

LINEAR PREDICTION

Linear prediction is an application of LMMSE estimation theory that is widely used in speech processing, spectral estimation, and elsewhere.

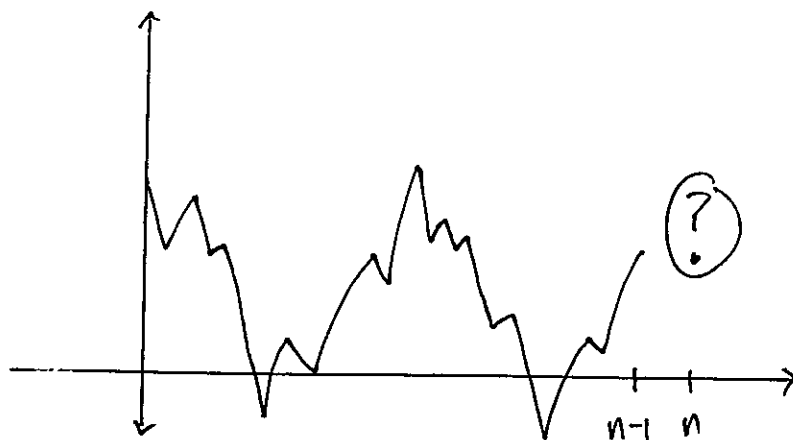
Let $\{x[n]\}$ be a zero-mean, WSS random process with ACF

$$r_{xx}[k] = E\{x[n]x[n+k]\}$$

We observe

$$\underline{x} = [x[n-1] \dots x[n-p]]^T$$

and our task is to predict the next value $x[n]$.



Let's view $x[n]$ as an unknown parameter to be estimated:

$$\theta = x[n]$$

A linear estimator for $x[n]$ has the form

$$\hat{\theta} = \hat{x}[n] = \sum_{k=1}^p h_p[k] x[n-k].$$

We know that the optimal predictor coefficients must satisfy the Wiener-Hopf equations

$$R_{xx} \underline{h}_p = R_{x\theta}$$

where $\underline{h}_p = [h_p[1] \dots h_p[p]]^T$.

To simplify notation, denote

$$r_k = r_{xx}[k].$$

Then

$$R_{xx} = E\{\underline{x} \underline{x}^T\} = \begin{bmatrix} r_0 & r_1 & r_2 & \dots & r_{p-1} \\ r_1 & r_0 & r_1 & \dots & r_{p-2} \\ r_2 & r_1 & r_0 & \dots & \vdots \\ \vdots & \vdots & \dots & \ddots & \vdots \\ r_{p-1} & r_{p-2} & \dots & \dots & r_0 \end{bmatrix} =: R_p$$

Toeplitz \nearrow

and

$$R_{x_0} = E \{ \underline{x} \cdot x[n] \} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_p \end{bmatrix} =: \underline{r}_p$$

Then the optimal linear predictor satisfies

$$R_p \cdot \underline{h}_p = \underline{r}_p$$



$$\begin{bmatrix} r_0 & r_1 & \dots & r_{p-1} \\ r_1 & r_0 & \dots & r_{p-2} \\ \vdots & \vdots & \ddots & \vdots \\ r_{p-1} & r_{p-2} & \dots & r_0 \end{bmatrix} \begin{bmatrix} h_p[1] \\ h_p[2] \\ \vdots \\ h_p[p] \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_p \end{bmatrix}$$

