

The prediction error is a **white noise**.

Voice Audio Data

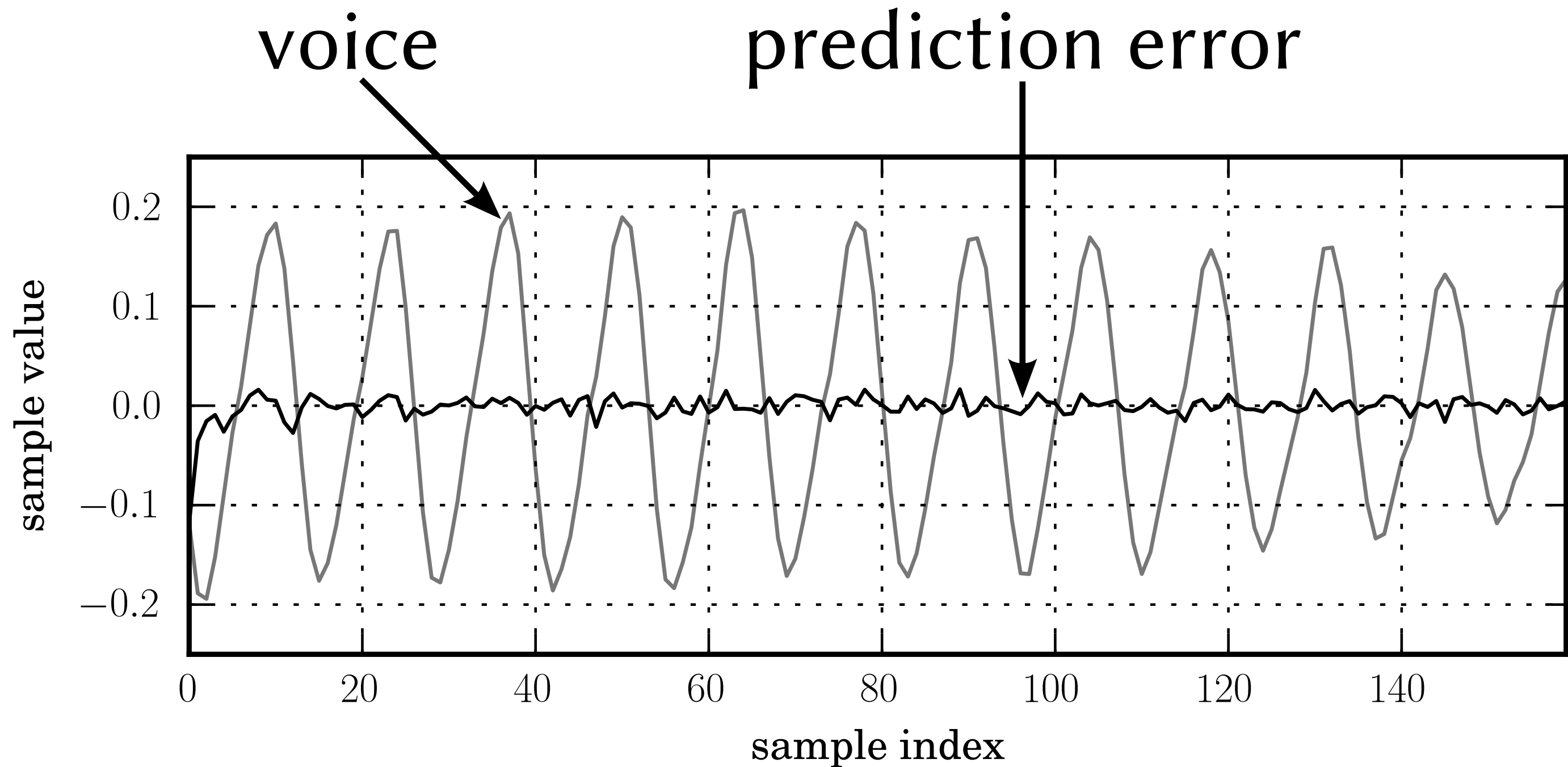
You Wanna Have Babies ?
8 kHz, mono.



← 20 ms frame

Voice Audio Data

Short-Term Prediction



order 16, autocorrelation

Analysis Filter

Finite Impulse Response filter – FIR

$$y_n = a_0 u_n + a_1 u_{n-1} + \cdots + a_{N-1} u_{n-N+1}$$



```
class FIR(Filter):
```

```
    def __call__(self, input):
```

```
        output = self.a[0] * input + dot(self.a[1:], self.state)
```

```
        self.state = r_[input, self.state[:-1]]
```

```
        return output
```


FIR Example

Moving Average

$$y_n = 0.25 (u_n + u_{n-1} + u_{n-2} + u_{n-3})$$

```
>>> ma = FIR(0.25 * ones(4))
```

```
>>> ma(1.0)
```

```
0.25
```

```
>>> ma(2.0)
```

```
0.75
```

```
>>> ma([3.0, 4.0])
```

```
array([1.5, 2.5])
```

FIR for Linear Prediction

```
>>> a = lp(data, order=m, ...)
```

Predictor

```
>>> predictor = FIR(r_[0, a])
```

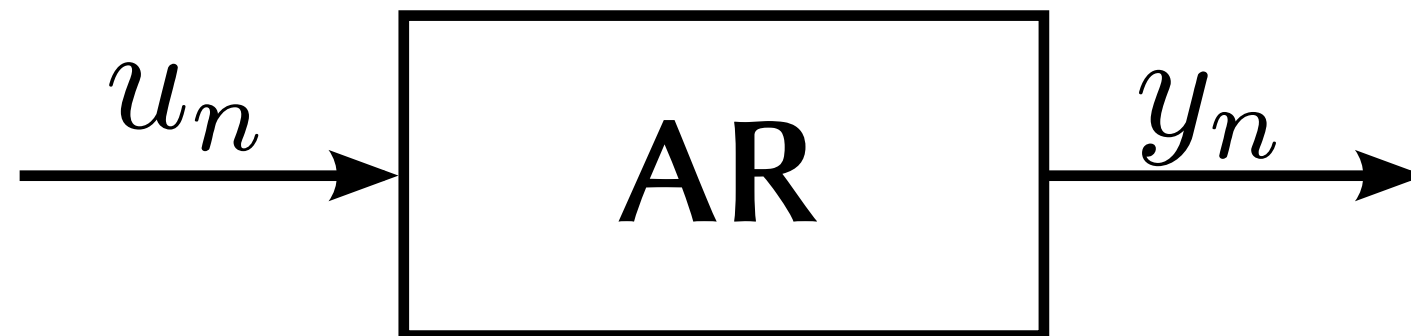
Analysis Filter

```
>>> error = FIR(r_[1.0, -a])
```

Synthesis Filter

Auto-Regressive filter - AR

$$y_n = a_1 y_{n-1} + \cdots + a_N y_{n-N} + u_n$$



```
class AR(Filter):
```

```
    def __call__(self, input):
```

```
        output = self.a[0] * input + dot(self.a, self.state)
```

```
        self.state = r_[output, self.state[:-1]]
```

```
        return output
```

AR Example

$$y_n = 0.5 \times y_{n-1} + u_n$$

```
>>> ar = AR([0.5])
```

```
>>> ar.state = [1.0]
```

```
>>> ar(0.0)
```

0.5

```
>>> ar(0.0)
```

0.25

```
>>> ar(0.0)
```

0.125

```
>>> ar(0.0)
```

0.0625

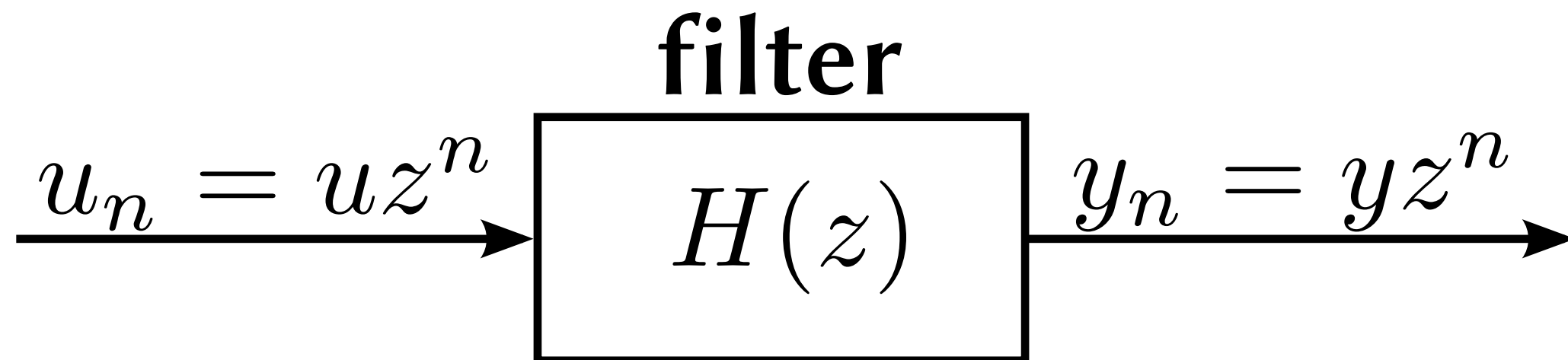
AR for Linear Prediction

Synthesis Filter

```
>>> a = lp(data, m, ...)  
>>> synthesis = AR(a)  
>>> data = synthesis(error)
```

