Levinson's Algorithm

Predictive Filters: A flutist blows air across the blow hole to create music. The input to the flute is hissy-sounding air commonly described as "white noise". The output is a musical note. The input is random in the sense that it has no predictable structure; the output is quite predictable. **Linear predictive filters** provide a reasonable model for this system. The idea is that, rather than trying to model the geometry of the flute and the differential equation governing the air pressure inside it, we might use the data to derive an empirical model. This idea is similar in sprit to using the tabulated values of a polynomial to deduce the degree and coefficients of the polynomial by finite differences. In the flute problem, we use the waveform to deduce the order and coefficients of a difference equation (or its equivalent).

Suppose a sequence y_i satisfies a "difference" equation with constant coefficients:

$$a_0 y_n + a_1 y_{n-1} + a_2 y_{n-2} + \cdots + a_p y_{n-p} = f_n \qquad \forall\ n \geq p.$$

Remark: The "difference" equation has been expanded to remove all the differences, collecting terms in the function y_j rather than differences $y_j - y_k$.

Remark: Choosing to index the a_i as above means the model is a convolution $(\mathbf{a} * \mathbf{y})_n = \mathbf{f}_n\ \forall\ n \geq p$. Engineers call convolutions **filters**.

In mathematics classes, the model parameters a_i are known, and we ask for the solution y_j in terms of initial values $y_0, y_1, \cdots, y_{p-1}$. Predictive filters address the dual problem: (a sample of) the y_j are known, and we ask for the coefficients a_i and the degree p of the difference equation.

The "driving force" f_n in the flute problem is "small" compared to the output. This suggests we solve (in the least-squares sense) the homogeneous problem with $f_n = 0$. Linear algebra guarantees that if $\mathbf{a} = [\,a_0 \quad a_1 \quad \cdots \quad a_p\,]$ is a solution to the homogeneous equations, then so is $\lambda\mathbf{a}$ for any scalar λ. For predictive problems, the most convenient constraint on the coefficients is $a_0 = 1$. Then the prediction of y_n based on the previous values of y_j is

$$y_{n+1} = -(a_1 y_n + a_2 y_{n-1} + \cdots + a_p y_{n-p+1}).$$

Remark: In the grand scheme of things, the negative sign is best left here rather than choosing $a_0 = -1$ to eliminate it from the predictions.

Remark: Predictive filters should not require data from the future in order to predict the future. Filters (= convolutions) that depend only on values from the past are called **causal** in the engineering literature.

Optimal Predictors: Given a data set $\{y_0, y_1, \ldots, y_N\}$ and a positive integer p, a matrix formulation of the predictive filter problem is

$$\underbrace{\begin{bmatrix} y_0 & y_1 & \cdots & y_{p-1} & y_p \\ y_1 & y_2 & \cdots & y_p & y_{p+1} \\ y_2 & y_3 & \cdots & y_{p+1} & y_{p+2} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ y_{N-p} & y_{N-p+1} & \cdots & y_{N-1} & y_N \end{bmatrix}}_{Y} \begin{bmatrix} a_p \\ a_{p-1} \\ \vdots \\ a_1 \\ a_0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}. \qquad (OP)$$

Wiener's optimal p^{th}-order predictive filter solves this problem in the least-squares sense, subject to the constraint $a_0 = 1$. In the flute problem, we called the residual f_n, but since the residual of the least-squares problem is supposed to be small, we change the name to ϵ_n. Solve the homogeneous problem $\mathbf{a} * \mathbf{y} = 0$ in

the least-squares sense and the resulting equation isn't homogeneous (because the residual isn't identically zero), it's the non-homogenous equation $\mathbf{a} * \mathbf{y} = \epsilon$.

Remark: The statistical viewpoint is that the residual $\mathbf{a} * \mathbf{y} = \epsilon$ is just noise — white (or some other color) noise, if you believe in the statistics. From this viewpoint, the noise ϵ stimulates the filter to give the response y according to the "difference" equation

$$y_n = \epsilon_n - a_1 y_{n-1} - a_2 y_{n-2} - \cdots - a_p y_{n-p}.$$

The input ϵ_n may be hissy and random, but the output is harmonic and, therefore, quite regular. In particular, the output is predictable: given a few values $y_{n-1}, y_{n-2}, \cdots, y_{n-p}$, we can make a good guess at the subsequent value y_n. This observation is the basis of voice compression, voice recognition, and voice synthesis algorithms: use the voice itself to model the vocal tract, then describe the voice using the model parameters. Any white-noise-like input will generate an output that sounds like the speaker. (Getting high-quality results requires more work, but the prediction filter is the ubiquitous starting point in each of these problems. A very good reference is "Linear Prediction of Speech" by J.D. Markel and A.H. Grey, Springer Verlag, 1976.)

