12. Recursive Least-Squares

From Chapter 5

The most important algorithms to be discussed are:

- The least-mean-squares algorithm (LMS)
- The normalized LMS algorithm (NLMS)
- The affine projection algorithm (APA)
- **The recursive least-squares algorithm (RLS)**

5.9 RLS Algorithm

The recursive-Least Squares (RLS) Algorithm

Block approximations of Ruu

There are times where we estimate the covariance from an N-length data set, with or without weighting. When R_{uu} is estimated based on a time history, the RLS Algorithm results (Sec. 5.9)

$$\hat{R}_{uu} = \frac{1}{i+1}\sum_{j=0}^{i} \lambda^{i-j} \cdot {u_i}^H \cdot u$$

or for $\lambda = 1$

$$\hat{R}_{uu} = \frac{1}{i+1}\sum_{j=0}^{i} {u_i}^H \cdot u$$

To see how this is applied, start with the regularized Newton's recursion of steepest decent.

$$w_i = w_{i-1} + \mu(i)\cdot\left[\varepsilon(i)\cdot I + R_{uu}\right]^{-1}\cdot\left(R_{du} - R_{uu}\cdot w_{i-1}\right)$$

using the instantaneous approximation for the "error term"

$$w_i = w_{i-1} + \mu(i)\cdot\left[\varepsilon(i)\cdot I + R_{uu}\right]^{-1}\cdot {u_i}^H \cdot\left(d(u) - u_i \cdot w_{i-1}\right)$$

and then select an estimate for the input covariance matrix

$$w_i = w_{i-1} + \mu(i)\cdot\left[\varepsilon(i)\cdot I + \hat{R}_{uu}\right]^{-1}\cdot {u_i}^H \cdot\left(d(u) - u_i \cdot w_{i-1}\right)$$

using the weighted estimate with λ, a scaled step size and a scaling of the epsilon

$$\mu(i) = \frac{1}{i+1} \quad \text{and} \quad \varepsilon(i) = \frac{\lambda^{i+1}}{i+1}\cdot\varepsilon$$

Substituting we have

$$w_i = w_{i-1} + \frac{1}{i+1}\cdot\left[\frac{\lambda^{i+1}}{i+1}\cdot\varepsilon\cdot I + \frac{1}{i+1}\sum_{j=0}^{i}\lambda^{i-j}\cdot {u_i}^H \cdot u_i\right]^{-1}\cdot {u_i}^H \cdot\left(d(u) - u_i\cdot w_{i-1}\right)$$

Manipulating the equation

$$w_i = w_{i-1} + \left[\lambda^{i+1} \cdot \varepsilon \cdot I + \sum_{j=0}^{i} \lambda^{i-j} \cdot u_i^{\;H} \cdot u_i \right]^{-1} \cdot u_i^{\;H} \cdot \left(d(u) - u_i \cdot w_{i-1} \right)$$

or

$$w_i = w_{i-1} + \left[\Phi_i \right]^{-1} \cdot u_i^{\;H} \cdot \left(d(u) - u_i \cdot w_{i-1} \right)$$

The inverse in brackets can be defined for i-step iteration as

$$\Phi_i = \left[\lambda^{i+1} \cdot \varepsilon \cdot I + \sum_{j=0}^{i} \lambda^{i-j} \cdot u_i^{\;H} \cdot u_i \right] = P_i^{-1}$$

and the iteration steps are

$$\Phi_i = \lambda \cdot \left[\lambda^{i} \cdot \varepsilon \cdot I + \sum_{j=0}^{i-1} \lambda^{i-1-j} \cdot u_i^{\;H} \cdot u_i \right] + u_i^{\;H} \cdot u_i$$

$$\Phi_i = \lambda \cdot \Phi_{i-1} + u_i^{\;H} \cdot u_i, \quad \text{where } \Phi_{-1} = \varepsilon \cdot I$$

This is a vector element change to an inverse matrix, P; therefore, the matrix inversion formula can be applied

$$\left(A + B \cdot C \cdot D \right)^{-1} = A^{-1} - A^{-1} \cdot B \cdot \left(C^{-1} + D \cdot A^{-1} \cdot B \right)^{-1} \cdot D \cdot A^{-1}$$

