



Adaptive Signal Processing in Communications

M. Rafie, Ph.D.

CTO - Wireless Communications

Caly Networks

rafie@calynet.com

Recursive Least-Squares Algorithm

◆ LMS used for LE / DFE

- Stochastic / steepest decent algorithm:

$$E\{\mathbf{e}_k \mathbf{u}_k^*\} \approx \mathbf{e}_k \mathbf{u}_k^*$$

- Other estimates - variation to LMS:

- Conjugate-gradient algorithm

$$S(k) = \mathbf{b}(k-1)S(k-1) - \nabla(k)$$

- Flecher-Powell algorithm

$$S(k) = -H(k)\nabla(k)$$

- Averaging the gradient vector

$$C[(k+1)M] = C(Mk) - \mathbf{m} \hat{\nabla}(k)$$

$$\hat{\nabla}(Mk) = -\frac{2}{M} \sum_{l=1}^M C(Mk+l) X^*(Mk+l)$$

- Lowpass filtering the gradient vector

Recursive Least-Squares Algorithm

◆ Steepest decent:

- Computational simplicity
- Slow convergence

$$\{\mathbf{I}_i\} \Rightarrow \frac{\mathbf{I}_{\max}}{\mathbf{I}_{\min}} \gg 1 \quad \mathbf{m} < \frac{1}{\text{tr}[\Gamma]}$$

- Only single parameter for controlling the convergence rate
- Convergence depends on the channel characteristics and # of taps

◆ Faster convergence

- More complex algorithm - N parameters, one for each eigen values
- Least square approach - min quadratic performance index
 - Time averaging - not statistical averaging

Why Recursive Least-Squares Alg.

- ◆ Many applications requires fast startup
- ◆ Channel fading - small coherence time, time selective
 - LMS may not be able to follow - lag error
- ◆ Multipath channel - coherence BW, freq selective
- ◆ Make # of iterations proportional to # of taps
 - Exponential construction of LMS algorithm
 - Complexity proportional to N^2 per iteration
 - Computationally prohibitive
 - Symbol rate
 - # of taps

RLS Algorithms

- ◆ Orthogonalize the signal vector presented to the adaptive filter
 - Processed signal vectors have a unity eigenvalue ratio
 - Remove the impact of eigenvalue spread on the convergence
 - Use Gram-Schmitt orthogonalization
 - If R (input gradient matrix) is known a priori
 - Orthogonal gradient LMS algorithm can be used
- ◆ Self-orthogonalizing algorithms

Recursive Least-Squares Algorithm

- ◆ If R has N different I_i : Use an alg. w/ N parameters
- ◆ Find $C(k)$, such that a weighted sum is minimized

$$\mathbf{e}_N = \sum_{k=1}^n w^{n-k} |e_N(k)|^2$$

$$e_N(k) = a(k) - C_N^H(n)U_N(k)$$

$$k=1, \dots, n \quad n=0, \dots, \infty \quad 0 < w < 1$$

$$U_N(k) = [u(k), u(k-1), \dots, u(k-N+1)]^T$$

$$U_N(n) = [u_{n+K1} \quad \dots \quad u_{n+1} \quad u_n \quad a_{n-1} \quad \dots \quad a_{n-K2}]^T$$

$$C_N(n) = [c_0(n), c_1(n), \dots, c_{N-1}(n)]^T$$

Normal equations:

$$R_N(n)C_N(n) = D_N(n)$$

$$C_N(n) = R_N^{-1}(n)D_N(n)$$

$$R_N(n) = \sum_{k=1}^n w^{n-k} U_N(k)U_N^H(k)$$

$$D_N(n) = \sum_{k=1}^n w^{n-k} U_N(k)a^*(k)$$

Recursive Least-Squares Algorithm

- Impractical to solve N linear equations for each Rx signal component

$$R_N(n) = w \left(\sum_{k=1}^{n-1} w^{n-1-k} U_N(k) U_N^H(k) \right) + U_N(n) U_N^H(n)$$

$$R_N(n) = w R_N(n-1) + U_N(n) U_N^H(n)$$

$$D_N(n) = w D_N(n-1) + U_N(n) a^*(n)$$

$$C_N(n) = R_N^{-1}(n) D_N(n)$$

- Matrix-inversion lemma:

$$R^{-1}(n) = \frac{1}{w} \left[R^{-1}(n-1) - \frac{R^{-1}(n-1) U(n) U^H(n) R^{-1}(n-1)}{w + U^H(n) R^{-1}(n-1) U(n)} \right]$$