The optimal filter minimizes

$$\|Y\mathbf{a}\|^2 = \mathbf{a}^T Y^T Y \mathbf{a}$$

subject to the constraint $a_0 = 1$. If $\mathbf{a}$ is the optimal solution, and if $\mathbf{h} = [\, h_0 \quad h_1 \quad \cdots \quad 0\,]$ is any permissible increment (so both $\mathbf{a}$ and $\mathbf{a} + \mathbf{h}$ satisfy the constraint), then

$$0 \le \|Y\,[\mathbf{a} + \mathbf{h}]\|^2 - \|Y\mathbf{a}\|^2 = 2\mathbf{h}^T Y^T Y \mathbf{a} + \mathbf{h}^T Y^T Y \mathbf{h}.$$

The first term on the right is linear in $\mathbf{h}$; the second is quadratic. By the usual calculus arguments, the term linear in $\mathbf{h}$ must be zero for all permissible $\mathbf{h}$:

$$\mathbf{h}^T Y^T Y \mathbf{a} = 0 \;\; \forall \;\; \mathbf{h} = [\, h_0 \quad h_1 \quad \cdots \quad 0\,]^T.$$

Let $\mathbf{y}_j = [\, y_j \quad y_{j+1} \quad \cdots \quad y_{j+N-p}\,]^T$ be column j of the matrix Y. Since $\mathbf{h}$ is arbitrary except in the last component, the optimal solution satisfies

$$Y^T Y \mathbf{a} = \begin{bmatrix} \mathbf{y}_0^T\mathbf{y}_0 & \mathbf{y}_0^T\mathbf{y}_1 & \cdots & \mathbf{y}_0^T\mathbf{y}_{p-1} & \mathbf{y}_0^T\mathbf{y}_p \\ \mathbf{y}_1^T\mathbf{y}_0 & \mathbf{y}_1^T\mathbf{y}_1 & \cdots & \mathbf{y}_1^T\mathbf{y}_{p-1} & \mathbf{y}_1^T\mathbf{y}_p \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \mathbf{y}_{p-1}^T\mathbf{y}_0 & \mathbf{y}_{p-1}^T\mathbf{y}_1 & \cdots & \mathbf{y}_{p-1}^T\mathbf{y}_{p-1} & \mathbf{y}_{p-1}^T\mathbf{y}_p \\ \mathbf{y}_{p-1}^T\mathbf{y}_0 & \mathbf{y}_{p-1}^T\mathbf{y}_1 & \cdots & \mathbf{y}_{p-1}^T\mathbf{y}_{p-1} & \mathbf{y}_{p-1}^T\mathbf{y}_p \end{bmatrix} \begin{bmatrix} a_p \\ a_{p-1} \\ \vdots \\ a_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ * \end{bmatrix}.$$

Finally, since the last component of $\mathbf{a}$ is $a_0 = 1$, the square norm of the residual is

$$\|Y\mathbf{a}\|^2 = [\, a_p \quad a_{p-1} \quad \cdots \quad a_1 \quad 1\,]\, Y^T Y \mathbf{a} = [\, a_p \quad a_{p-1} \quad \cdots \quad a_1 \quad 1\,] \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ * \end{bmatrix} = *,$$

so the optimal solution satisfies

$$Y^T Y \mathbf{a} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ \|Y\mathbf{a}\|^2 \end{bmatrix}.$$

