Statistical Signal Processing

Lecture 10

chapter 3: Optimal Filtering

Wiener filtering

- FIR Wiener filtering
  - iterative solution: steepest-descent algorithm

chapter 4: Adaptive Filtering

- LMS algorithm
- Normalized LMS (NLMS) algorithm
- tracking behavior of LMS and RLS
- optimal tracking via Kalman filtering

chapter 5: Sinusoids in Noise

RLS Algorithm

- LS: replace the statistical averages by a time averages:

$$\xi_k(H) = \sum_{i=1}^{k} \left(x_i - H^T Y_i\right)^2 + (H - H_0)^T R_0 (H - H_0) \quad ,$$

where the second term with $R_0 = R_0^T > 0$ allows for a proper initialization of the algorithm (the first term alone has a singular Hessian ($= 2\sum_{i=1}^{k} Y_i Y_i^T$) for $k < N$).

- We can rewrite

$$\begin{aligned}
\xi_k(H) &= H^T \left(\sum_{i=1}^{k} Y_i Y_i^T\right) H - 2H^T \left(\sum_{i=1}^{k} Y_i x_i\right) + \sum_{i=1}^{k} x_i^2 + (H - H_0)^T R_0 (H - H_0) \\
&= H^T \left(R_0 + \sum_{i=1}^{k} Y_i Y_i^T\right) H - 2H^T \left(R_0 H_0 + \sum_{i=1}^{k} Y_i x_i\right) + \sum_{i=1}^{k} x_i^2 + H_0^T R_0 H_0 \\
&= H^T R_k H - 2H^T P_k + \sum_{i=1}^{k} x_i^2 + H_0^T R_0 H_0
\end{aligned}$$

where

$$\begin{aligned}
R_k &= R_0 + \sum_{i=1}^{k} Y_i Y_i^T &= R_{k-1} + Y_k Y_k^T \\
P_k &= R_0 H_0 + \sum_{i=1}^{k} Y_i x_i &= P_{k-1} + Y_k x_k \quad .
\end{aligned}$$

Recursive Least-Squares Algorithm (2)

- By putting the gradient of $\xi_k(H)$ equal to zero and noting that the Hessian $2R_k > 0$, we find that the LS filter H_k that minimizes the LS criterion solves the following normal equations

  $$R_k\, H_k = P_k \quad .$$

  To solve this set of equations at each time instant k would take $\mathcal{O}(N^3)$ operations at each time instant. In what follows, we shall derive the Recursive LS algorithm, which allows us, using information obtained at time $k-1$, to obtain H_k with only $\mathcal{O}(N^2)$ operations.

- we can rewrite $P_k = P_{k-1} + Y_k x_k$ as

  $$\begin{aligned} R_k\, H_k &= R_{k-1} H_{k-1} + Y_k x_k \\ &= \left(R_k - Y_k Y_k^T\right) H_{k-1} + Y_k x_k \\ &= R_k H_{k-1} + Y_k \epsilon_k^p \end{aligned}$$

  where $\epsilon_k^p = x_k - H_{k-1}^T Y_k$ as in the LMS algorithm. This leads immediately to

  $$H_k = H_{k-1} + R_k^{-1} Y_k \epsilon_k^p$$

  where $R_k^{-1} Y_k$ is called the Kalman gain (the RLS algorithm is a special case of the so-called Kalman filter).

Recursive Least-Squares Algorithm (3)

- Clearly, the RLS algorithm requires the recursive update of R_k^{-1}. This can be obtained using the Matrix Inversion Lemma:

$$\begin{aligned} R_k^{-1} &= \left(R_{k-1} + Y_k Y_k^T\right)^{-1} \\ &= R_{k-1}^{-1} - R_{k-1}^{-1} Y_k \left(1 + Y_k^T R_{k-1}^{-1} Y_k\right)^{-1} Y_k^T R_{k-1}^{-1} \quad . \end{aligned}$$

This equation allows us to obtain R_k^{-1} from R_{k-1}^{-1} and Y_k using $\mathcal{O}(N^2)$ operations. When multiplying both sides with Y_k to the right, we obtain

$$R_k^{-1} Y_k = R_{k-1}^{-1} Y_k \left(1 + Y_k^T R_{k-1}^{-1} Y_k\right)^{-1} \quad .$$

