A Tutorial on Multivariate Recursive Least Squares

W. Cannon Lewis II[*]

April 5, 2020

Abstract

Least squares estimation—also known as linear regression—is one of the fundamental tools underlying modern data science and machine learning. Its typical exposition assumes a fixed dataset which is analyzed as a whole, but this assumption is violated when data arrives in a stream over time. The least squares estimate can instead be computed online using an algorithm known as recursive least squares. In this note we will derive the update equations for recursive least squares applied to both centered and uncentered data. Additionally, we will draw connections between practical implementations of recursive least squares and l^2-regularized least squares, which is also known as ridge regression.

Contents

[*] Rice University Computer Science Department (cannontwo.com)

1 Notation

Symbol	Space	Meaning
n	$\mathbb{N}$	Input feature dimension
m	$\mathbb{N}$	Output dimension
φ_t	$\mathbb{R}^{1\times n}$	Feature vector at time t
$\mu_x(t)$	$\mathbb{R}^{1\times n}$	Feature mean at time t
X_t	$\mathbb{R}^{t\times n}$	Design matrix of stacked feature vectors from timesteps 0 to t
y_t	$\mathbb{R}^{1\times m}$	Output vector at time t
$\mu_y(t)$	$\mathbb{R}^{1\times n}$	Output mean at time t
Y_t	$\mathbb{R}^{t\times m}$	Output matrix of stacked output vectors from timesteps 0 to t
Θ	$\mathbb{R}^{n\times m}$	True linear model coefficients
$\hat{\Theta}$	$\mathbb{R}^{n\times m}$	Estimated model coefficients
P_t	$\mathbb{R}^{n\times n}$	Inverse sample covariance matrix
Q_t	$\mathbb{R}^{n\times n}$	Inverse mean-corrected sample covariance matrix
R_t	$\mathbb{R}^{n\times n}$	Rank-2 update to corrected sample covariance matrix
V_t	$\mathbb{R}^{2\times n}$	$\begin{bmatrix}\mu_x(t-1)^\top & \varphi_t^\top\end{bmatrix}^\top$

Table 1: Notation used in this paper

In addition to the table above, we use the notation $x^\top$ to represent the transpose of a matrix x.

2 Problem Formulation

Let us define a data stream as a function $F(t) : \mathbb{N} \to \mathbb{R}^{1\times n} \times \mathbb{R}^{1\times m}$ that, for $t \in \mathbb{N}$, can be defined as

$$F(t) := (\varphi_t, y_t) \tag{1}$$

where φ_t and y_t are related according to

$$y_t = \varphi_t \Theta + \epsilon_t, \quad \mathbb{E}\left[\epsilon_t\right] = 0 \tag{2}$$

Note that we assume that the output of the underlying linear model with true parameters Θ is corrupted at each time step by some additive, zero-mean white noise ϵ_t. Since the purpose of this note is not to explore the statistical properties of recursive least squares, we skip over further clarification of the properties of this noise; results follow from the standard statistical analysis of linear regression [1].

2.1 Static Least Squares

If we wait to observe $T \in \mathbb{N}$ time steps of input-output pairs, we can assemble the matrices X_T and Y_T by vertically stacking the φ_t and y_t observations for $t = 1, \ldots, T$. Given these matrices, the least squares estimation problem is formulated as

$$\min_{\hat{\Theta}} = \sum_{t=1}^{\top} ||y_t - \varphi_t \hat{\Theta}||_2^2 \tag{3}$$

$$= ||Y_T - X_T \hat{\Theta}||_2^2 \tag{4}$$

It is well known (see, e.g., [1]) that the solution to this problem is given by the "normal equation"

$$\hat{\Theta}_{LS}(T) := \left(X_T^\top X_T\right)^{-1} X_T^\top Y_T \tag{5}$$

3 Centered Data

From Equation 5 we can begin to derive the recursive least squares estimator. It is worth noting at the beginning of this derivation that we have implicitly assumed that our data is already centered; in other words, the relationship between φ_t and y_t has no offset term, and so our model $\hat{\Theta}_{LS}$ will always predict an output vector of all zeros for an input vector of all zeros. This is a fine assumption when all of the data has been collected ahead of time, but breaks down when we want to do recursive least squares because we cannot estimate the means of our inputs and outputs ahead of time. Since it is easier to derive the recursive least squares estimator in the centered case than in the uncentered case, we tackle this limited version first.

