Linear Filtering

Adaptive Filtering with RLS

Recursive Least Squares (RLS) I

- Superior convergence rate and steady state performance
- Computational efficient online scheme for solving the LS optimization task
- More computationally expensive compared to LMS and APA.

Recursive Least Squares (RLS) II

- Forgetting factor $0 < \beta \leq 1$
- Regularization is time-varying
- Exponentially weighted sum of squared errors cost function

$$J(\boldsymbol{\theta}, \beta, \lambda) = \sum_{i=0}^{n} \beta^{n-i} (y_i - \boldsymbol{\theta}^T \boldsymbol{x}_i)^2 + \lambda \beta^{n+1} \|\theta\|^2 \tag{1}$$

- For $\beta = 1$: Ridge regression

Recursive Least Squares (RLS) III

Minimizing cost function:

$$\frac{d}{d\boldsymbol{\theta}}J(\boldsymbol{\theta},\beta,\lambda) = -2\sum_{i=0}^{n}\beta^{n-i}(y_i - \boldsymbol{\theta}^T\boldsymbol{x}_i)\boldsymbol{x}_i + 2\lambda\beta^{n+1}\boldsymbol{I}\boldsymbol{\theta}$$

$$\boldsymbol{\Phi}_n = \sum_{i=0}^{n}\beta^{n-i}\boldsymbol{x}_i\boldsymbol{x}_i^T + \lambda\beta^{n+1}\boldsymbol{I}\mathbf{x}_n^T)$$

$$\boldsymbol{p}_n = \sum_{i=0}^{n}\beta^{n-i}\boldsymbol{x}_i y_i$$

$$\boldsymbol{\theta}_n = \boldsymbol{\Phi}_n^{-1}\boldsymbol{p}_n$$

Recursive Least Squares (RLS) IV

$$
\begin{aligned}
\mathbf{p}_n &= \sum_{i=0}^{n} \beta^{n-i} y_i \mathbf{x}_i \\
&= \sum_{i=0}^{n-1} \beta^{n-i} y_i \mathbf{x}_i + \beta^{n-n} y_n \mathbf{x}_n \\
&= \beta \left(\sum_{i=0}^{n-1} \beta^{n-1-i} y_i \mathbf{x}_i \right) + y_n \mathbf{x}_n \\
&= \beta \mathbf{p}_{n-1} + y_n \mathbf{x}_n
\end{aligned}
$$

Recursive Least Squares (RLS) V

$$\mathbf{\Phi}_n = \beta \mathbf{\Phi}_{n-1} + \mathbf{x}_n \mathbf{x}_n^T$$
$$\boldsymbol{p}_n = \beta \boldsymbol{p}_{n-1} + \boldsymbol{x}_n y_n$$
$$\boldsymbol{\theta}_n = \mathbf{\Phi}_n^{-1} (\beta \boldsymbol{p}_{n-1} + \boldsymbol{x}_n y_n)$$

Recursive Least Squares (RLS) VI

Using Woodbury's matrix inversion formula:

$$\mathbf{\Phi}_n^{-1} = (\beta\mathbf{\Phi}_{n-1} + \mathbf{x}_n\mathbf{x}_n^T)^{-1} = \beta^{-1}\mathbf{\Phi}_{n-1}^{-1} - \beta^{-1}\mathbf{k}_n\mathbf{x}_n^T\mathbf{\Phi}_n^{-1} \tag{2}$$

Kalman gain:

$$\mathbf{k}_n = \frac{\beta^{-1}\mathbf{\Phi}_{n-1}^{-1}\mathbf{x}_n}{1 + \beta^{-1}\mathbf{x}_n^T\mathbf{\Phi}_{n-1}^{-1}\mathbf{x}_n} \tag{3}$$