We find for the *a posteriori* error

$$\epsilon_k = x_k - H_k^T Y_k = \left(1 - Y_k^T R_k^{-1} Y_k\right) \epsilon_k^p = \left(1 + Y_k^T R_{k-1}^{-1} Y_k\right)^{-1} \epsilon_k^p \; .$$

- All this can be formulated as the RLS algorithm:

$$\left\{ \begin{aligned} \epsilon_k^p &= x_k - H_{k-1}^T Y_k \\ \epsilon_k &= \epsilon_k^p \left(1 + Y_k^T R_{k-1}^{-1} Y_k\right)^{-1} \\ H_k &= H_{k-1} + R_{k-1}^{-1} Y_k \epsilon_k \\ R_k^{-1} &= R_{k-1}^{-1} - R_{k-1}^{-1} Y_k \left(1 + Y_k^T R_{k-1}^{-1} Y_k\right)^{-1} Y_k^T R_{k-1}^{-1} \quad . \end{aligned} \right.$$

Recursive Least-Squares Algorithm (4)

- The initial values for R_k^{-1} and H_k are R_0^{-1} and H_0. Compared to the LMS algorithm, the scalar stepsize μ gets replaced by a matrix stepsize R_k^{-1}. The RLS algorithm takes $\mathcal{O}(N^2)$ operations while the LMS algorithm takes only $2N$ operations. However, it converges much faster.
- *performance analysis* : with $x_k = H^{oT} Y_k + \tilde{x}_k$ (and $R_0 = 0$), we get $R_k\, H_k = P_k = \sum_{i=1}^{k} Y_i x_i = R_k H^o + \sum_{i=1}^{k} Y_i \tilde{x}_i$. Hence

$$\widetilde{H}_k = H^o - H_k = -R_k^{-1} \sum_{i=1}^{k} Y_i \tilde{x}_i$$

From this, we obtain

$$C_k \stackrel{\triangle}{=} E\, \widetilde{H}_k \widetilde{H}_k^T = \sigma_{\tilde{x}}^2 R_k^{-1} \quad .$$

Since R_k^{-1} behaves as $1/k$, we see that C_k converges to zero as $1/k$.

- *Exponential Weighting* In order to be able to track a possibly time-varying $H^o = H_k^o$, one introduces an exponential forgetting factor $\lambda \in (0,1)$ into the cost function to obtain

$$\xi_k\,(H) = \sum_{i=1}^{k} \lambda^{k-i} \left(x_i - H^T Y_i\right)^2 + \lambda^k\, (H - H_0)^T\, R_0\, (H - H_0) \quad .$$

This implies that the past (and in particular the initial conditions H_0, R_0) is forgotten exponentially fast with a window with time constant $1/(1{-}\lambda)$.

Recursive Least-Squares Algorithm (5)

- Wiener filtering: x_k and y_k are two joint stochastic processes and we're trying to estimate x_k from the y_k using a LMMSE estimator. For an FIR Wiener filter, there are a finite set of coefficients H^o involved in this LMMSE estimator.
  RLS approach: replaced statistical averages with temporal averages.

Parameter estimation interpretation

- Assume now our usual model for the *measurements* x_k,

$$x_k = H^{oT} Y_k + \tilde{x}_k$$

  where the $\tilde{x}_k$ are iid with zero mean and variance $\sigma^2_{\tilde{x}}$. Consider here $\{y_k\}$ as a deterministic signal, so the only randomness comes from the $\{\tilde{x}_k\}$. The H^o are the unknown parameters governing the model.

- The analysis of the RLS algorithm is much simpler than that of the LMS algorithm since for each k the RLS solution H_k coincides with the solution of a Least-Squares problem with a closed-form solution: $H_k = R_k^{-1} P_k$.

- Assume now $k \geq N$, $H_0 = 0$, $R_0 = 0$, and that R_k is nonsingular. The performance of the least-squares estimate is simple to analyze and leads to

$$C_k \stackrel{\triangle}{=} E\, \widetilde{H}_k \widetilde{H}_k^T = \sigma^2_{\tilde{x}} R_k^{-1} \quad .$$

- If $\tilde{x}_k$ Gaussian, $\Rightarrow$ H_k = ML estimate of H^o (efficient, C_k = CRB).