Transfer Function

Given $z \in \mathbb{C}$, the inputs and outputs below



are related by:

$$y = H(z)u$$

FIR: $H(z) = a_0 + a_1 z^{-1} + \cdots + a_{N-1} z^{-N+1}$

AR: $H(z) = \frac{1}{1 - a_1 z^{-1} - \cdots - a_N z^{-N}}$

(I/O) Stability

A filter with a transfer function $H(z)$ is **stable** if:

- any bounded input yields a bounded output,
- the modulus of any pole $H(z)$ is less than 1.

Every FIR is stable, AR filters may be unstable ...

```
>>> ar = AR([1.0, 1.0, 1.0, 1.0])
```

```
>>> ar.poles()
```

```
array([ 1.92756198+0.j, -0.77480411+0.j,  
       -0.07637893+0.81470365j, -0.07637893-0.81470365j])
```

```
>>> max(abs(pole) for pole in ar.poles())
```

```
1.9275619754829254
```

Frequency Response

The output of a filter,

- with a transfer function $H(z)$,
- an input signal $u(t) = A(\cos 2\pi f t + \phi)$,

is $y(t) = A'(\cos 2\pi f t + \phi')$ with

$$A' e^{i\phi'} = H(\exp(i2\pi f \Delta t)) A e^{i\phi'}$$

The function $f \mapsto H(\exp(i2\pi f \Delta t))$, denoted $H(f)$ is the filter **frequency response**.

Its **gain** and **phase** are $|H(f)|$ and $\angle H(f)$.

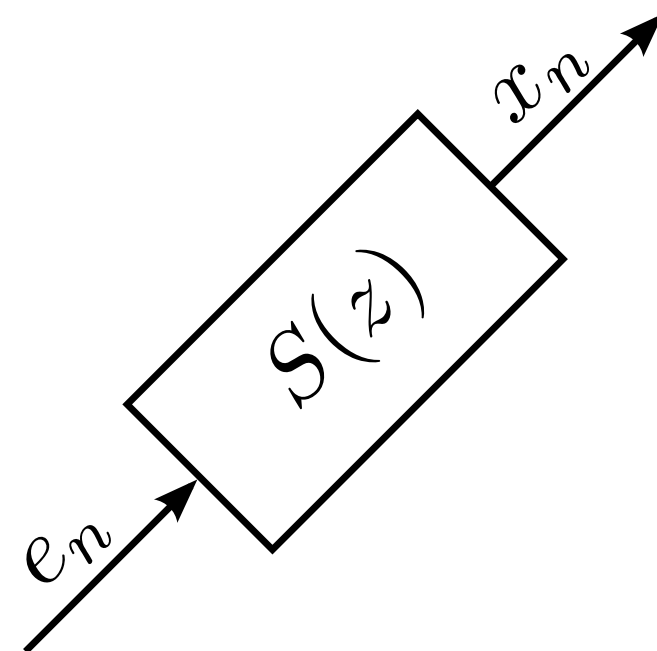
Spectral Analysis

A voice spectrum may be (locally) computed by a (windowed) FFT, or **periodogram**.

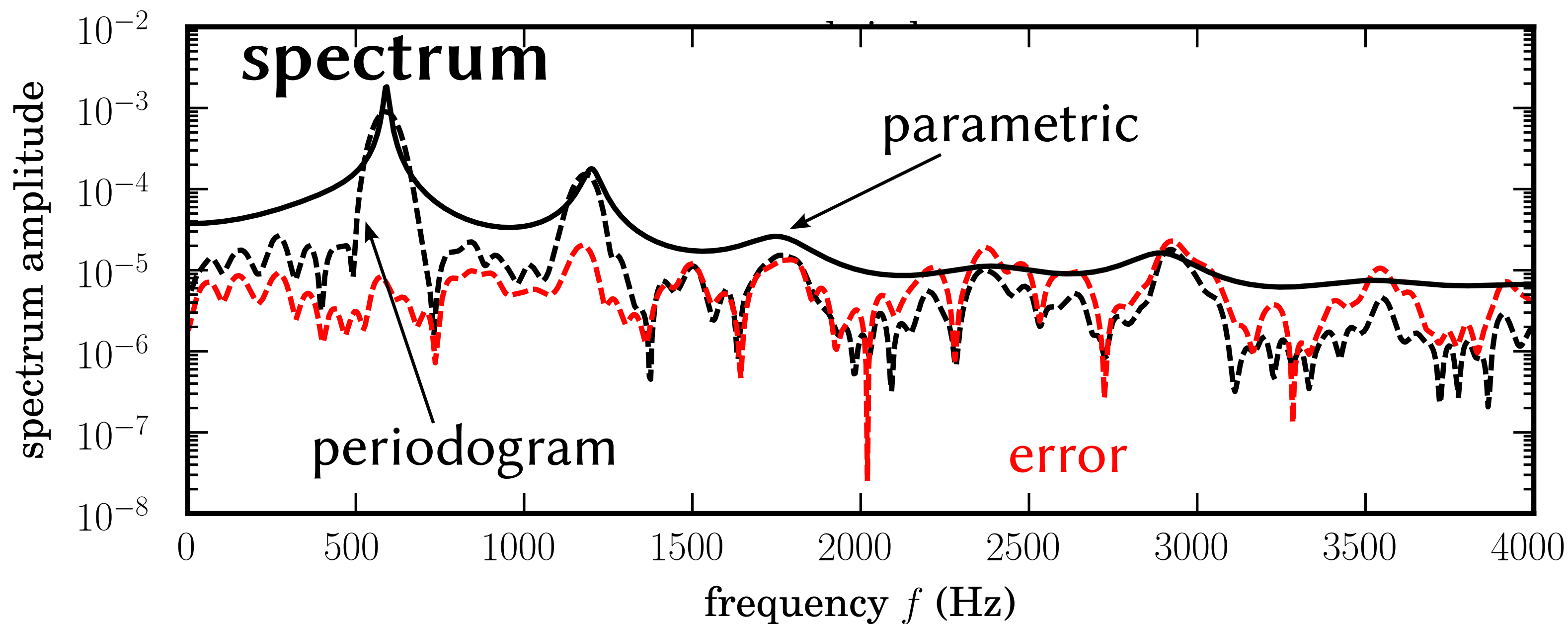
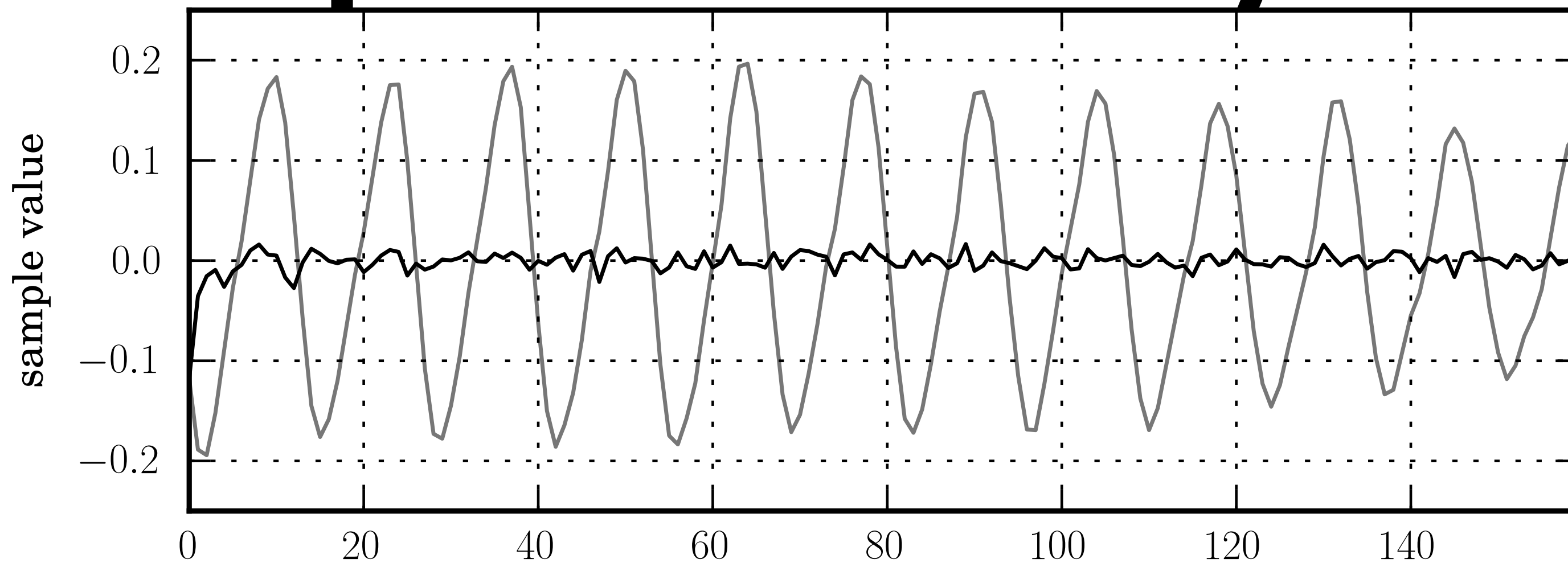
$$|x(f)| = \Delta t \left| \sum_{t \in \mathbb{Z} \Delta t} x(t) \exp(-i2\pi f t) \right|$$