Mean-Squared Prediction Error

$$E \{ (x[n] - \hat{x}[n])^2 \}$$

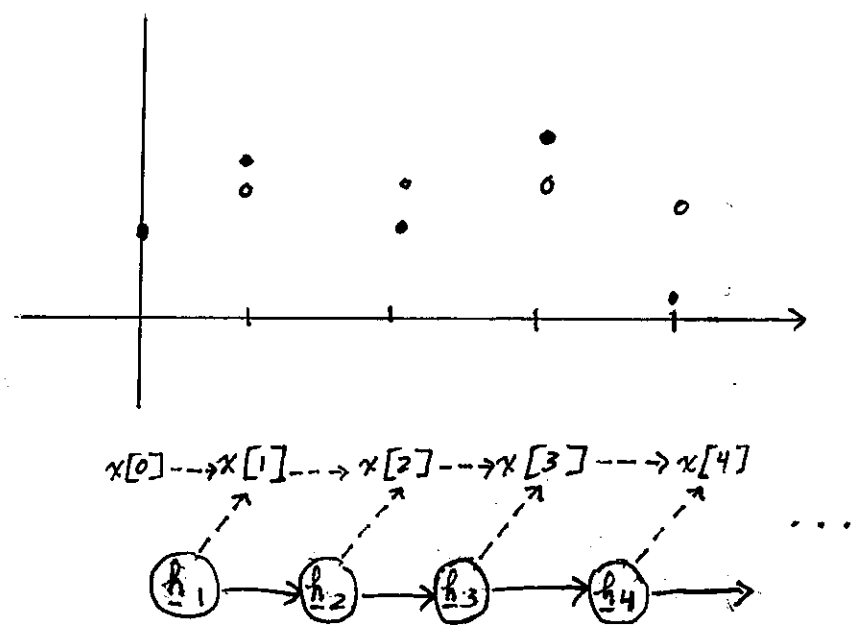
(a)

=

The Levinson-Durbin Algorithm

The Levinson-Durbin algorithm exploits the Toeplitz structure of R_p to compute all of the predictor coefficients $\underline{h}_1, \dots, \underline{h}_p$ in $O(p^2)$ operations.

The algorithm is recursive and therefore ideal for streaming data:



Key |

- recursive update
- > prediction
- observed value
- predicted value

Base of recursion

The WH equations for $p=1$ are

$$[r_0] \cdot [h, [1]] = [r_1]$$

$$\Rightarrow h, [1] = \frac{r_1}{r_0}$$

This will initialize the recursive algorithm

General recursion

We wish to express

$$\frac{h}{r_p} = \begin{bmatrix} \frac{h}{r_{p-1}} \\ 0 \end{bmatrix} + \begin{bmatrix} \frac{d_{p-1}}{k_p} \end{bmatrix}$$

The goal is to find $\frac{d_{p-1}}{k_p}$.

Our approach is to exploit the recursive structure of the WH equations.

According to the WH equations

$$R_p \cdot \underline{h}_p = \underline{r}_p.$$



$$\begin{bmatrix} r_0 & r_1 & \dots & r_{p-1} \\ r_1 & r_0 & \dots & r_{p-2} \\ \vdots & \vdots & \ddots & \vdots \\ r_{p-1} & r_{p-2} & \dots & r_0 \end{bmatrix} \begin{bmatrix} h_p(1) \\ h_p(2) \\ \vdots \\ h_p(p) \end{bmatrix} = \begin{bmatrix} r_1 \\ r_2 \\ \vdots \\ r_p \end{bmatrix}$$

Now observe

$$R_p = \begin{bmatrix} R_{p-1} & \underline{\tilde{r}}_{p-1} \\ \underline{\tilde{r}}_{p-1}^T & r_0 \end{bmatrix}, \quad \underline{\tilde{r}}_p = \begin{bmatrix} r_{p-1} \\ r_p \end{bmatrix}$$

where $\underline{\tilde{r}}_p := "$ \underline{r}_p upside-down." Plugging in

the recursive formula for \underline{h}_p we get

$$\begin{bmatrix} R_{p-1} & \underline{\tilde{r}}_{p-1} \\ \underline{\tilde{r}}_{p-1}^T & r_0 \end{bmatrix} \cdot \left\{ \begin{bmatrix} \underline{h}_{p-1} \\ 0 \end{bmatrix} + \begin{bmatrix} \underline{d}_{p-1} \\ k_p \end{bmatrix} \right\} = \begin{bmatrix} r_{p-1} \\ r_p \end{bmatrix}$$