Recursive Least-Squares Algorithm (6)

A Bayesian Context - A Priori Information

- Instead of treating the filter coefficients H^o as unknown constant parameters, we could also consider H^o as a stochastic parameter vector about which we have some prior information, possibly from previous adaptive filtering experience. Assume now that, prior to obtaining the measurements $x_1, x_2, \cdots$, we know that H^o has a distribution with mean $E\,H^o = H_0$ and covariance $E\,(H^o - H_0)(H^o - H_0)^T = C_0$. So now the randomness in the x_k comes from both the $\tilde{x}_k$ and H^o.

- The problem formulation can now be recognized to be one of a *Bayesian Linear Model*. The AMMSE estimator can be shown to be the filter estimate resulting from the original RLS criterion with $R_0 = \sigma^2_{\tilde{x}} C_0^{-1}$. $C_k = E\,\widetilde{H}_k \widetilde{H}_k^T$ now satisfies

$$C_k^{-1} = \sigma_{\tilde{x}}^{-2} R_k = \sigma_{\tilde{x}}^{-2} \sum_{i=1}^{k} Y_i Y_i^T + C_0^{-1} \quad .$$

  Note that C_k^{-1} is an increasing function of C_0^{-1} and hence C_k is a decreasing function of C_0^{-1} and hence of R_0.

- So we see that H_0 and R_0 in the LS cost function have the interpretation of the prior mean and the inverse of the prior covariance of H^o. We'll choose R_0 small if we don't have a lot of confidence in our prior guess H_0 (C_0 big). In practice, R_0 is often chosen as $R_0 = \eta I_N$.

Other Adaptive Filtering Algorithms

- Fast RLS algorithms: Fast Transversal Filter (FTF) algorithm ($8N$), Fast Lattice/QR Algorithms ($\mathcal{O}(N)$ complexity)
- LMS with prewhitened input
- block processing/frequency domain LMS
- subband structures
- Fast Newton Transversal Filter (FNTF): replace R^{-1} in RLS by a banded matrix (appropriate for AR processes, hence speech)
- projection algorithms (like NLMS) on an extended subspace of L input vectors (FAP: Fast Affine Projection: complexity $2N + \mathcal{O}(L^2)$ or $2N + \mathcal{O}(L)$)
- Fast Subsampled Updating (FSU) versions of LMS and FTF: introduce some delay to reduce complexity below $\mathcal{O}(N)$
- multistage Wiener filter / polynomial expansion:

$$r_0\, R^{-1} = [\frac{1}{r_0}\, R]^{-1} = [\underbrace{I}_{\text{diagonal}} + \underbrace{(\frac{1}{r_0}\, R - I)}_{\text{off-diagonal part}}]^{-1} = \sum_{i=0}^{\infty} (I - \frac{1}{r_0}\, R)^i = \sum_{i=0}^{\infty} \alpha_i\, R^i$$

- convergence speed (RLS best) versus tracking speed (FAP best?)

Initial Convergence RLS

- Consider now $H_0 \neq 0$, $R_0 \neq 0$,

  $R_k = R_0 + R_{1:k}$, $R_{1:k} = Y_{1:k}Y_{1:k}^T$, $Y_{1:k} = [Y_1 \cdots Y_k]$, $P_k = R_0 H_0 + P_{1:k}$

- $\widetilde{H}_k = H^o - H_k = H^o - R_k^{-1} P_k = (R_0 + R_{1:k})^{-1}(R_0 \widetilde{H}_0 - \sum_{i=1}^{k} Y_i \widetilde{x}_i)$

- $C_k = E\, \widetilde{H}_k \widetilde{H}_k^T$ hence

$$\begin{aligned} C_k &= (R_0 + R_{1:k})^{-1} R_0 \widetilde{H}_0 \widetilde{H}_0^T R_0 (R_0 + R_{1:k})^{-1} + \sigma_{\tilde{x}}^2 (R_0 + R_{1:k})^{-1} R_{1:k} (R_0 + R_{1:k})^{-1} \\ &= \underbrace{\sigma_{\tilde{x}}^2 (R_0 + R_{1:k})^{-1}}_{\sim \frac{1}{k}} + \underbrace{(R_0 + R_{1:k})^{-1} (R_0 \widetilde{H}_0 \widetilde{H}_0^T R_0 - \sigma_{\tilde{x}}^2 R_0)(R_0 + R_{1:k})^{-1}}_{\sim \frac{1}{k^2} \text{ due to initialization}} \end{aligned}$$