If the prediction of this data has been successful, the error is (almost) white: $|e(f)| \simeq \text{const.}$, and the synthesis filter $S(z)$ provides a **parametric** spectrum estimate:

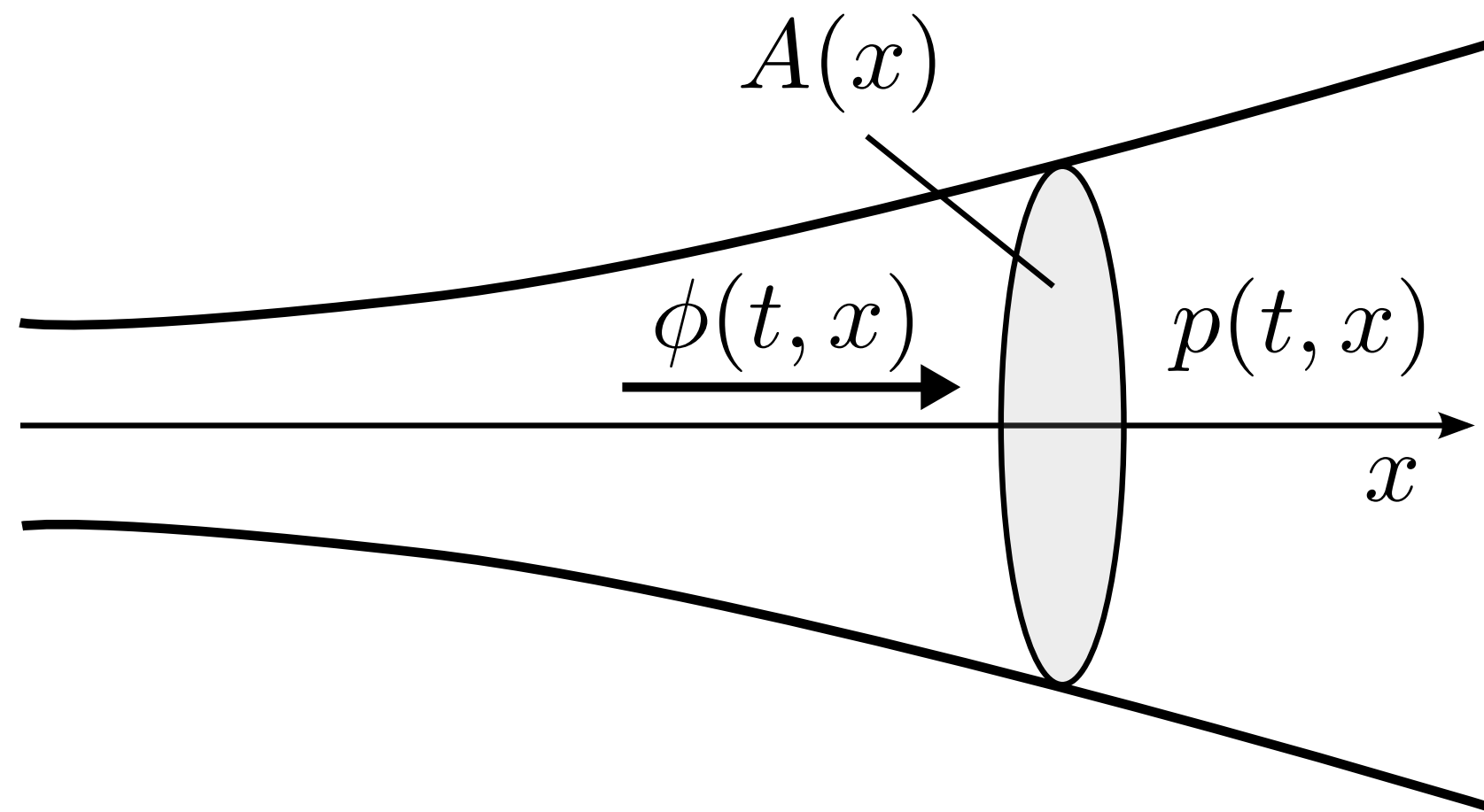
$$|x(f)| = |S(f)| |e(f)| \propto |S(f)|$$



Spectral Analysis



Vocal Tract: Horn Model



A : cross-section

p : pressure

ϕ : air flow

Law of Motion

$$\underbrace{\rho}_{\text{density}} \frac{\partial \phi}{\partial t} = -A \frac{\partial p}{\partial x}$$

Compressibility

$$\underbrace{K}_{\text{bulk modulus}} \frac{\partial \phi}{\partial x} = -A \frac{\partial p}{\partial t}$$