This gives us a system of equations

$$R_{p-1} \underline{h}_{p-1} + R_{p-1} \underline{d}_{p-1} + \tilde{r}_{p-1} k_p = \underline{r}_{p-1} \quad (1)$$

$$\tilde{r}_{p-1}^T \underline{h}_{p-1} + \tilde{r}_{p-1}^T \underline{d}_{p-1} + r_0 k_p = r_p \quad (2)$$

Recall: we're assuming \underline{h}_{p-1} known and we need to solve for \underline{d}_{p-1} , k_p .

How can we simplify the first equation?

$$R_{p-1} \underline{h}_{p-1} = \underline{r}_{p-1} \implies \underline{d}_{p-1} = -k_p R_{p-1}^{-1} \tilde{\underline{r}}_{p-1}$$

Exercise | Show that $R_{p-1}^{-1} \tilde{\underline{r}}_{p-1} = \tilde{\underline{h}}_{p-1}$, or

equivalently, $R_{p-1} \tilde{\underline{h}}_{p-1} = \tilde{\underline{r}}_{p-1}$. For concreteness,

you may wish to consider $p = 3$ or 4 .

Solution | Suppose

$$A \underline{b} = \underline{c}$$

where A is $m \times n$, \underline{b} is $n \times 1$, \underline{c} is $m \times 1$.

When is $A \tilde{\underline{b}} = \tilde{\underline{c}}$?

If $A = (a_{ij})$, then define

$$\tilde{A} = (\tilde{a}_{ij})$$

where $\tilde{a}_{ij} = a_{m-i, n-j}$. Then

$$A \underline{b} = \underline{c} \iff \sum_{j=1}^n a_{ij} b_j = c_i \quad \forall i$$

$$\iff \sum_{j=1}^n a_{i, n-j} b_{n-j} = c_i \quad \forall i$$

$$\iff \sum_{j=1}^n a_{m-i, n-j} b_{n-j} = c_{m-i} \quad \forall i$$

$$\iff \sum_{j=1}^n \tilde{a}_{ij} \tilde{b}_j = \tilde{c}_i$$

$$\iff \tilde{A} \tilde{\underline{b}} = \tilde{\underline{c}}$$

So we need $A = \tilde{A}$. This is true when

A is symmetric and Toeplitz, as is the case

for R_{p-1} .

Conclusion:

$$\boxed{d_{p-1} = -k_p \tilde{h}_{p-1}}$$

Thus far we have shown

$$\underline{h}_p = \begin{bmatrix} \underline{h}_{p-1} \\ 0 \end{bmatrix} + \begin{bmatrix} d_{p-1} \\ k_p \end{bmatrix} = \begin{bmatrix} \underline{h}_{p-1} \\ 0 \end{bmatrix} + k_p \begin{bmatrix} -\tilde{h}_{p-1} \\ 1 \end{bmatrix}.$$

It remains to find k_p .

Recall equation (2):

$$\tilde{r}_{p-1}^T \underline{h}_{p-1} + \tilde{r}_{p-1}^T d_{p-1} + r_0 k_p = r_p.$$

Plugging in $d_{p-1} = -k_p \tilde{h}_{p-1}$ we get

$$\tilde{r}_{p-1}^T \underline{h}_{p-1} + k_p (-\tilde{r}_{p-1}^T \tilde{h}_{p-1} + r_0) = r_p$$

or

$$\boxed{k_p = \frac{r_p - \tilde{r}_{p-1}^T \underline{h}_{p-1}}{-\tilde{r}_{p-1}^T \tilde{h}_{p-1} + r_0}}$$

The two boxed formulas define the general Levinson-Durbin recursion.

Extensions

The L-D algorithm may be extended to other settings including

1. l-step prediction: given $x[n-1], \dots, x[n-p]$, predict $x[n+l]$
2. multiple predictions: given $x[n-1], \dots, x[n-p]$, predict $x[n], \dots, x[n+l]$.