Substituting

$$P_i = \Phi_i^{\;-1} = \left(\lambda \cdot \Phi_{i-1} + u_i^{\;H} \cdot 1 \cdot u_i \right)^{-1}$$

$$P_i = \left(\lambda \cdot \Phi_{i-1} \right)^{-1} - \left(\lambda \cdot \Phi_{i-1} \right)^{-1} \cdot u_i^{\;H} \cdot \left(1 + u_i \cdot \left(\lambda \cdot \Phi_{i-1} \right)^{-1} \cdot u_i^{\;H} \right)^{-1} \cdot u_i \cdot \left(\lambda \cdot \Phi_{i-1} \right)^{-1}$$

$$P_i = \lambda^{-1} \cdot P_{i-1} - \lambda^{-1} \cdot P_{i-1} \cdot u_i^{\;H} \cdot \left(1 + u_i \cdot \lambda^{-1} \cdot P_{i-1} \cdot u_i^{\;H} \right)^{-1} \cdot u_i \cdot \lambda^{-1} \cdot P_{i-1}$$

$$P_i = \lambda^{-1} \cdot \left[P_{i-1} - \frac{\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\;H} \cdot u_i \cdot P_{i-1}}{1 + \lambda^{-1} \cdot u_i \cdot P_{i-1} \cdot u_i^{\;H}} \right], \quad \text{where } P_{-1} = \varepsilon^{-1} \cdot I$$

The above equation is iterated based on the previous P matrix and the most recent sample data.

The weight update becomes

$$w_i = w_{i-1} + P_i \cdot u_i^{\;H} \cdot \left(d(i) - u_i \cdot w_{i-1} \right)$$

To summarize the adaptive iterations:

$$e(i) = d(i) - u_i \cdot w_{i-1}$$

$$P_i = \lambda^{-1} \cdot \left[P_{i-1} - \frac{\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\;H} \cdot u_i \cdot P_{i-1}}{1 + \lambda^{-1} \cdot u_i \cdot P_{i-1} \cdot u_i^{\;H}} \right], \quad \text{where } P_{-1} = \varepsilon^{-1} \cdot I$$

$$w_i = w_{i-1} + P_i \cdot u_i^{\;H} \cdot \left(d(i) - u_i \cdot w_{i-1} \right)$$

$$J(i) = \left| e(i) \right|^2$$

In performing the update computations, there are some steps to reduce the number of operations, There is a repeated term that can be computed (and reduce multiple matrix multiplications to one)

$$P_i = \left[\lambda^{-1} \cdot P_{i-1} - \frac{\left(\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H}\right) \cdot \left(u_i \cdot P_{i-1} \cdot \lambda^{-1}\right)}{1 + u_i \cdot \left(\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H}\right)} \right]$$

Therefore, the following steps could be taken

1: $$u_i \cdot w_{i-1}$$

2: $$e(i) = d(u) - u_i \cdot w_{i-1}$$

3: $$\lambda^{-1} \cdot u_i^{\,H}$$

4: $$P_{i-1} \cdot \left(\lambda^{-1} \cdot u_i^{\,H}\right) = \lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H} \text{ where } P_{-1} = \frac{1}{\varepsilon} \cdot I$$

5: $$u_i \cdot \left(\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H}\right)$$

6: $$1 + u \cdot \left(\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H}\right)$$

7: $$\frac{1}{1 + u \cdot \left(\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H}\right)}$$

8: $$\frac{\left(\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H}\right)}{1 + u \cdot \left(\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H}\right)}$$

9: $$\frac{\left(\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H}\right)}{1 + u \cdot \left(\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H}\right)} \cdot \left(\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H}\right)^H$$

9: $$\lambda^{-1} \cdot P_{i-1}$$

10: $$P_i = \left[\lambda^{-1} \cdot P_{i-1} - \frac{\left(\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H}\right) \cdot \left(u_i \cdot P_{i-1} \cdot \lambda^{-1}\right)}{1 + u_i \cdot \left(\lambda^{-1} \cdot P_{i-1} \cdot u_i^{\,H}\right)} \right]$$

11: $$u_i^{\,H} \cdot e(i)$$

12: $$P_i \cdot u_i^{\,H} \cdot \left(d(u) - u_i \cdot w_{i-1}\right)$$

13: $$w_i = w_{i-1} + P_i \cdot u_i^{\,H} \cdot \left(d(u) - u_i \cdot w_{i-1}\right)$$

In MATLAB

```
%RLS
    hat_sRLS(i) = u*wRLS;
    eRLS(i) = dRLS(i)- hat_sRLS(i);
    Pdrls = 1+lambdainv*(u*Pmat*u');
    Pmat = lambdainv*(Pmat-(lambdainv*(Pmat*(u'*u)*Pmat))/Pdrls);
    wRLS = wRLS + Pmat*u'*eRLS(i);
```