Webster's Equation

Wave Velocity

$$c = \sqrt{\frac{K}{\rho}}$$

Impedance

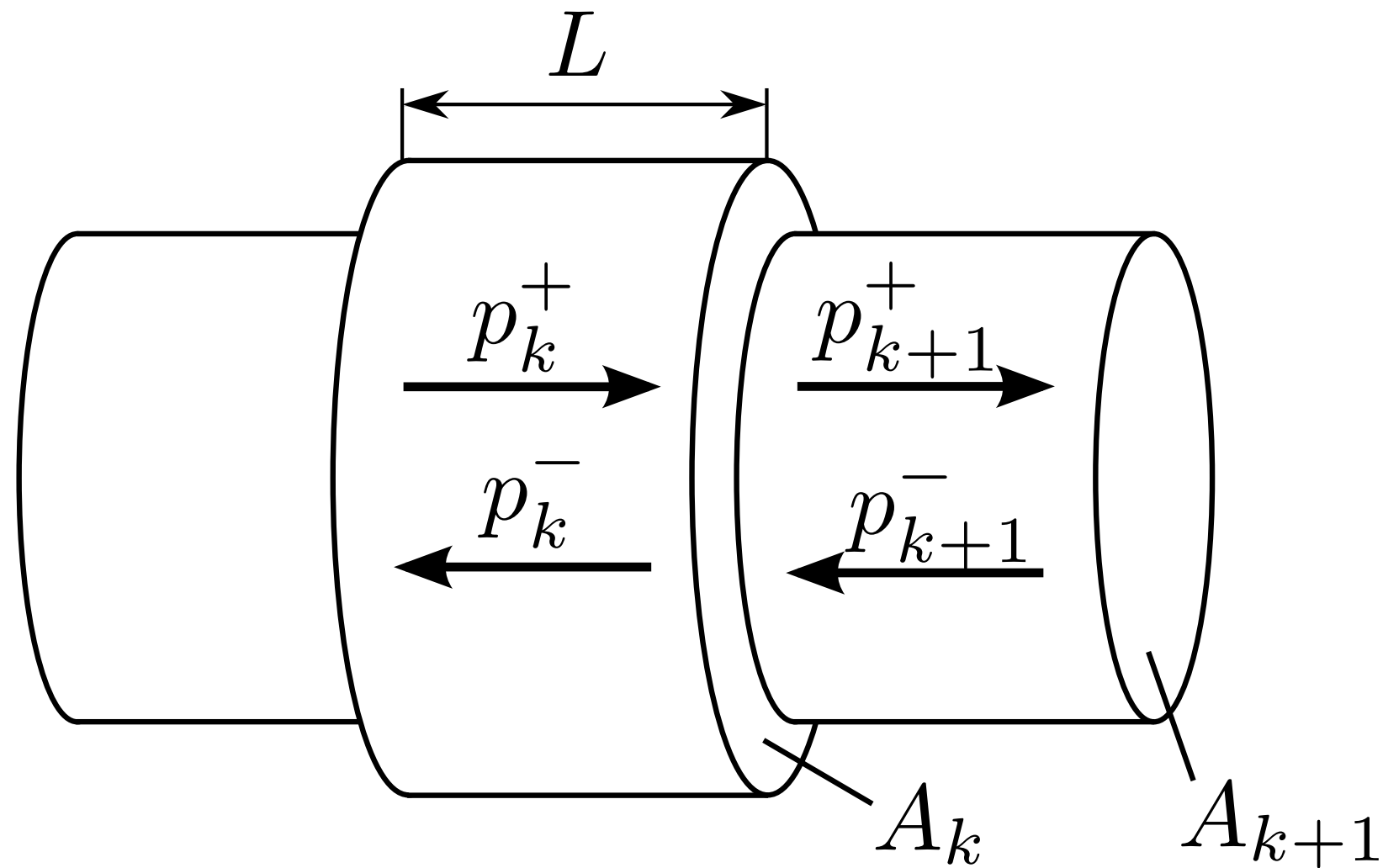
$$Z = \frac{\sqrt{K\rho}}{A}$$

$$\frac{1}{c^2} \frac{\partial^2 p}{\partial t^2} - \frac{1}{A} \frac{dA}{dx} \frac{\partial p}{\partial x} - \frac{\partial^2 p}{\partial x^2} = 0$$

If A is constant:

$$\left| \begin{array}{l} p(t, x) = p^+(x - ct) - p^-(x + ct) \\ \phi(t, x) = \phi^+(x - ct) - \phi^-(x + ct) \end{array} \right. \quad Z = \pm \frac{p^\pm}{\phi^\pm}$$

Discrete Model



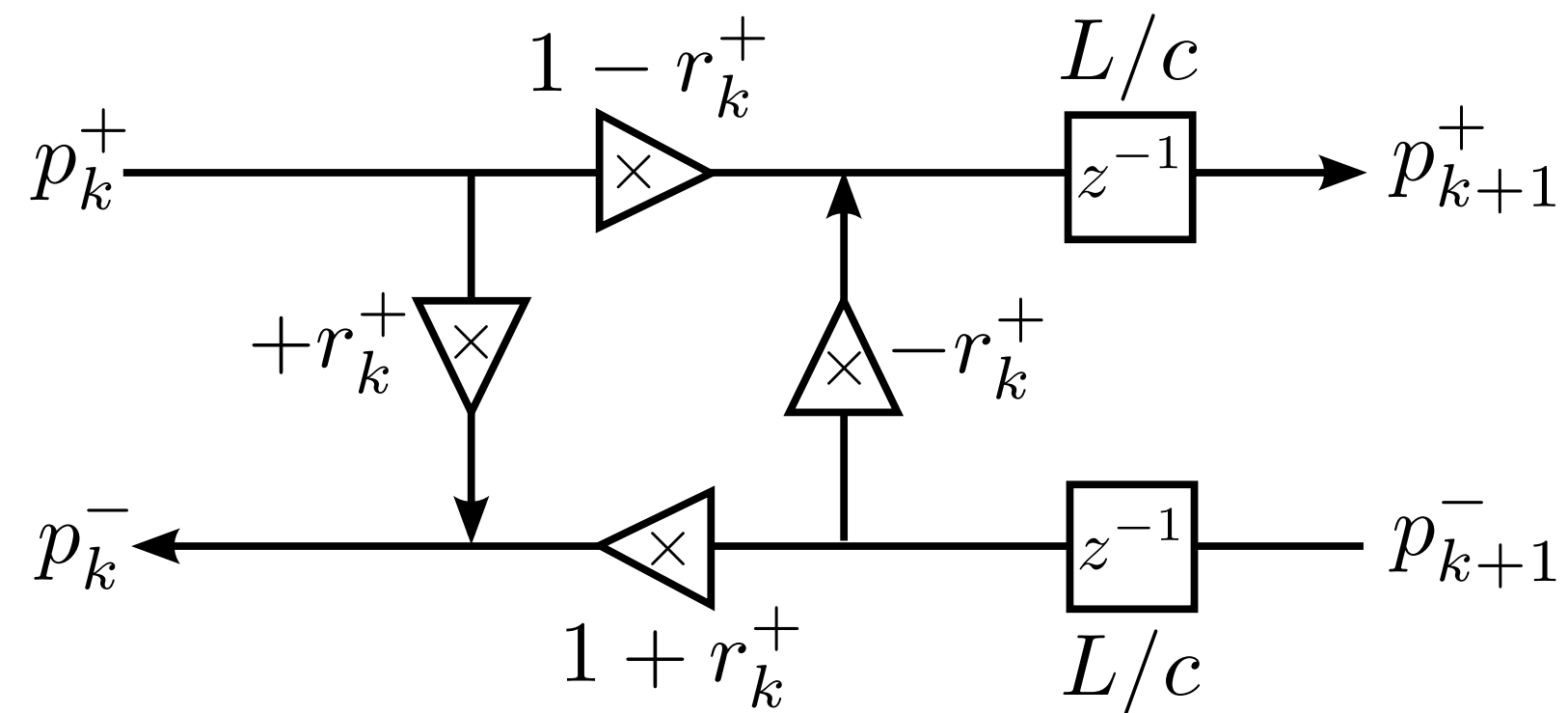
$$\begin{aligned} p_{k+1}^+(x - ct) &= (1 - r_k^+) p_k^+(x - ct) + r_{k+1}^- p_{k+1}^-(x + ct) \\ p_k^-(x + ct) &= (1 - r_{k+1}^-) p_{k+1}^-(x + ct) + r_k^+ p_k^+(x - ct) \end{aligned}$$

Reflection Coefficients:

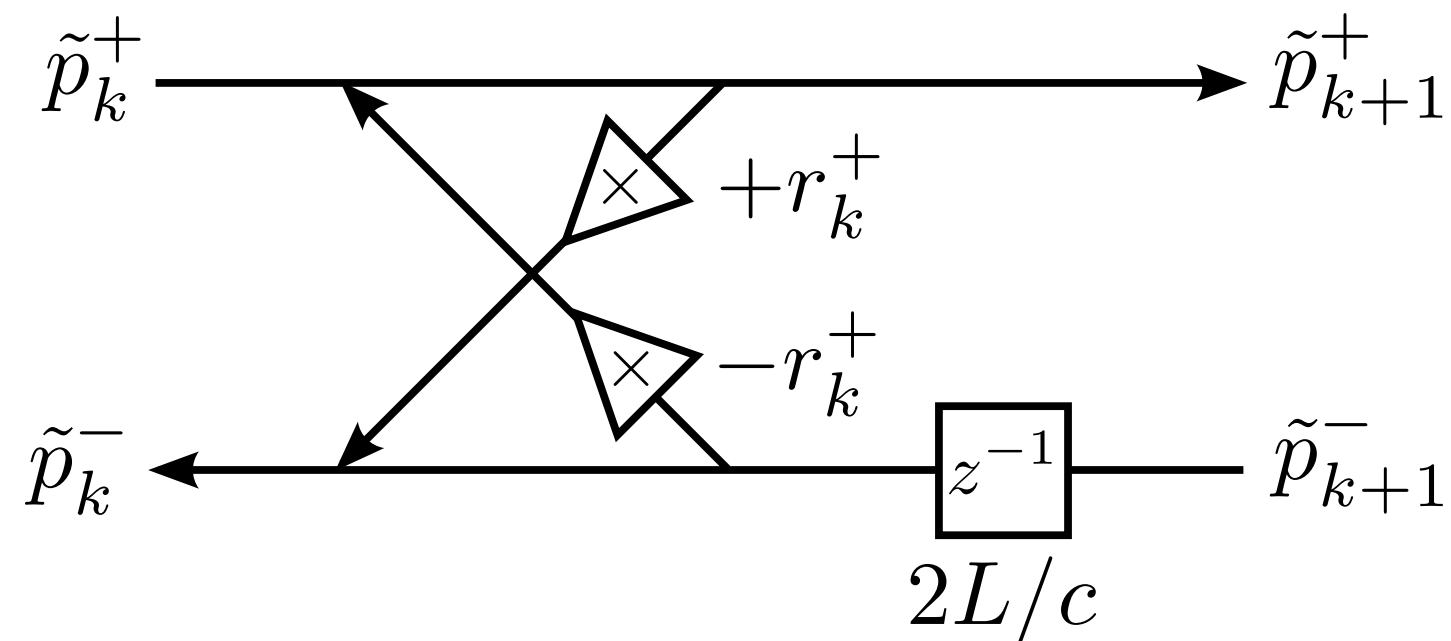
$$r_k^+ = -r_{k+1}^- = \frac{A_{k+1} - A_k}{A_{k+1} + A_k}.$$