Set $\mathbf{P}_n = \mathbf{\Phi}_n^{-1}$

$$\mathbf{P}_n = \beta^{-1}\mathbf{P}_{n-1} - \beta^{-1}\mathbf{k}_n\mathbf{x}_n^T\mathbf{P}_{n-1} \tag{4}$$

$$\mathbf{k}_n = (\beta^{-1}\mathbf{P}_{n-1}\mathbf{x}_n - \beta^{-1}\mathbf{k}_n\mathbf{x}_n^T\mathbf{P}_{n-1})\mathbf{x}_n \tag{5}$$

$$= \mathbf{P}_n\mathbf{x}_n \tag{6}$$

Recursive Least Squares (RLS) VII

Deriving Update Rules:

$$
\begin{aligned}
\boldsymbol{\theta}_n &= \boldsymbol{\Phi}_n^{-1}\boldsymbol{p}_n \\
&= \boldsymbol{\Phi}_n^{-1}(\beta\boldsymbol{p}_{n-1} + \boldsymbol{x}_n y_n) \\
&= \beta\boldsymbol{\Phi}_n^{-1}\boldsymbol{p}_{n-1} + \boldsymbol{\Phi}_n^{-1}\boldsymbol{x}_n y_n \\
&= \beta\left(\beta^{-1}\boldsymbol{\Phi}_{n-1}^{-1} - \beta^{-1}\boldsymbol{k}_n\boldsymbol{x}_n^T\boldsymbol{\Phi}_{n-1}^{-1}\right)\boldsymbol{p}_{n-1} + \boldsymbol{\Phi}_n^{-1}\boldsymbol{x}_n y_n \\
&= \boldsymbol{\theta}_{n-1} - \boldsymbol{k}_n\boldsymbol{x}_n^T\boldsymbol{\theta}_{n-1} + \boldsymbol{\Phi}_n^{-1}\boldsymbol{x}_n y_n \\
&= \boldsymbol{\theta}_{n-1} + \boldsymbol{k}_n(y_n - \boldsymbol{x}_n^T\boldsymbol{\theta}_{n-1}) \\
&= \boldsymbol{\theta}_{n-1} + \boldsymbol{k}_n e_n \text{ (update equation)}
\end{aligned}
$$

Recursive Least Squares (RLS) VIII

RLS algorithm

1: Initialize $\theta_{-1} = 0 \in \mathbb{R}^l$; any other value is also possible.
2: Select β close to 1.
3: Set $P_{-1} = \lambda^{-1} I$, where $\lambda > 0$ is a user-defined variable.
4: **for** $n = 0, 1, 2, \ldots$ **do**
5: $\quad e_n = y_n - \theta_{n-1}^T x_n$ ▷ Compute the error
6: $\quad z_n = P_{n-1} x_n$ ▷ Update intermediate variable
7: $\quad k_n = \frac{z_n}{\beta + x_n^T z_n}$ ▷ Compute the gain
8: $\quad P_n = \beta^{-1} P_{n-1} - \beta^{-1} k_n z_n^T$ ▷ Update covariance
9: $\quad \theta_n = \theta_{n-1} + k_n e_n$ ▷ Update the estimate
10: **end for**

Relation between RLS and Newton's method I

Newton's iterative scheme: $\boldsymbol{\theta}^{(i)} = \boldsymbol{\theta}^{(i-1)} - \mu_i \left(\nabla^2 J(\boldsymbol{\theta}^{(i-1)})\right)^{-1} \nabla J(\boldsymbol{\theta}^{(i-1)})$

RLS can be derived using Newton's iterative scheme applied to the MSE:

$$J(\boldsymbol{\theta}) = \mathbb{E}\left[\frac{1}{2}(y - \boldsymbol{\theta}^T \boldsymbol{x})^2\right] = \frac{\sigma_y^2}{2} + \frac{1}{2}\boldsymbol{\theta}^T \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^T]\boldsymbol{\theta} - \boldsymbol{\theta}^T \mathbb{E}[\boldsymbol{x}y] \tag{7}$$

$$-\nabla J(\boldsymbol{\theta}) = \mathbb{E}[\boldsymbol{x}y] - \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^T]\boldsymbol{\theta} = \mathbb{E}\left[\boldsymbol{x}(y - \boldsymbol{x}^T\boldsymbol{\theta})\right] = \mathbb{E}[\boldsymbol{x}e] \tag{8}$$

$$\nabla^2 J(\boldsymbol{\theta}) = \mathbb{E}[\boldsymbol{x}\boldsymbol{x}^T] \tag{9}$$

$$\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + \mu_n \Sigma_x^{-1} \boldsymbol{x}_n e_n \tag{10}$$

Relation between RLS and Newton's method II

$$\Sigma_x \approx \frac{1}{n+1}\left(\lambda\beta^{n+1}I + \sum_{i=0}^{n}\beta^{n-i}\boldsymbol{x}_i\boldsymbol{x}_i^T\right)$$