- noiseless case: $\sigma_{\tilde{x}}^2 = 0$

$$C_k = (R_0 + R_{1:k})^{-1} R_0 \widetilde{H}_0 \widetilde{H}_0^T R_0 (R_0 + R_{1:k})^{-1}$$

- initial convergence: $1 \leq k < N$, consider $R_0 = \eta\, I$

$$C_k = \eta^2 \, (\eta\, I + R_{1:k})^{-1} \widetilde{H}_0 \widetilde{H}_0^T (\eta\, I + R_{1:k})^{-1}$$

Initial Convergence RLS (2)

- *Singular Value Decomposition* (SVD): $Y_{1:k},\ N \times k$ assumed full column rank

$$Y_{1:k} = V\Sigma U^T \quad V^TV = I_k, U^{-1} = U^T, \Sigma = \mathrm{diag}\{\sigma_1, \ldots, \sigma_k\}$$

$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_k \geq 0$ "singular values" full column rank $\leftrightarrow \sigma_k > 0$

- *Moore-Penrose pseudo-inverse*: $Y_{1:k}^+ = U\Sigma^{-1}V^T = (Y_{1:k}^T Y_{1:k})^{-1} Y_{1:k}^T$
- projection on column space: $\mathrm{P}_{Y_{1:k}} = Y_{1:k}Y_{1:k}^+ = VV^T$
- $V^+ = V^T$ $\quad \mathrm{P}_{Y_{1:k}} = VV^T = VV^+ = \mathrm{P}_V$ $\quad \mathrm{P}_V^+ = \mathrm{P}_V$
- eigendecomposition: $R_{1:k} = Y_{1:k}Y_{1:k}^T = V\Sigma^2V^T$
- let $V^\perp$ be such that $[V\ V^\perp]$ is orthogonal:

$$[V\ V^\perp][V\ V^\perp]^T = I = VV^T + V^\perp V^{\perp\,T} = \mathrm{P}_V + \mathrm{P}_{V^\perp} = \mathrm{P}_V + \mathrm{P}_V^\perp$$

$\mathrm{P}_V^\perp = I - \mathrm{P}_V = \mathrm{P}_{V^\perp}$, $\ V^\perp$ spans orthogonal complement of V

- SVD alternatively: $Y_{1:k} = [V\ V^\perp]\begin{bmatrix}\Sigma \\ 0\end{bmatrix}U^T$, $\begin{bmatrix}\Sigma \\ 0\end{bmatrix}^+ = [\Sigma^+\ \ 0]$, $\sigma^+ = \begin{cases} 1/\sigma & ,\sigma > 0 \\ 0 & ,\sigma = 0\end{cases}$
- eigendecomposition projection:

$$\mathrm{P}_V = V\,I\,V^T = [V\ V^\perp]\begin{bmatrix} I & 0 \\ 0 & 0\end{bmatrix}[V\ V^\perp]^T \quad \text{eigenvalues are 1 or 0}$$

Initial Convergence RLS (3)

- let $\eta \ll \sigma_k^2$ be small, then

$$\begin{aligned}
\eta(\eta I + R_{1:k})^{-1} &= \eta(\eta V^{\perp}V^{\perp\,T} + \eta V V^T + V\Sigma^2 V^T)^{-1} \approx \eta(\eta V^{\perp}V^{\perp\,T} + V\Sigma^2 V^T)^{-1} \\
&= \eta\left([V\ V^{\perp}]\begin{bmatrix}\Sigma^2 & 0\\ 0 & \eta I\end{bmatrix}[V\ V^{\perp}]^T\right)^{-1} = \eta[V\ V^{\perp}]\begin{bmatrix}\Sigma^2 & 0\\ 0 & \eta I\end{bmatrix}^{-1}[V\ V^{\perp}]^T \\
&= \eta V\Sigma^{-2}V^T + V^{\perp}V^{\perp\,T} = \eta\, R_{1:k}^{+} + \mathrm{P}_{R_{1:k}}^{\perp}
\end{aligned}$$

where $R_{1:k}^{+} = Y_{1:k}(Y_{1:k}^T Y_{1:k})^{-2}Y_{1:k}^T$ and $\mathrm{P}_{R_{1:k}} = \mathrm{P}_{Y_{1:k}}$