Ladder/Lattice Filters

Kelly-Lochbaum Junction



Compensate p^+ for delay and attenuation:



Linear Predictive Coding

AR synthesis filters: lattice filters often replace register-based implementations.

- **Levinson-Durbin** and **Schur** algorithms are fast and provide directly the reflection coefficients that match the experimental data.

$$r_k^+ \longleftrightarrow a_i$$

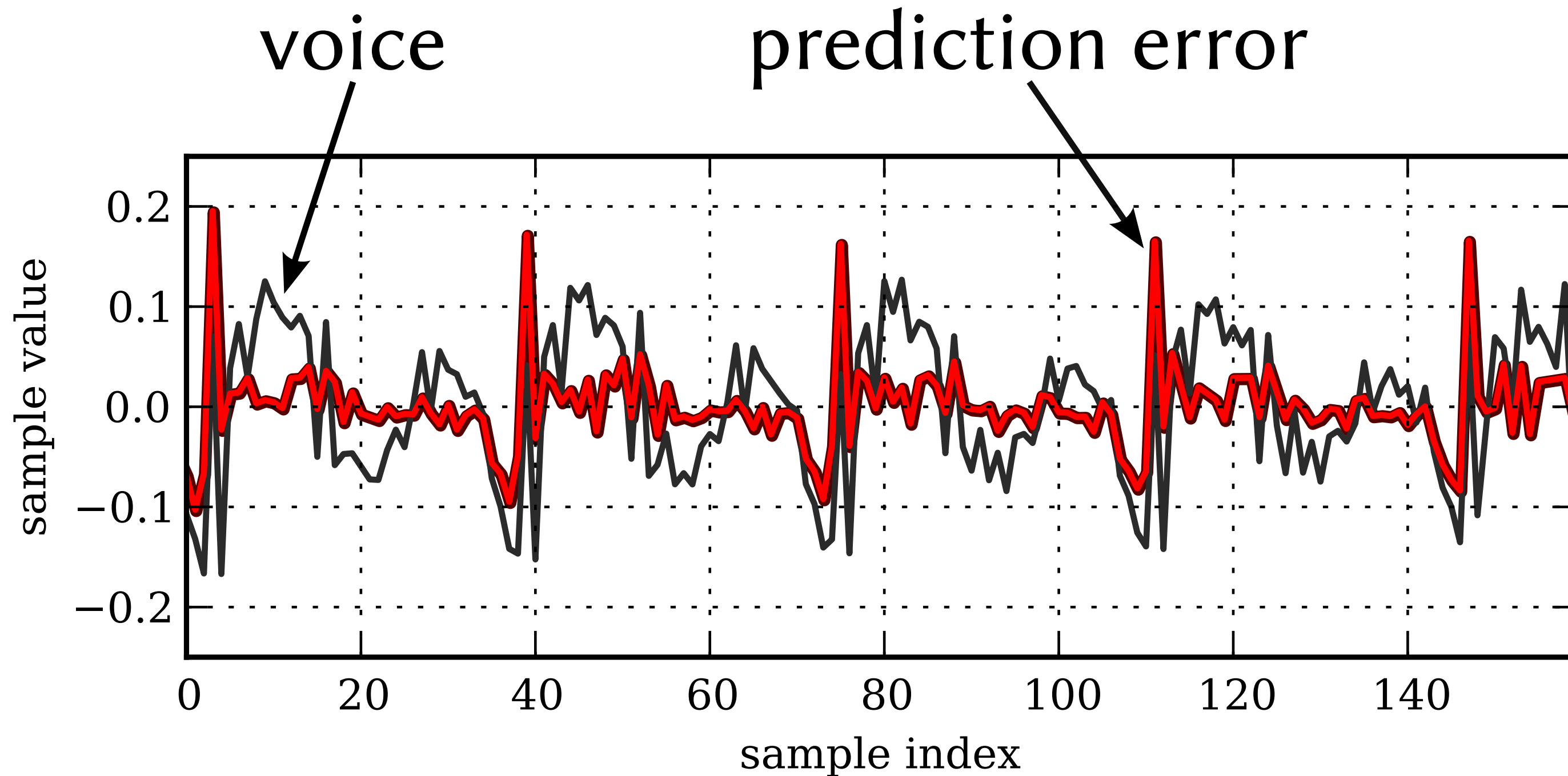
- Synthesis filter are stable iff $|r_k^+| < 1$.

Quantize the area ratio and preserve stability:

