A Tutorial on Recursive methods in Linear Least Squares Problems

by Arvind Yedla

---

1 Introduction

This tutorial motivates the use of Recursive Methods in Linear Least Squares problems, specifically Recursive Least Squares (RLS) and its applications. Section 2 describes linear systems in general and the purpose of their study. Section 3 describes the different interpretations of Linear Equations and Least Squares Solutions. Section 4 motivates the use of recursive methods for least squares problems and Sections 5 and 6 describe an important application of Recursive Least Squares and similar algorithms.

2 Linear Systems

Linear methods are of interest in practice because they are very efficient in terms of computation. They also provide insight into the development of many non-linear algorithms. Linear models are the simplest non-trivial approximations to a complicated non-linear system. Linear models can be broadly classified into two types, which are outlined in the next two sub-sections.

2.1 Continuous Time Linear Dynamical Systems

A Continuous Time Linear Dynamical System is modeled as

$$\dot{\mathbf{x}}(t) = A(t)\mathbf{x}(t) + B(t)\mathbf{u}(t), \qquad \mathbf{y}(t) = C(t)\mathbf{x}(t) + D(t)\mathbf{u}(t),$$

where $\dot{\mathbf{x}} = \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t}$ and $\mathbf{x}$ is the state of the system, $\mathbf{u}$ is the input to the system, $\mathbf{y}$ is the output of the system, and A, B, C, and D are matrices which define the system.

2.2 Discrete Time Linear Dynamical Systems

A Discrete Time Linear Dynamical System is modeled as

$$\mathbf{x}(t+1) = A(t)\mathbf{x}(t) + B(t)\mathbf{u}(t), \qquad \mathbf{y}(t) = C(t)\mathbf{x}(t) + D(t)\mathbf{u}(t),$$

where $\mathbf{x}$, $\mathbf{u}$, $\mathbf{y}$, A, B, C and D are as defined in Section 2.1.

3 Least Squares

Consider a system of linear equations given by

$$\mathbf{y} = A\mathbf{x},$$

where $\mathbf{x} \in \mathbb{R}^n$, $A \in \mathbb{R}^{mxn}$ and $\mathbf{y} \in \mathbb{R}^{m1}$. This system of equations can be interpreted in different ways. For example, $\mathbf{y}$ is a measurement or observation and $\mathbf{x}$ is an unknown to be determined, or $\mathbf{x}$ is an input to a linear system and $\mathbf{y}$ is the output. In general, $\mathbf{y}$ is observed and we need to determine $\mathbf{x}$ which is likely to have caused that observation. The system is called over-determined if $m > n$. In this case, the existence of a solution is not guaranteed, and we seek to find an approximate solution such that the mean squared error $\|\mathbf{y} - A\mathbf{x}\|_2^2$ is minimized. This is called the least squares approximate solution, denoted by $\hat{\mathbf{x}}_{ls}$, and can be found by applying the orthogonality principle [3]. Let $\mathbf{a}_i$, for $i = 1, \ldots, n$ denote the columns of A. Then, the standard inner product $\langle A\hat{\mathbf{x}}_{ls} - \mathbf{y} \mid \mathbf{a}_i \rangle = 0$, $\forall\, i$. This can be written as $A^T A\hat{\mathbf{x}}_{ls} - A^T\mathbf{y} = 0$, which gives the least squares approximate solution[2]

$$\hat{\mathbf{x}}_{ls} = (A^T A)^{-1} A^T \mathbf{y}. \tag{1}$$

The matrix $(A^T A)^{-1} A^T$ is a left inverse of A and is denoted by $A^\dagger$. In general, it is computed using matrix factorization methods such as the QR decomposition [3], and the least squares approximate solution is given by $\hat{\mathbf{x}}_{ls} = R^{-1}Q^T\mathbf{y}$.

4 Recursive Methods

We motivate the use of recursive methods using a simple application of linear least squares (data fitting) and a specific example of that application.

4.1 Data Fitting

Given a set of basis functions $f_1, \ldots, f_n : S \to \mathbb{R}$, and a set of measurements (u_i, y_i), $i = 1, \ldots, m$ where $u_i \in S$, we need to find coefficients $x_1, \ldots, x_n \in \mathbb{R}$,

[1]We use real numbers to focus on the least squares problem. The methods and algorithms presented here can be easily extended to the complex numbers.