- hence

$$C_k = (\eta\, R_{1:k}^{+} + \mathrm{P}_{R_{1:k}}^{\perp})\, \widetilde{H}_0 \widetilde{H}_0^T\, (\eta\, R_{1:k}^{+} + \mathrm{P}_{R_{1:k}}^{\perp})$$

$\eta\, R_{1:k}^{+}\, \widetilde{H}_0$ reduced to $\mathcal{O}(\eta)$ in k-dim. subspace, column space of $Y_{1:k}$

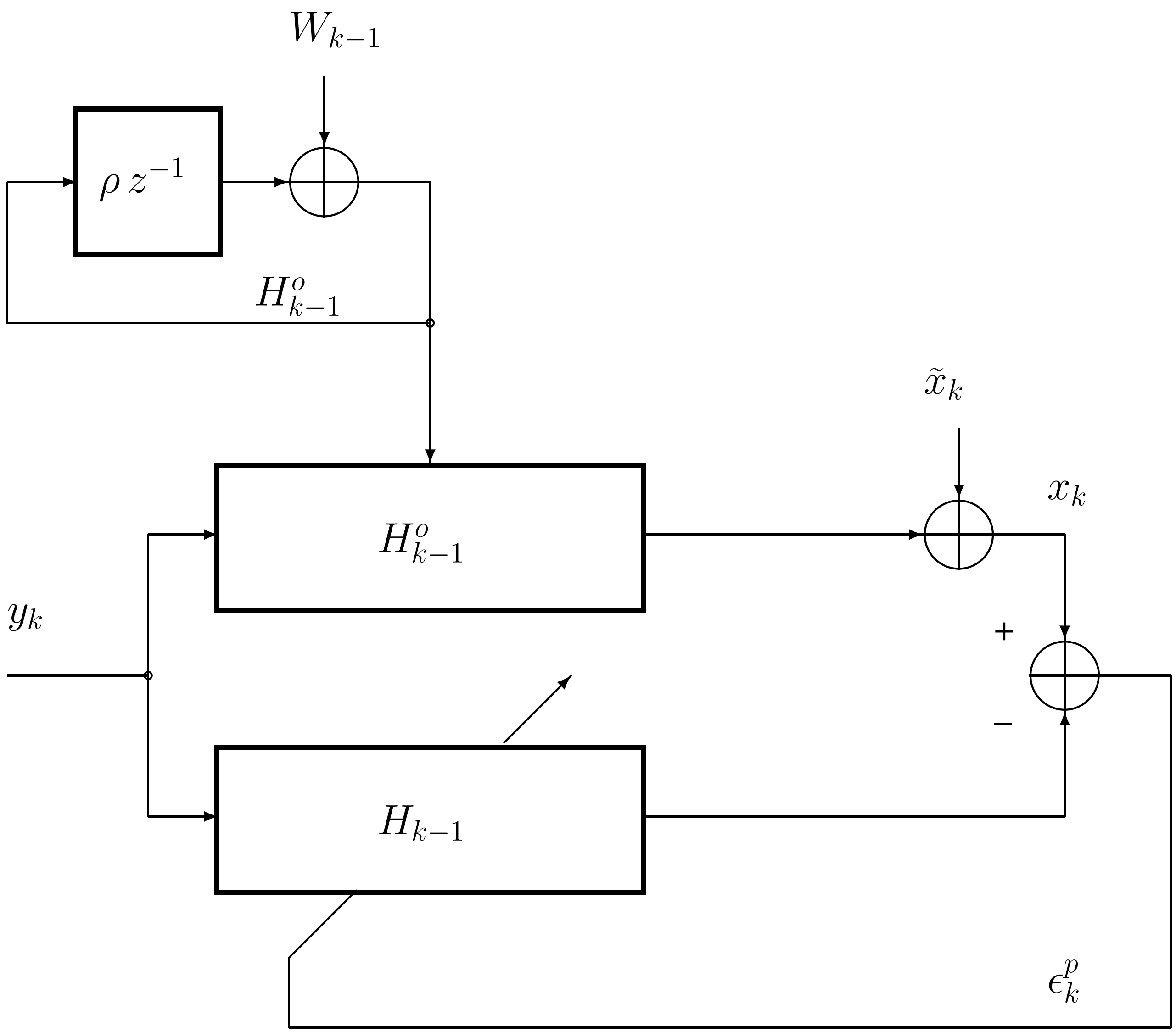
$\mathrm{P}_{R_{1:k}}^{\perp}\, \widetilde{H}_0$ unchanged in $(N-k)$-dim. orthogonal complement