$$\frac{1 + r_k^+}{1 - r_k^+} = \frac{A_{k+1}}{A_k}$$

Voice Audio Data

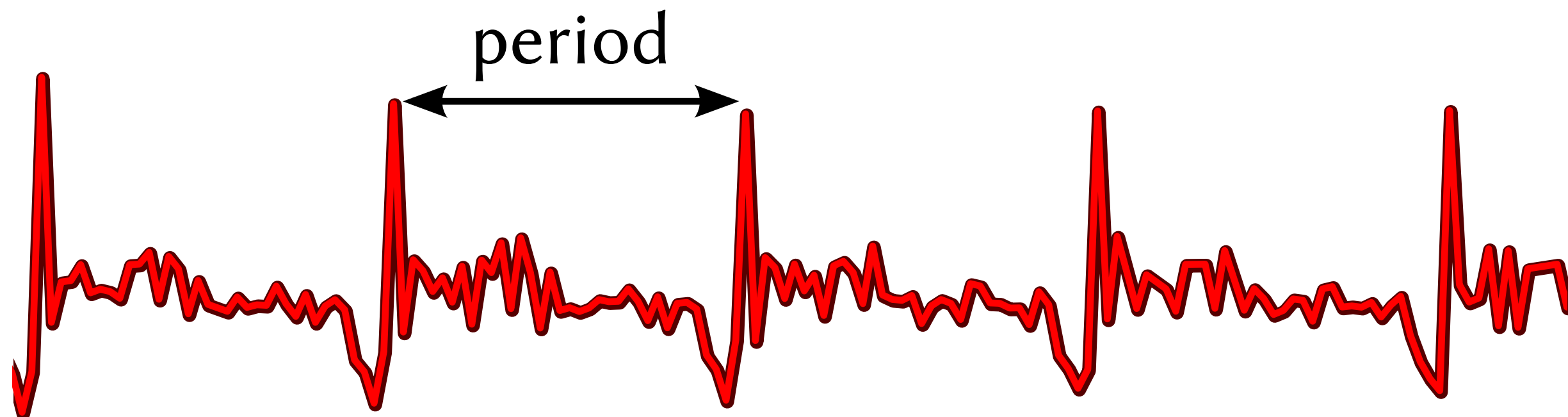
Short-Term Prediction



order 16, autocorrelation

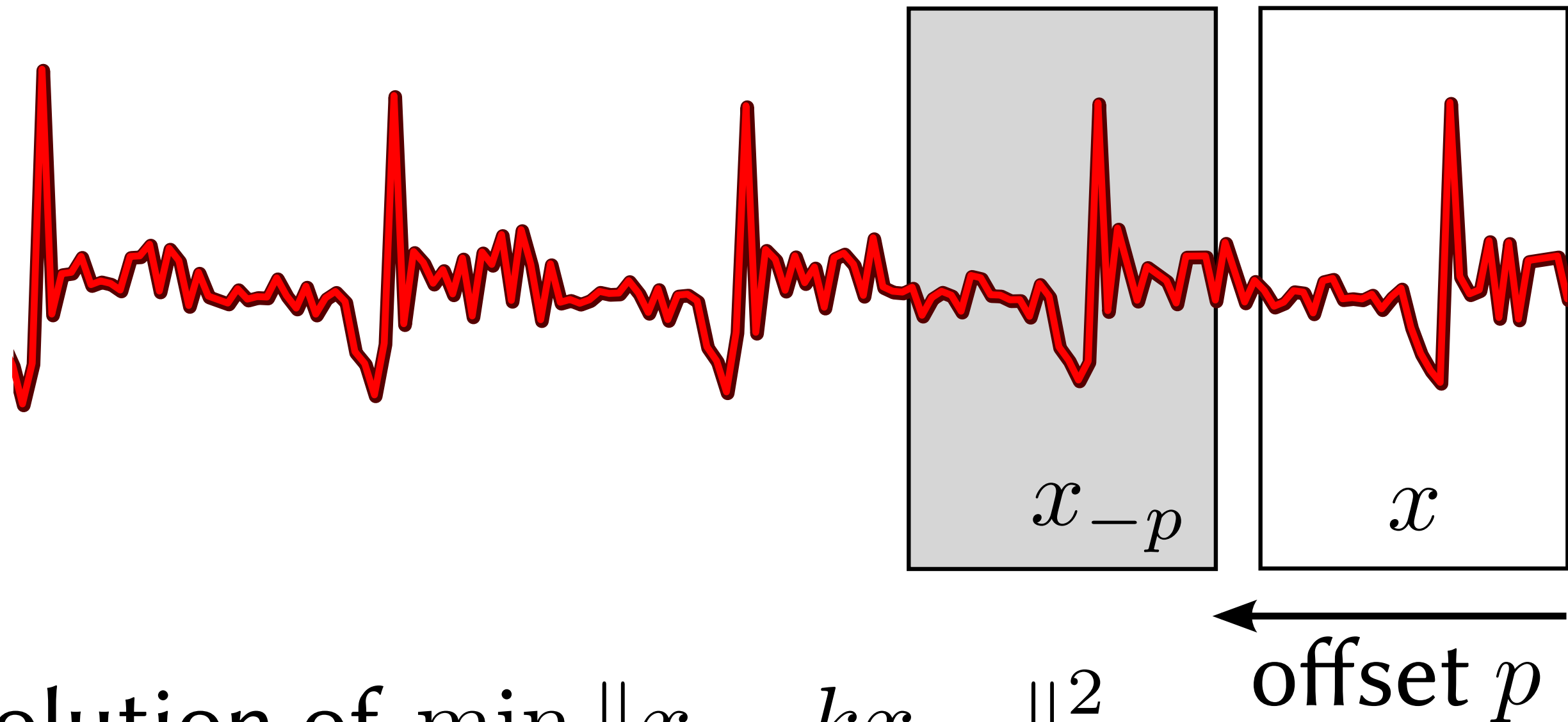
Pitch

Voiced sounds are identified by prediction errors that are sequences of impulses.



The **pitch** frequency is the inverse of the (pseudo-)period.

Pitch Detection

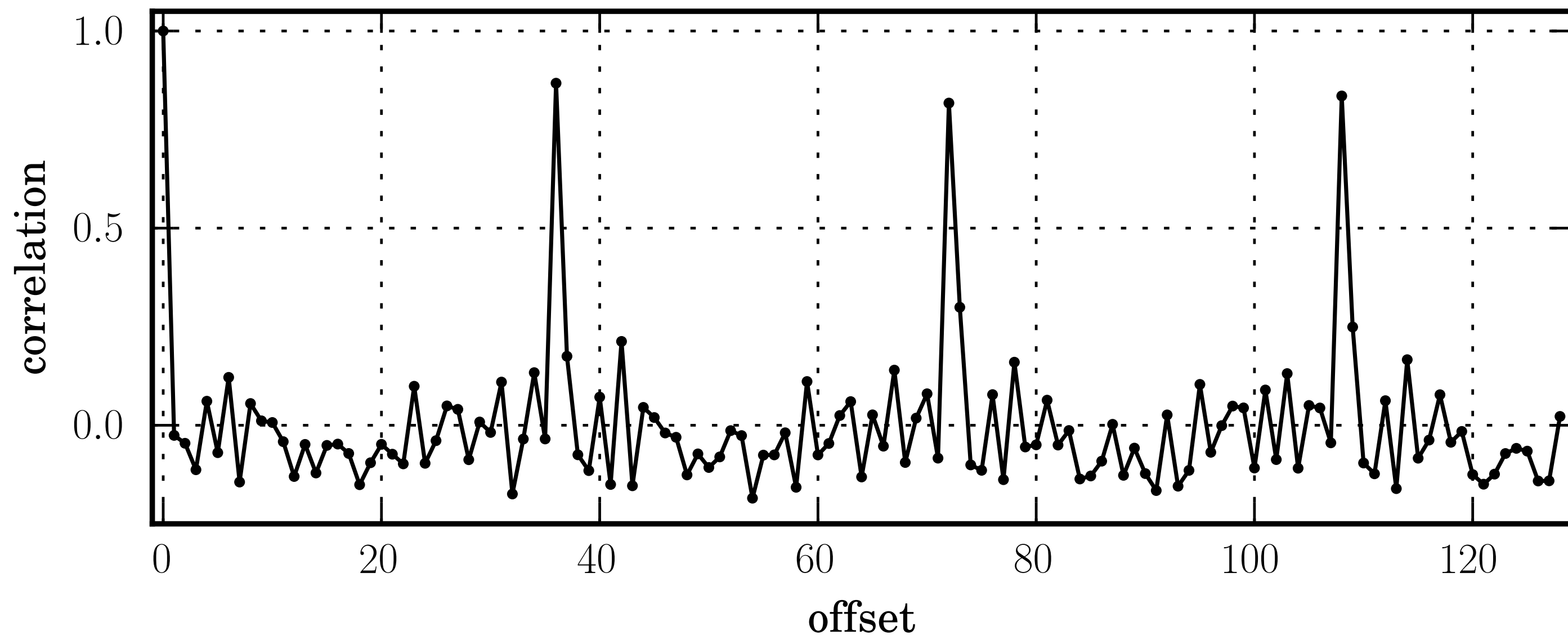


The solution of $\min_{p,k} \|x - kx_{-p}\|^2$

is given by

$$\left| \begin{array}{l} p = \arg \max_p \left| \left\langle \frac{x_{-p}}{\|x_{-p}\|}, \frac{x}{\|x\|} \right\rangle \right| \\ k = \frac{\langle x_{-p}, x \rangle}{\|x\|^2} \end{array} \right|$$

Autocorrelation Function



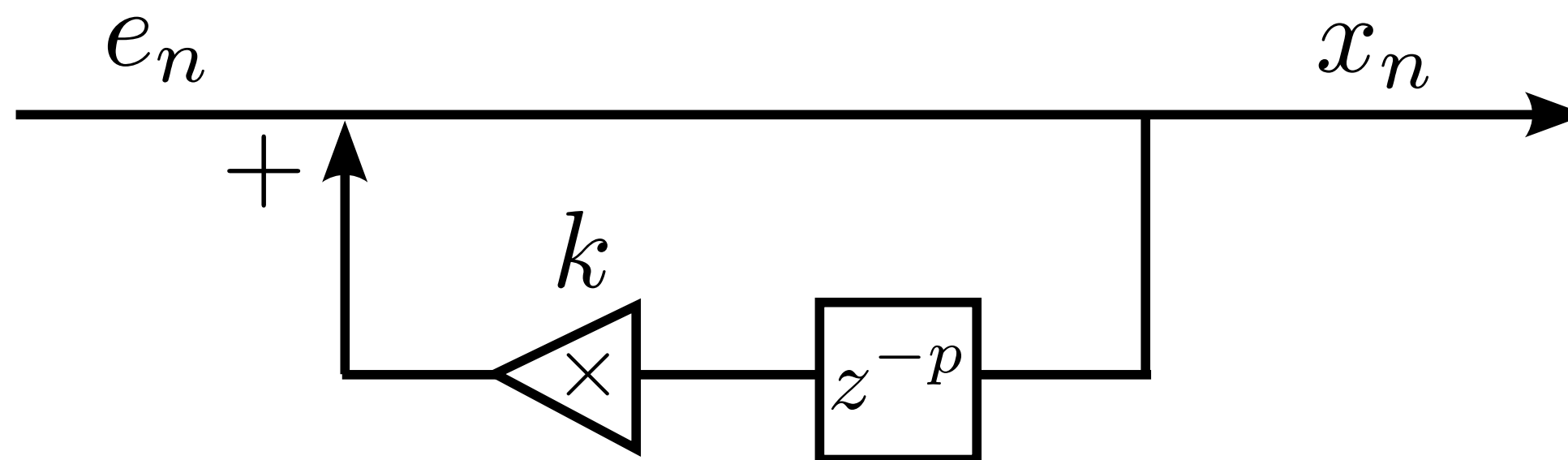
```
def ACF(data, frame_length):  
    def normalize(x):  
        return x / norm(x)  
    m, n = frame_length, len(data)  
    A = array([normalize(data[n-m-i:n-i]) for i in range(n-m+1)])  
    return dot(A, normalize(data[-frame_length:]))
```