Define the coefficients:

$$\begin{aligned}
\mu_n &= \frac{1}{n+1}, \\
\boldsymbol{k}_n &= \boldsymbol{P}_n\boldsymbol{x}_n, \\
\boldsymbol{P}_n &= \left(\lambda\beta^{n+1}I + \sum_{i=0}^{n}\beta^{n-i}\boldsymbol{x}_i\boldsymbol{x}_i^T\right)^{-1}.
\end{aligned}$$

Update Rule: $\boldsymbol{\theta}_n = \boldsymbol{\theta}_{n-1} + \mu_n\boldsymbol{k}_n e_n$

Exercise 5.2.2 I

Convergence rate for RLS and NLMS in stationary environment:

$$y_n = \boldsymbol{\theta}_o^T \boldsymbol{x_n} + \eta_n$$

where $\boldsymbol{\theta}_o \in \mathbb{R}^{200}$ is generated randomly.

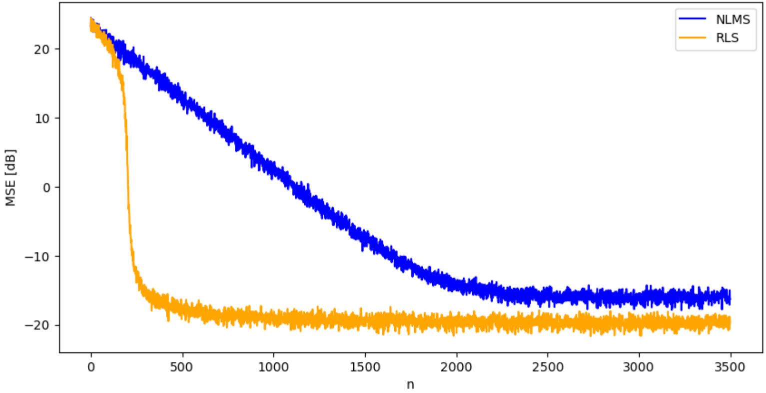
- Simulate data for 100 experiments with 3500 data samples:

```
X = np.random.randn(L, N)
X = X / np.std(X,0)
noise = np.sqrt(eta)*np.random.randn(N)
y = X.T@theta + noise
```

- Run algorithms:

```
_, E_RLS[:, i] = rls(X, y, L, beta, lambda)
_, E_NLMS[:, i] = nlms(X, y, L, mu, delta)
```

Exercise 5.2.2 II



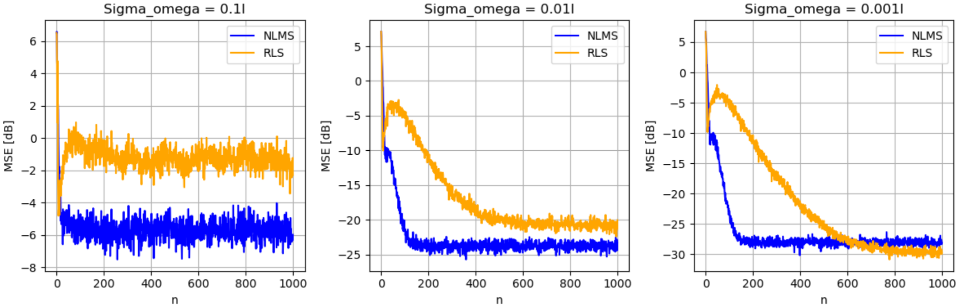
Exercise 5.3 I

- Demonstrate a case where NLMS is superior
- Time-varying parameter model with autoregressive coefficients:

$$\boldsymbol{\theta}_{0,n} = \alpha \boldsymbol{\theta}_{0,n-1} + \boldsymbol{w}_n$$

  where $\boldsymbol{w}_n$ is white noise.
- $\alpha = 0.97$
- $N_{exp} = 200$, $N = 1000$

Exercise 5.3 II



LMS/RLS expected error ratio:

$$\frac{J_{\text{min}}^{\text{LMS}}}{J_{\text{min}}^{\text{RLS}}} = \frac{\text{trace}\{\Sigma_x\}\text{trace}\{\Sigma_\omega\}}{I \cdot \text{trace}\{\Sigma_\omega \Sigma_x\}} \tag{11}$$