- C_k rank 1 (noiseless case): only the mean of $\widetilde{H}_k$ needs to converge

- RLS: the mean of $\widetilde{H}_k$ has essentially converged (filter estimate unbiased) after $k = N \;\Rightarrow\;$ very fast (mean dominates initial convergence in general)

  LMS: the mean needs to converge exponentially, dynamics of steepest-descent

Tracking Time-Varying Filters



Time-Varying System Identification Set-Up

- system processes:

$$
\begin{aligned}
x_k &= Y_k^T H_{k-1}^o + \widetilde{x}_k & E\widetilde{x}_k\widetilde{x}_i &= \xi^o\,\delta_{ki} \\
H_k^o &= \rho\, H_{k-1}^o + W_k & EW_kW_i^T &= Q\,\delta_{ki} \\
\widetilde{H}_k &= H_k^o - H_k & EW_k\widetilde{x}_i &= 0
\end{aligned}
$$

- time-varying filter modeled as AR(1) process, requires $|\rho| < 1$ for stationarity $\rightarrow$ stationary case of nonstationarity
- adaptive filter a priori error signal:

$$\epsilon_k^p \;=\; x_k - Y_k^T H_{k-1} \;=\; Y_k^T \widetilde{H}_{k-1} + \widetilde{x}_k$$

- learning curve: (independence assumption)

$$\xi_k = E(\epsilon_k^p)^2 = \xi^o + \xi_k^e = \xi^o(1+\mathcal{M})\ ,\quad \xi_k^e = \mathrm{tr}\{R_{YY}\,C_{k-1}\}\ ,\quad C_k = E\,\widetilde{H}_k\widetilde{H}_k^T$$

- consider $\frac{1}{1-\rho} \gg$ adaptation time constants so that we can take $\rho = 1$ for the analysis

Tracking Analysis LMS

- filter deviation recursion:

$$\begin{aligned}\widetilde{H}_k &= \widetilde{H}_{k-1} - \mu\,\epsilon^p_k\,Y_k + W_k \\ &= (I - \mu\,Y_k Y_k^T)\,\widetilde{H}_{k-1} - \mu\,\widetilde{x}_k\,Y_k + W_k \\ &\approx (I - \mu\,R_{YY})\,\widetilde{H}_{k-1} - \mu\,\widetilde{x}_k\,Y_k + W_k\end{aligned}$$

  where we introduced the averaging approach in the last step

- filter error correlation matrix recursion:

$$C_k = (I - \mu\,R_{YY})\,C_{k-1}\,(I - \mu\,R_{YY}) + \mu^2\,\xi^o\,R_{YY} + Q$$

- the stationary nonstationarity combined with a constant stepsize leads to a steady-state, for which we get (with small μ):

$$R_{YY}\,C_\infty + C_\infty\,R_{YY} = \mu\,\xi^o\,R_{YY} + \frac{1}{\mu}\,Q$$

- steady-state misadjustment: $\mathcal{M}_{LMS} = \underbrace{\frac{\mu}{2}\,\mathrm{tr}R_{YY}}_{\text{estimation noise}} + \underbrace{\frac{1}{2\mu\xi^o}\,\mathrm{tr}Q}_{\text{lag noise}}$

Tracking Analysis RLS

- filter deviation recursion: $\lambda < 1$ to allow tracking

$$\begin{aligned}\widetilde{H}_k &= \widetilde{H}_{k-1} - R_k^{-1}\,Y_k\,\epsilon_k^p + W_k \\ &= (I - R_k^{-1}Y_kY_k^T)\,\widetilde{H}_{k-1} - R_k^{-1}\,Y_k\,\widetilde{x}_k + W_k\end{aligned}$$