Long-Term Prediction

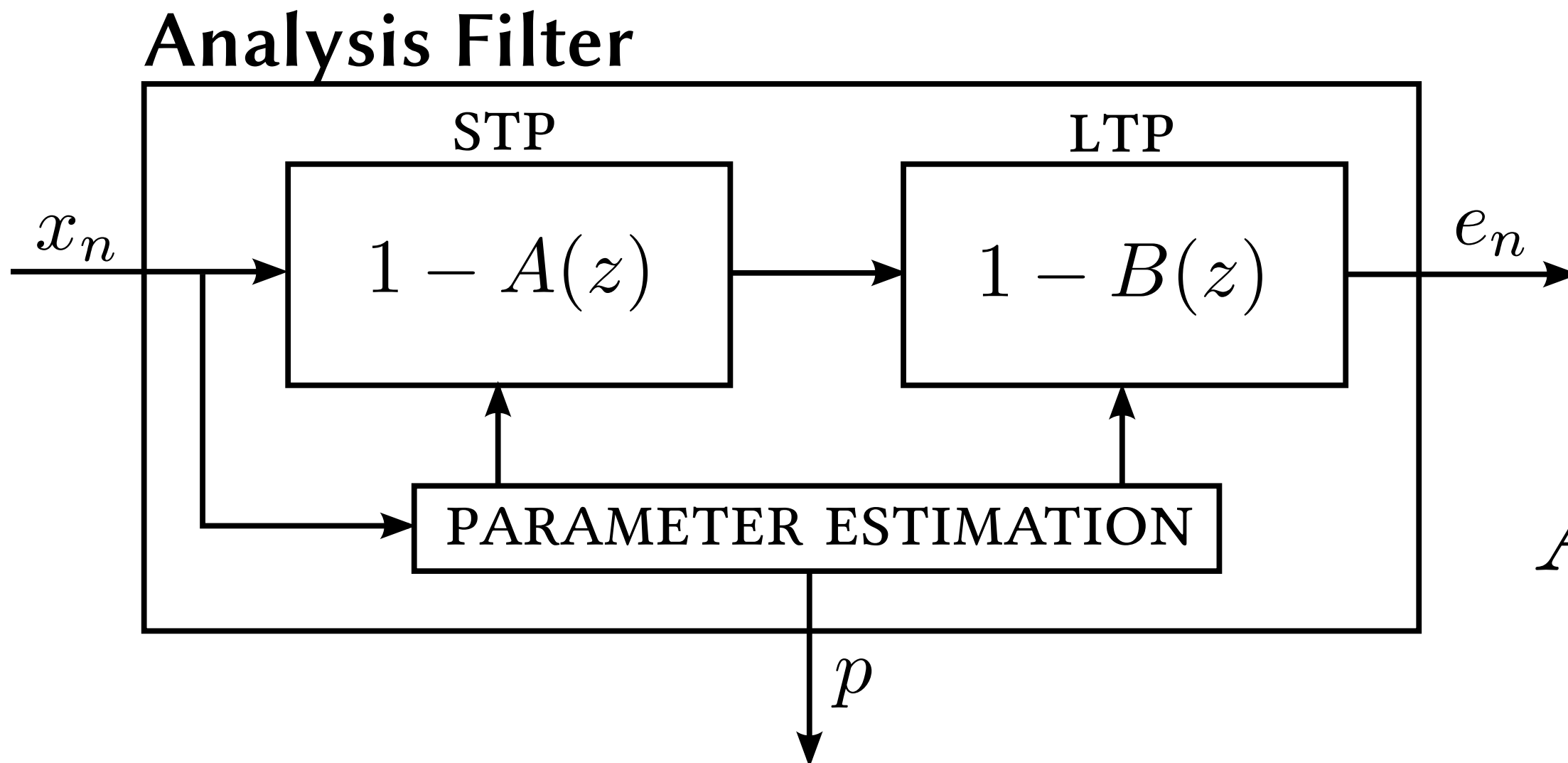
LTP residual

$$x_n = kx_{n-p} + e_n$$

LTP Synthesis (AR) Filter

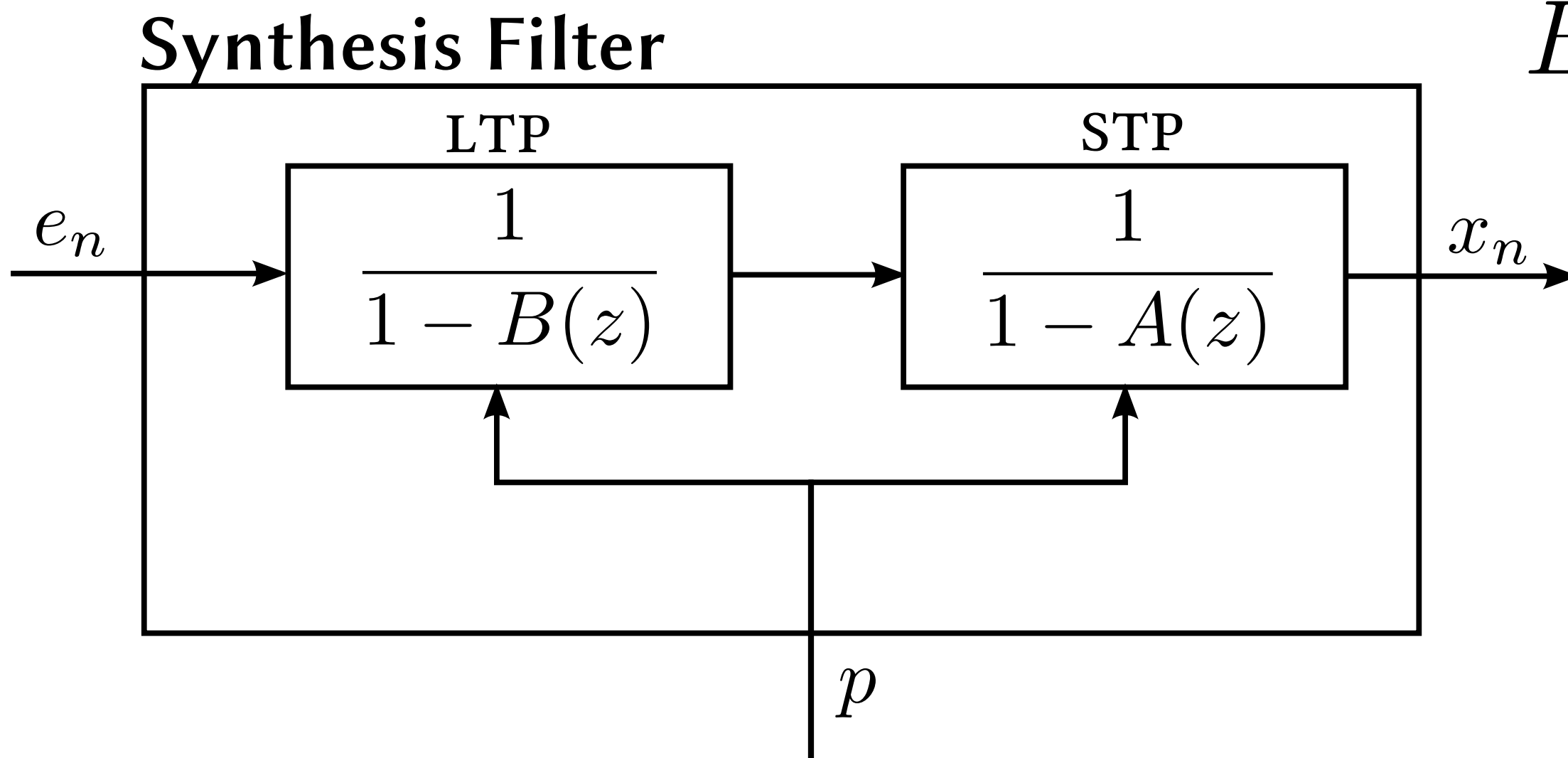


Linear Predictive Coding

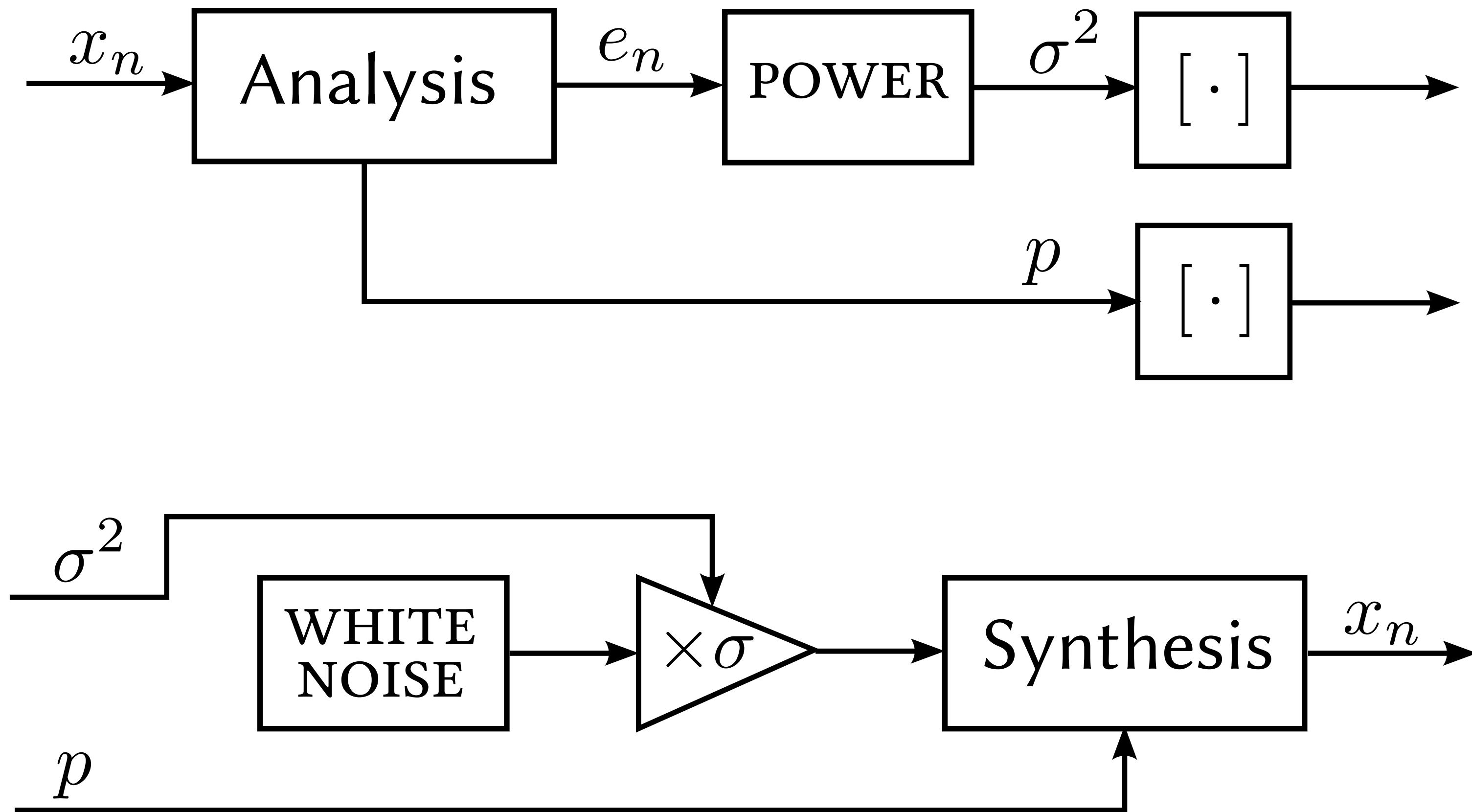