[2]This can be easily generalized to a weighted least squares problem, using the weighted inner product defined by $\langle \mathbf{u} \mid \mathbf{v} \rangle_W = \mathbf{v}^T W\mathbf{u}$, which gives the least squares approximate solution $\hat{\mathbf{x}}_{ls} = (A^T WA)^{-1} A^T W\mathbf{y}$.

such that the total square fitting error

$$E = \sum_j \left(y_j - \sum_i x_i f_i(u_j) \right)^2 \tag{2}$$

is minimized. This can be written in matrix notation as a least squares formulation,

$$\text{minimize } E = \|\mathbf{y} - A\mathbf{x}\|_2^2,$$

where $\mathbf{y} = \begin{bmatrix} y_1 & y_2 & \dots & y_m \end{bmatrix}^T$, $\mathbf{x} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}^T$ and

$$A = \begin{bmatrix} f_1(u_1) & f_2(u_1) & \dots & f_n(u_1) \\ f_1(u_2) & f_2(u_2) & \dots & f_n(u_2) \\ \vdots & \vdots & \ddots & \vdots \\ f_1(u_m) & f_2(u_m) & \dots & f_n(u_m) \end{bmatrix}.$$

The columns of A, denoted by $\mathbf{a}_i$, $i = 1, \dots, n$ are called regressors. There are two interesting cases when we need to update the least squares approximation, and are outlined in the next two sub-sections.

4.1.1 Growing sets of Regressors

If we need to refine our estimate of $\mathbf{y}$, by adding another function f_{n+1} to the basis, the new least squares approximate solution can be found very efficiently using a recursive algorithm. In this case, the updated approximation can be computed by adding one more step to the QR decomposition of A. Let $\mathbf{q}_i$ and $\mathbf{r}_i$ denote the i^{th} column of Q and R respectively. Then,

$$\begin{aligned} \mathbf{q}_{n+1} &= \frac{\mathbf{a}_{n+1} - \sum_{i=1}^{n} \langle \mathbf{q}_i \mid \mathbf{a}_{n+1} \rangle}{\|\mathbf{a}_{n+1} - \sum_{i=1}^{n} \langle \mathbf{q}_i \mid \mathbf{a}_{n+1} \rangle\|} \\ \mathbf{r}_{n+1} &= \begin{bmatrix} \langle \mathbf{q}_1 \mid \mathbf{a}_{n+1} \rangle & \langle \mathbf{q}_2 \mid \mathbf{a}_{n+1} \rangle & \dots \end{bmatrix}^T, \end{aligned} \tag{3}$$

where, $\langle \cdot \mid \cdot \rangle$ denotes the standard inner product. The matrices Q and R are updated using $\mathbf{q}_{n+1}$ and $\mathbf{r}_{n+1}$ respectively. The updated least squares approximate solution is then given by $R^{-1}Q^T$. This is illustrated using the example in Section 4.2.

4.1.2 Growing sets of Measurements

If we need to refine the least squares estimate due to a new observation (u_{m+1}, y_{m+1}), RLS provides an efficient way to update the least squares approximate solution. It is useful to consider the least squares solution in terms

of the rows of A. Denote the i^{th} row of A by $\tilde{\mathbf{a}}_i$. Then $\mathbf{x}_{ls} = (A^TA)^{-1}A^T\mathbf{y} = \left(\sum_i \tilde{\mathbf{a}}_i\tilde{\mathbf{a}}_i^T\right)^{-1} \cdot (\sum_i y_i\tilde{\mathbf{a}}_i)$. This would result in a very efficient method for computing the updated least-squares approximate solution, using the Sherman-Morrison formula [2]. The updated least square estimate is given by

$$\left(\sum_{i=1}^{n} \tilde{\mathbf{a}}_i\tilde{\mathbf{a}}_i^T + \tilde{\mathbf{a}}_{n+1}\tilde{\mathbf{a}}_{n+1}^T\right)^{-1} \cdot \left(\sum_{i=1}^{n} y_i\tilde{\mathbf{a}}_i + y_{n+1}\tilde{\mathbf{a}}_{n+1}\right)$$

The first term is computed efficiently using the Sherman-Morrison formula as shown in 4.

$$\left(A + \mathbf{u}\mathbf{v}^T\right)^{-1} = A^{-1} - \frac{A^{-1}\mathbf{u}\mathbf{v}^TA^{-1}}{1 + \mathbf{v}^TA^{-1}\mathbf{u}}, \tag{4}$$

for an invertible matrix A and any vectors $\mathbf{u}$ and $\mathbf{v}$ of the appropriate dimensions. This is illustrated using the example in Section 4.2.

4.2 Example

Generate a least squares fit for the data points $(0, 0)$, $(1, 1)$, $(4, 2)$, $(6, 3)$ and $(9, 4)$, using a polynomial of degree 2. Then,

- update the solution to fit the data with a polynomial of degree 3.
- update the solution if there is a new data point $(16, 5)$.