- after averaging in steady-state, assuming small $1-\lambda$ $(I - R_k^{-1}Y_kY_k^T = \lambda\,R_k^{-1}R_{k-1} \approx \lambda\,I\;,\;\; R_k^{-1} \approx (1-\lambda)R_{YY}^{-1})$

$$\widetilde{H}_k = \lambda\,\widetilde{H}_{k-1} - (1-\lambda)\,R_{YY}^{-1}\,Y_k\,\widetilde{x}_k + W_k \quad \text{(dynamics indep. of } R_{YY}\text{)}$$

- filter error correlation matrix recursion:

$$C_k = \lambda^2\,C_{k-1} + (1-\lambda)^2\,\xi^o\,R_{YY}^{-1} + Q$$

- which leads to the steady-state value (assuming small $1-\lambda$)

$$C_\infty = \frac{1-\lambda}{2}\,\xi^o\,R_{YY}^{-1} + \frac{1}{2(1-\lambda)}\,Q$$

- steady-state misadj.: $\mathcal{M}_{RLS} = \underbrace{\frac{1-\lambda}{2}\,N}_{\text{estimation noise}} + \underbrace{\frac{1}{2(1-\lambda)\xi^o}\,\mathrm{tr}\{R_{YY}Q\}}_{\text{lag noise}}$

Tracking Optimization & LMS-RLS Comparison

- stepsize μ, $1-\lambda$ design result of compromise between:
  estimation noise: finite stepsize prevents convergence, consistency
  lag noise: small stepsize leads to lowpass filtering and to a filter estimate that lags behind the true filter
- LMS: $\mu^{opt} = \sqrt{\dfrac{\text{tr}Q}{\xi^o\,\text{tr}R_{YY}}}\,,\quad \mathcal{M}^{opt}_{LMS} = \sqrt{\dfrac{\text{tr}R_{YY}\ \text{tr}Q}{\xi^o}}$
- RLS: $\lambda^{opt} = 1 - \sqrt{\dfrac{\text{tr}\{R_{YY}Q\}}{N\,\xi^o}}\,,\quad \mathcal{M}^{opt}_{RLS} = \sqrt{\dfrac{N\,\text{tr}\{R_{YY}Q\}}{\xi^o}}$
- comparison:

$$\frac{\mathcal{M}^{opt}_{LMS}}{\mathcal{M}^{opt}_{RLS}} = \sqrt{\frac{\text{tr}R_{YY}\ \text{tr}Q}{N\,\text{tr}\{R_{YY}Q\}}}$$

$$Q = \begin{cases} q\,I & : \text{equal performance, at least for small } q \\ q\,R_{YY} & : \text{LMS is better} \\ q\,R_{YY}^{-1} & : \text{RLS is better} \end{cases}$$

- faster initial convergence of RLS could be exploited for *jumping* parameters, if windowsize properly adapted

Optimal Tracking via Kalman Filtering

- *state-space model*: state = AR(1) vector process

$$
\begin{array}{rrcl}
\textit{state}\text{ equation} & H^o_k & = & A_k\, H^o_{k-1} + W_k \\
\textit{measurement}\text{ equation} & x_k & = & Y_k^T\, H^o_{k-1} + \widetilde{x}_k
\end{array}
\quad , \quad E \begin{bmatrix} W_k \\ \widetilde{x}_k \end{bmatrix} \begin{bmatrix} W_i \\ \widetilde{x}_i \end{bmatrix}^T = \begin{bmatrix} Q_k & P_k^T \\ P_k & \xi^o_k \end{bmatrix} \delta_{ki}
$$

  time-varying (at the very least due to Y_k), usually $P_k = 0$
  state noise: W_k, measurement noise: $\widetilde{x}_k$, state transition matrix A_k

- Kalman filter (KF): estimates recursively in time the *state* H^o_{k-1} on the basis of the *measurements* $x_0, \ldots, x_k$ in a LMMSE sense.
  Sources of randomness: W_k, $\widetilde{x}_k$. Signal y_k treated as deterministic.

- special case: $H^o_k = H^o_{k-1} \;\Rightarrow$ Kalman filter $\rightarrow$ RLS algorithm

- RLS with exponential weighting can be interpreted as KF for the case of some non-zero Q_k

- in the time-invariant case (x_k and H^o_k jointly stationary apart from initial conditions), the KF converges to the causal Wiener filter.