Stationarity: In many applications, the dot products $\mathbf{y}_i^T\mathbf{y}_j$ depend, statistically, only on the difference $|j-i|$:

$$\mathbf{y}_i^T\mathbf{y}_j = R_{|j-i|}.$$

Signals satisfying this statistical property are called **stationary**, and the R_k are called **correlations**. Stationarity reduces the least-squares problem to

$$\begin{bmatrix} R_0 & R_1 & \cdots & R_{p-1} & R_p \\ R_1 & R_0 & \cdots & R_{p-2} & R_{p-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ R_{p-1} & R_{p-2} & \cdots & R_0 & R_1 \\ R_p & R_{p-1} & \cdots & R_1 & R_0 \end{bmatrix} \begin{bmatrix} a_p \\ a_{p-1} \\ \vdots \\ a_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ E_p \end{bmatrix}, \qquad (SOP_p)$$

where E_p is the square norm of the residual for the p^{th}-order filter. The matrix is symmetric and Toeplitz. The equation name "(SOP_p)" stands for "stationary optimal predictor of order p".

Remark: If (OP) represents filtering the data (the y_j) to get zero, then (SOP) represents filtering the correlations (the R_j) to get zero (except in the last component). In other words, stationarity exchanges filtering the data for filtering the statistics. Engineers see filters everywhere — we should, too.

Levinson's Algorithm: Levinson's Algorithm solves (SOP_p) using recursion on the filter order p. The first-order problem is

$$\begin{bmatrix} R_0 & R_1 \\ R_1 & R_0 \end{bmatrix} \begin{bmatrix} a_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ E_1 \end{bmatrix} \qquad (SOP_1)$$

and its solution and square residual are

$$\begin{bmatrix} a_1^{(1)} \\ 1 \end{bmatrix} = \begin{bmatrix} -\frac{1}{R_0}R_1 \\ 1 \end{bmatrix} \qquad \text{and} \qquad E_1 = R_0\left(1 - \frac{R_1^2}{R_0^2}\right). \qquad (E_1)$$

The superscript $^{(1)}$ indicates that $\mathbf{a}^{(1)}$ is the optimal solution to the first-order problem.

The second-order problem is

$$\begin{bmatrix} R_0 & R_1 & R_2 \\ R_1 & R_0 & R_1 \\ R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} a_2 \\ a_1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ E_2 \end{bmatrix}. \qquad (SOP_2)$$

Iterative least squares seeks a solution which is an increment from the first-order solution:

$$\begin{bmatrix} R_0 & R_1 & R_2 \\ R_1 & R_0 & R_1 \\ R_2 & R_1 & R_0 \end{bmatrix} \left(\begin{bmatrix} 0 \\ a_1^{(1)} \\ 1 \end{bmatrix} + \begin{bmatrix} h_2 \\ h_1 \\ 0 \end{bmatrix} \right) = \begin{bmatrix} R_2 + a_1^{(1)}R_1 \\ 0 \\ E_1 \end{bmatrix} + \begin{bmatrix} R_0 & R_1 & R_2 \\ R_1 & R_0 & R_1 \\ R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} h_2 \\ h_1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ E_2 \end{bmatrix}. \qquad (L_2)$$

The increment therefore satisfies

$$\begin{bmatrix} R_0 & R_1 \\ R_1 & R_0 \end{bmatrix} \begin{bmatrix} h_2 \\ h_1 \end{bmatrix} = \begin{bmatrix} -R_2 - a_1^{(1)}R_1 \\ 0 \end{bmatrix}. \qquad (h_2)$$

This is just the first-order problem (SOP_1) turned "upside down". The first order solution satisfies

$$\begin{bmatrix} R_0 & R_1 \end{bmatrix} \begin{bmatrix} a_1^{(1)} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \end{bmatrix} \qquad \text{so} \qquad \begin{bmatrix} R_1 & R_0 \end{bmatrix} \begin{bmatrix} 1 \\ a_1^{(1)} \end{bmatrix} = \begin{bmatrix} 0 \end{bmatrix}.$$

In English: reversing the order of both vectors does not change the dot product. Consequently, there is a scalar γ_2 for which

$$\begin{bmatrix} h_2 \\ h_1 \\ 0 \end{bmatrix} = -\gamma_2 \begin{bmatrix} 1 \\ a_1^{(1)} \\ 0 \end{bmatrix},$$

(the negative sign simplifies later formulas). The point is that **the increment is a scalar multiple of the reverse of the solution to the first order problem**.