$$A(z) = \sum_{i=1}^m a_i z^{-i}$$

$$B(z) = k z^{-p}$$

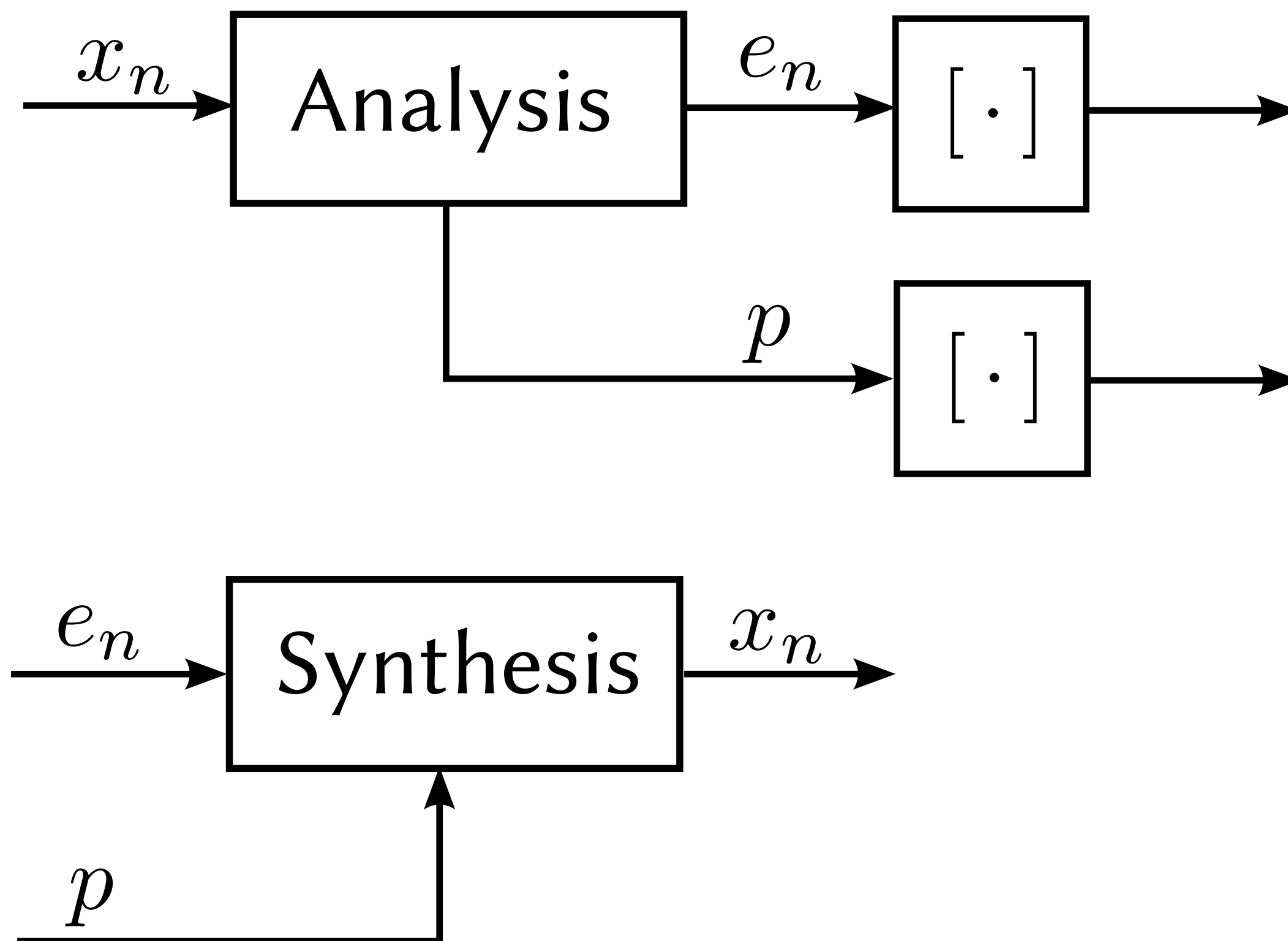


"Pure" LPC



APC / RELP

Adaptative Predictive Coding
Residual-Excited Linear Prediction



CELP

Code-Excited Linear Prediction