Application: Linear Predictive Coding

Given: A signal $x[0], x[1], \dots, x[N]$

Task: Compress signal (store with as few bits as possible while still providing an accurate representation).

Idea: Store the first value as is, and encode the prediction errors of the remaining values:

$$e[n] = x[n] - \hat{x}[n]$$

where

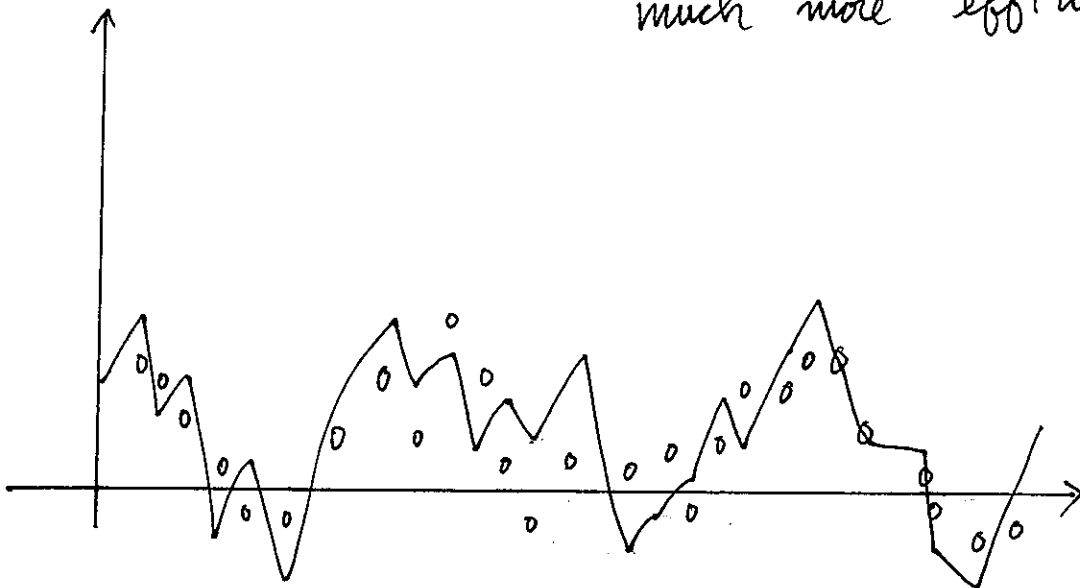
$$\hat{x}[n] = \sum_{k=1}^n h_n[k] x[n-k]$$

In other words, there are two equivalent representations

$$x[0], x[1], \dots, x[N]$$

$$x[0], e[1], \dots, e[N]$$

but the prediction errors have much smaller variance and can therefore be encoded much more efficiently.



LPC is a key ingredient in modern speech processing applications such as compression and synthesis. Since speech is not a stationary process, speech signals are encoded in short blocks and signal characteristics are updated for each block (otherwise the prediction errors would get really big).

Summary

- Linear prediction: application of LMMSE theory
- WSS process \Rightarrow Toeplitz auto-covariance matrix
- Toeplitz ACV matrix \Rightarrow Levinson-Durbin algorithm for fast matrix inversion and predictor coefficient updates.
- Important application: LPC, widely used in speech modeling.

Key a.

$$E \{ (x[n] - \hat{x}[n])^2 \}$$

$$= E \{ (x[n] - \underline{h}_p^T \underline{x}) (x[n] - \underline{h}_p^T \underline{x}) \}$$

$$= E \{ (x[n] - \underline{h}_p^T \underline{x}) \cdot x[n] \}$$

$$= E \{ x[n]^2 \} - \underline{r}_p^T R_p^{-1} E \{ \underline{x} x[n] \}$$

$$= r_0 - \underline{r}_p^T R_p^{-1} \underline{r}_p$$

by orthogonality principle