Recursive Least-Squares Algorithm

- Define

$$P(n) = R^{-1}(n) \qquad C(n) = R^{-1}(n)D(n)$$

$$C(n) = P(n)D(n) = wP(n)D(n-1) + P(n)U(n)a^*(n)$$

- Kalman gain:

$$K(n) = \frac{w^{-1}P(n-1)U(n)}{1 + w^{-1}U^H(n)P(n-1)U(n)}$$

$$P(n) = \frac{1}{w} \left[P(n-1) - K(n)U^H(n)P(n-1) \right]$$

- Can show that:

$$K(n) = P(n)U(n)$$

Recursive Least-Squares Algorithm

- RLS algorithm summary:

$$C(n) = C(n-1) + K(n)[a(n) - C^H(n-1)U(n)]$$

$$e'(n) = a(n) - C^H(n-1)U(n)$$

$$C(n) = C(n-1) + K(n)e'(n)$$

$$P(n) = w^{-1}P(n-1) - w^{-1}K(n)U^H(n)P(n-1)$$

$$K(n) = \frac{w^{-1}P(n-1)U(n)}{1 + w^{-1}U^H(n)P(n-1)U(n)}$$

RLS Summary

- Suppose we have: $C(n-1)$, $P(n-1)$ and $U(n)$

- Compute

$$\hat{a}(n) = U^H(n)C(n-1)$$

- Compute error

$$e'(n) = a(n) - \hat{a}(n)$$

- Compute Kalman gain

$$K(n) = \frac{w^{-1}P(n-1)U(n)}{1 + w^{-1}U^H(n)P(n-1)U(n)}$$

- Update inverse

$$P(n) = w^{-1}P(n-1) - w^{-1}K(n)U^H(n)P(n-1)$$

- Update coefficients

$$C(n) = C(n-1) + K(n)e'(n)$$

Fast Kalman

- ◆ $U(n-1)$ of length N
- ◆ $U(n)$ is obtained by shifting $U(n-1)$ by one
 - Add $u(n)$ and discard $u(n-N)$
- ◆ Use least-squares linear prediction for this shifting property
- ◆ Update $K(n)$ without computation of $P(n)$ - reduced mult.
- ◆ Complexity is proportional to N
 - about 4X more complex than LMS
- ◆ Stability problem ($w < 1$)
- ◆ Matrix computations are replaced by recursion forward/backward predictor to update $K(n)$

Lattice Structures

- ◆ Kalman / fast Kalman - fast convergence
 - Orthogonalizing the adjustment made to coeff of an ordinary linear transversal equalizer
- ◆ Lattice:
 - Uses a lattice filter / structure to orthogonalize a set of received signal components
 - Better numerical stability
 - Order-recursive structure
 - An N-coeff equalizer uses N-1 stages
 - Complexity proportional to N
 - About 2X more complex than fast Kalman
 - Avoids matrix multiplication in $K(n)$

Lattice Structures

◆ Lattice algorithms

- Gradient lattice

- Reflection coefficients are updated as:

$$f_m(n) = f_{m-1}(n) - k_m(n-1)b_{m-1}(n-1)$$

$$b_m(n) = b_{m-1}(n-1) - k_m(n-1)f_{m-1}(n)$$

- TDL weights are updated according to LMS type

- Least-squares algorithm

- Lattice and TDL taps are updated via least-squares algorithm - "Least-Squares Lattice"

Tracking Performance of LMS & RLS

- ◆ All adaptive systems get degraded in non-stationary channels
- ◆ Steady-state error / tracking:
 - Narrow BW - small (\mathbf{n}, w)
 - Large BW - larger (\mathbf{n}, w)

Blind Equalizers

- ◆ Some applications - startup / retaining done w/o a training sequence
 - System is trained blind
 - If eye open \Rightarrow use decision-directed
- ◆ Examples:
 - Point-to-multi-point system:
 - If one of the nodes go down, it is desirable to retain only that node
 - Severe fading in digital microwave links
 - When reverse channel is not available

Blind Equalizers

- ◆ Constant modulus algorithm (CMA)

$$\min_c \left\{ E \left[\left(|q(n)|^2 - R^2 \right)^2 \right] \right\}$$

$$C(n+1) = C(n) - \mu q(n) \left(|q(n)|^2 - R^2 \right) U(n)$$

- ◆ Reduced Constellation algorithm

$$\min_c \left\{ E \left(\left| q(n) - \hat{b}(n) \right| \right)^2 \right\}$$

$$C(n+1) = C(n) - \mu \left(q(n) - \hat{b}(n) \right) e^{j\hat{q}(n)} U(n) a^*(n)$$