Notation:

$$\text{The } \textbf{reverse} \text{ of } \mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_m \end{bmatrix}^T \quad \text{is} \quad \tilde{\mathbf{x}} = \begin{bmatrix} x_m \\ x_{m-1} \\ \vdots \\ x_0 \end{bmatrix} = \underbrace{\begin{bmatrix} 0 & 0 & \cdots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 0 & \cdots & 0 \end{bmatrix}}_{J} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_m \end{bmatrix},$$

where J is the "anti-diagonal" matrix. Visually, the order of the components of the vector is simply reversed. If convolution with $\mathbf{a}$ represents *pre*diction, then convolution with $\tilde{\mathbf{a}}$ represents *post*diction. Engineers speak of filtering forward (in time) and backward (in time).

To compute γ_2, use the top row of Equation (h_2):

$$\begin{bmatrix} R_0 & R_1 \end{bmatrix} (-\gamma_2) \begin{bmatrix} 1 \\ a_1^{(1)} \end{bmatrix} = -R_2 - R_1 a_1^{(1)} \qquad \Longrightarrow \qquad \gamma_2 = \frac{R_2 + R_1 a_1^{(1)}}{R_0 + R_1 a_1^{(1)}}. \tag{γ_2}$$

In Filter Language: The denominator $R_0 + R_1 a_1^{(1)}$ is the first-order optimal filter applied forward in time to $\begin{bmatrix} R_0 & R_1 \end{bmatrix}^T$, while the numerator is the first-order optimal filter applied backward in time to $\begin{bmatrix} R_1 & R_2 \end{bmatrix}^T$. The scalar γ_2 is called a **reflection coefficient**. All that remains is to compute the residual's square norm

$$\begin{aligned} E_2 &= E_1 - \begin{bmatrix} R_2 & R_1 & R_0 \end{bmatrix} \begin{bmatrix} h_2 \\ h_1 \\ 0 \end{bmatrix} \\ &= E_1 - \gamma_2 \left(R_2 + R_1 a_1^{(1)} \right) \\ &= E_1 - \gamma_2^2 \left(R_0 + R_1 a_1^{(1)} \right) \\ &= E_1 \left(1 - \gamma_2^2 \right). \end{aligned} \tag{E_2}$$

Remark: Since $E_2 \leq E_1$, the reflection coefficient satisfies $-1 \leq \gamma_2 \leq 1$.

The structure of the solution to (L_2) holds for all filter orders. If we have computed the optimal p^{th}-order filter, which satisfies

$$\underbrace{\begin{bmatrix} R_0 & R_1 & \cdots & R_{p-1} & R_p \\ R_1 & R_0 & \cdots & R_{p-2} & R_{p-1} \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ R_{p-1} & R_{p-2} & \cdots & R_0 & R_1 \\ R_p & R_{p-1} & \cdots & R_1 & R_0 \end{bmatrix}}_{A_p} \underbrace{\begin{bmatrix} a_p^{(p)} \\ a_{p-1}^{(p)} \\ \vdots \\ a_1^{(p)} \\ 1 \end{bmatrix}}_{\mathbf{a}^{(p)}} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ E_p \end{bmatrix}, \tag{$\mathbf{a}^{(p)}$}$$

we may write the optimal $(p+1)^{\text{st}}$-order filter as an increment onto the p^{th}-order filter:

$$\begin{bmatrix} a_{p+1}^{(p+1)} \\ a_p^{(p+1)} \\ a_{p-1}^{(p+1)} \\ \vdots \\ a_1^{(p+1)} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ a_p^{(p)} \\ a_{p-1}^{(p)} \\ \vdots \\ a_1^{(p)} \\ 1 \end{bmatrix} - \gamma_{p+1} \begin{bmatrix} h_{p+1} \\ h_p \\ h_{p-1} \\ \vdots \\ h_1 \\ 0 \end{bmatrix} \tag{h_{p+1}}$$

As in the computation of the 2nd-order filter above, the increment satisfies

$$\underbrace{\begin{bmatrix} R_0 & R_1 & \cdots & R_p & R_{p+1} \\ R_1 & R_0 & \cdots & R_{p-1} & R_p \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ R_p & R_{p-1} & \cdots & R_0 & R_1 \\ R_{p+1} & R_p & \cdots & R_1 & R_0 \end{bmatrix}}_{A_{p+1}} \gamma_{p+1} \begin{bmatrix} h_{p+1} \\ h_p \\ \vdots \\ h_1 \\ 0 \end{bmatrix} = \begin{bmatrix} R_1 a_p^{(p)} + R_2 a_{p-1}^{(p)} + \cdots + R_p a_1^{(p)} + R_{p+1} \\ 0 \\ \vdots \\ 0 \\ E_p - E_{p+1} \end{bmatrix}.$$