Here, $f_1(x) = 1$, $f_2(x) = x$, $f_3(x) = x^2$, $\mathbf{y} = \begin{bmatrix} 0 & 1 & 4 & 6 & 9 \end{bmatrix}^T$, and

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 3 & 9 \\ 1 & 4 & 16 \end{bmatrix}$$

The QR factorization of A is given by

$$Q = \begin{bmatrix} -0.44721 & -0.63246 & 0.53452 \\ -0.44721 & -0.31623 & -0.26726 \\ -0.44721 & 0.00000 & -0.53452 \\ -0.44721 & 0.31623 & -0.26726 \\ -0.44721 & 0.63246 & 0.53452 \end{bmatrix}, \qquad R = \begin{bmatrix} -2.2361 & -4.4721 & -13.416 \\ 0.0000 & 3.1623 & 12.649 \\ 0.0000 & 0.0000 & 3.7417 \end{bmatrix}$$

The least squares approximate solution to $\mathbf{y} = A\mathbf{x}$ is given by $\mathbf{x}_{ls} = R^{-1}Q^T\mathbf{y} = \begin{bmatrix} -0.17143 & 1.4429 & 0.21429 \end{bmatrix}^T$ and the least squares fit is given by $y =$

$-0.17143 + 1.4429\,x + 0.21429\,x^2$. For a polynomial of degree 3, the solution can be easily updated by finding the modified Q and R, given by 3. Note that only one new column has to be computed, to update Q and R. The updated solution is given by $\mathbf{x}_{ls} = \begin{bmatrix} -0.071429 & 0.72619 & 0.71429 & -0.083333 \end{bmatrix}^T$ and the polynomial fit is given by $y = -0.071429 + 0.72619\,x + 0.71429\,x^2 - 0.083333\,x^3$. The solution to the original problem could have been computed by

$$\begin{aligned}
\mathbf{x}_{ls} &= \left(\sum_i \tilde{\mathbf{a}}_i \tilde{\mathbf{a}}_i^T \right)^{-1} \cdot \left(\sum_i y_i \tilde{\mathbf{a}}_i \right) \\
&= \begin{bmatrix} 5 & 10 & 30 \\ 10 & 30 & 100 \\ 30 & 100 & 354 \end{bmatrix}^{-1} \begin{bmatrix} 20 \\ 63 \\ 215 \end{bmatrix} \\
&= \begin{bmatrix} 0.88571 & -0.77143 & 0.14286 \\ -0.77143 & 1.2429 & -0.28571 \\ 0.14286 & -0.28571 & 0.071429 \end{bmatrix} \begin{bmatrix} 20 \\ 63 \\ 215 \end{bmatrix}.
\end{aligned}$$

This solution can be easily updated to

$$\mathbf{x}_{ls} = \left(\begin{bmatrix} 5 & 10 & 30 \\ 10 & 30 & 100 \\ 30 & 100 & 354 \end{bmatrix} + \begin{bmatrix} 1 \\ 5 \\ 25 \end{bmatrix} \begin{bmatrix} 1 & 5 & 25 \end{bmatrix} \right)^{-1} \left(\begin{bmatrix} 20 \\ 63 \\ 215 \end{bmatrix} + 16 \begin{bmatrix} 1 \\ 5 \\ 25 \end{bmatrix} \right),$$

which can be computed efficiently using the Sherman-Morrison formula.

5 The Recursive Least Squares Filter

Consider the scenario of transmitting a signal $u[t]$ over a noisy fading channel. Assume that $u[t] = 0$, for $t < 1$ (the *pre-windowing* approach [3]). We can model the received signal x at time t by

$$x[t] = \sum_{k=0}^{m-1} c_i[k] u[t-k] + n[t],$$

where $c_i[k]$ are the channel parameters and m is the memory of the channel. The transmitted signal is estimated by the use of an adaptive filter $\mathbf{h}$. In other words, we are filtering a received signal $x[t]$, using a filter with impulse response $h[t]$, of length m, to produce an output that matches $u[t]$. Denote the output of the filter by $y[t]$. Then,

$$y[t] = \sum_{i=0}^{m-1} h_t[i] x[t-i],$$

and the error at each time instant t is given by

$$e[t] = y[t] - u[t].$$

The problem is to minimize the weighted least squares error function

$$E = \sum_{i=1}^{t} \lambda^{t-i} \left| e[i] \right|^2 . \tag{5}$$

The parameter λ, called the *forgetting* factor [1], is chosen according to the specific application. The squared error is the induced norm of the weighted inner product defined by