The increment must therefore satisfy

$$\underbrace{\begin{bmatrix} R_0 & R_1 & \cdots & R_p \\ R_1 & R_0 & \cdots & R_{p-1} \\ \vdots & \vdots & \ddots & \vdots \\ R_p & R_{p-1} & \cdots & R_0 \end{bmatrix}}_{A_p} \begin{bmatrix} h_{p+1} \\ h_p \\ \vdots \\ h_1 \end{bmatrix} = \begin{bmatrix} R_1 a_p^{(p)} + R_2 a_{p-1}^{(p)} + \cdots + R_p a_1^{(p)} + R_{p+1} \\ \vdots \\ 0 \end{bmatrix}.$$

This is just (SOP_p) "reversed": the non-zero component on the right is at the top of the vector rather than at the bottom. Since A_p is symmetric and Toeplitz, $[\, h_{p+1} \quad h_p \quad \cdots \quad h_1 \,]^T$ is a scalar multiple of the reverse of $\mathbf{a}^{(p)}$:

$$\begin{bmatrix} h_{p+1} \\ h_p \\ \vdots \\ h_1 \end{bmatrix} = -\gamma_{p+1} \tilde{\mathbf{a}}^{(p)}.$$

Levinson's algorithm is, therefore,

$$\mathbf{a}^{(p+1)} = \begin{bmatrix} 0 \\ \mathbf{a}^{(p)} \end{bmatrix} - \gamma_{p+1} \begin{bmatrix} \tilde{\mathbf{a}}^{(p)} \\ 0 \end{bmatrix}. \tag{$\mathbf{a}^{(p+1)}$}$$

The value of γ_{p+1} comes from the top row:

$$[\, R_0 \quad R_1 \quad \cdots \quad R_p \quad R_{p+1} \,] \left(\begin{bmatrix} 0 \\ \mathbf{a}^{(p)} \end{bmatrix} - \gamma_{p+1} \begin{bmatrix} \tilde{\mathbf{a}}^{(p)} \\ 0 \end{bmatrix} \right) = 0.$$

Spelled out:

$$R_1 a_p^{(p)} + R_2 a_{p-1}^{(p)} + \cdots + R_p a_1^{(p)} + R_{p+1} = \gamma_{p+1} \underbrace{\left(R_0 + R_1 a_1^{(p)} + \cdots + R_p a_p^{(p)} \right)}_{E_p}. \tag{γ_{p+1}}$$

In Filter-ese: the left side is the result of filtering $[\, R_1 \quad \cdots \quad R_p \quad R_{p+1} \,]^T$ forward in time, and the right side is the result of filtering $[\, R_0 \quad \cdots \quad R_{p-1} \quad R_p \,]^T$ backward in time, both using the optimal p^{th}-order filter $\mathbf{a}^{(p)}$. The square norm of the residual of the p^{th}-order problem appears on the right side, too.

Lastly, the square norm of the residual is

$$\begin{aligned} E_{p+1} &= E_p - \gamma_{p+1} [\, R_{p+1} \quad R_p \quad \cdots \quad R_1 \quad R_0 \,] \begin{bmatrix} \tilde{\mathbf{a}}^{(p)} \\ 0 \end{bmatrix} \\ &= E_p - \gamma_{p+1}^2 \left(R_0 + R_1 a_1^{(p)} + \cdots + R_p a_p^{(p)} \right) \\ &= E_p \left(1 - \gamma_{p+1}^2 \right). \end{aligned} \tag{E_{p+1}}$$

////

Remark: Levinson's "forward in time plus backward in time" solution "feels like" d'Alembert's "forward traveling wave plus backward traveling wave" solution to the wave equation. This is the reason the γ_ps are called reflection coefficients.

Remark: A thoughtful implementation of Levinson's algorithm monitors the reflection coefficients and stops when the coefficients cease to be "large". This is the sense in which the order p is determined by the data.