$$\langle \mathbf{u} \mid \mathbf{v} \rangle_{\Lambda_t} = \mathbf{v}^T \Lambda_t \mathbf{u},$$

where $\Lambda_t = \text{diag}\left(\lambda^0, \lambda^1, \ldots, \lambda^{t-1}\right)$. Define $\mathbf{y}[t] = \begin{bmatrix} y[t] & y[t-1] & \ldots & y[0] \end{bmatrix}^T$, $\mathbf{u}[t] = \begin{bmatrix} u[t] & u[t-1] & \ldots & u[0] \end{bmatrix}^T$ and $\mathbf{h}[t] = \begin{bmatrix} h_t[0] & h_t[1] & \ldots & h_t[m-1] \end{bmatrix}^T$. The filtered output can be written as

$$\mathbf{y}[t] = A_t \mathbf{h}[t] \approx \mathbf{u}[t],$$

where

$$A_t = \begin{bmatrix} x[1] & 0 & 0 & \ldots & 0 \\ x[2] & x[1] & 0 & \ldots & 0 \\ x[3] & x[2] & x[1] & \ldots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x[m] & x[m-1] & x[m-2] & \ldots & x[1] \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x[t] & x[t-1] & x[t-2] & \ldots & x[t-m+1] \end{bmatrix}$$

The weighted least squares solution is given by $\mathbf{h}[t] = \left(A_t^T \Lambda_t A_t\right)^{-1} A_t^T \Lambda_t \mathbf{u}[t]$. Define $\mathbf{q}[i] = \begin{bmatrix} x[i] & x[i-1] & \ldots & x[i-m+1] \end{bmatrix}^T$,

$$\begin{aligned} \mathbf{p}[t] = A_t^T \Lambda_t \mathbf{u}[t] &= \sum_{i=1}^{t} \lambda^{t-i} \mathbf{q}[i]^T u[i], \text{ and} \\ R_x[t] &= \sum_{i=1}^{t} \mathbf{q}[i] \Lambda_t \mathbf{q}[i]^T. \end{aligned} \tag{6}$$

Then, $\mathbf{h}[t] = R_x[t]\mathbf{p}[t]$. We now derive a recursive method to compute $R_x[t]$ and $\mathbf{p}[t]$ from $R_x[t-1]$ and $\mathbf{p}[t-1]$. From 6, it follows that

$$\begin{aligned} \mathbf{p}[t] &= \lambda \mathbf{p}[t-1] + \mathbf{q}[t] u[t] \\ R_x[t] &= \lambda R_x[t-1] + \mathbf{q}[t] \mathbf{q}[t]^T \end{aligned} \tag{7}$$

Define $P[t] = R_x[t]^{-1}$. From 7 and the Sherman-Morrison formula , we can compute

$$R_x[t]^{-1} = P[t] = \lambda^{-1}P[t-1] - \mathbf{k}[t]\mathbf{q}^T[t]\lambda^{-1}P[t-1],$$

where

$$\mathbf{k}[t] = \frac{P[t-1]\mathbf{q}[t]}{\lambda + \mathbf{q}^T[t]P[t-1]\mathbf{q}^T[t]} \tag{8}$$

is known as the Kalman Gain vector. Rearranging the terms of 8, we get $\mathbf{k}[t] = P[t]\mathbf{q}[t]$. From this, we obtain

$$\mathbf{h}[t] = \mathbf{h}[t-1] + \mathbf{k}[t]\left(\mathbf{u}[t] - \mathbf{q}^T[t]\mathbf{h}[t-1]\right)$$

The quantity $\epsilon[t] = \mathbf{u}[t] - \mathbf{q}^T[t]\mathbf{h}[t-1]$ represents the filter error when the output of the filter at time t is used with the filter coefficients from time $t-1$. As the filter has not been updated using the data at time t, this error is sometimes called the *a-priori* estimation error. To get the algorithm started, it is common to set $P[0] = \delta^{-1}I$, for some small positive δ and $\mathbf{h}[0] = \mathbf{0}$.

6 Other Algorithms

A similar recursive filter is the Least Mean Squares (LMS) filter [3]. It is computationally much faster when compared to the RLS filter, but the convergence of the LMS filter is much slower than that of the RLS filter. An extension to the RLS filter is the Kalman filter [3].

References

[1] http://en.wikipedia.org/wiki/Recursive_least_squares_filter.

[2] http://en.wikipedia.org/wiki/Sherman-Morrison_formula.

[3] Moon, T. K. and Stirling, W. C., *Mathematical Methods and Algorithms for Signal Processing*, Prentice Hall.