3.1 Breaking Up the Normal Equation

We begin by writing out the normal equation as two sums multiplied together

$$\begin{aligned}\hat{\Theta}_{LS}(T) &= (X_T^\top X_T)^{-1} X_T^\top Y_T &(6)\\ &= \left[\sum_{t=1}^{T} \varphi_t^\top \varphi_t\right]^{-1} \left[\sum_{t=1}^{T} \varphi_t^\top y_t\right] &(7)\end{aligned}$$

Let us define the inverse sample covariance matrix P_T to be the left-hand term in Equation 7, so that we then have

$$\begin{aligned}P_T^{-1} &= \sum_{t=1}^{T} \varphi_t^\top \varphi_t &(8)\\ &= P_{T-1}^{-1} + \varphi_T^\top \varphi_T &(9)\\ \implies P_{T-1}^{-1} &= P_T^{-1} - \varphi_T^\top \varphi_T &(10)\end{aligned}$$

Similarly, we can break up the right-hand term in Equation 7:

$$\sum_{t=1}^{T} \varphi_t^\top y_t = \sum_{t=1}^{T-1} \varphi_t^\top y_t + \varphi_T^\top y_T \tag{11}$$

3.2 Deriving the Θ Update

Our normal equation is now

$$\hat{\Theta}_{LS} = P_T \cdot \left[\sum_{t=1}^{T-1} \varphi_t^\top y_t + \varphi_T^\top y_T\right] \tag{12}$$

Using the definition of $\hat{\Theta}_{LS}(T-1)$, we get

$$\hat{\Theta}_{LS}(T) = P_T \cdot \left[P_{T-1}^{-1} \hat{\Theta}_{LS}(T-1) + \varphi_T^\top y_T\right] \tag{13}$$

Substituting in Equation 10:

$$\begin{aligned}
\hat{\Theta}_{LS}(T) &= P_T \cdot \left[(P_T^{-1} - \varphi_T^\top \varphi_T)\hat{\Theta}_{LS}(T-1) + \varphi_T^\top y_T \right] && (14)\\
&= \hat{\Theta}_{LS}(T-1) - P_T \varphi_T^\top \varphi_T \hat{\Theta}_{LS}(T-1) + P_T \varphi_T^\top y_T && (15)\\
&= \hat{\Theta}_{LS}(T-1) + P_T \varphi_T^\top \left[y_T - \varphi_T \hat{\Theta}_{LS}(T-1) \right] && (16)
\end{aligned}$$

Note that the last term in Equation 16 ($y_T - \varphi_T \hat{\Theta}_{LS}(T-1)$) is the prediction error of our model at timestep $T-1$ on the new datum, so the new estimate of Θ that we get is the old estimate plus the prediction error on a new datum filtered by $P_T \varphi_T^\top$. Intuitively, this represents a reweighting of the prediction error using our existing estimate of the sample covariance, which effectively rescales the update to Θ to take into account the scales of the coordinates of the features.

3.3 Deriving the P_T Update

Though Equation 9 gives us a way to update P_T^{-1} easily with each new datum, to recover P_T and update $\hat{\theta}_{LS}$ we would need to invert an $n \times n$ matrix at on each time step. This is not only computationally expensive for all but small values of n, it also introduces the risk of running into floating-point errors if P_T^{-1} ever becomes ill-conditioned[1].

We can get around these issues by doing away with P_T^{-1} all together and deriving a direct update for P_T. We do this with the Woodbury matrix identity [3], also known as the "matrix inversion lemma." This result tells us that for matrices A, U, C, V such that UCV has rank k, the inverse of the rank-k update is given by:

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U\left(C^{-1} + VA^{-1}U\right)^{-1} VA^{-1} \tag{17}$$

In our case, since we are doing a rank-1 update $\varphi_T^\top \varphi_T$ to P_{T-1}^{-1}, this lemma gives us that

$$\left(P_{T-1}^{-1} + \varphi_T^\top \varphi_T\right)^{-1} = P_{T-1} - \frac{P_{T-1}\varphi_T^\top \varphi_T P_{T-1}}{1 + \varphi_T P_{T-1} \varphi_T^\top} \tag{18}$$

And just like that, we're done! Equations 16 and 18 give us the update equations that define the recursive least squares algorithm. At each timestep t, we simply need to:

1. Record φ_t and y_t from our datastream $F(t)$.
2. Calculate P_t from P_{t-1} and φ_t using Equation 18.
3. Calculate $\hat{\Theta}_{LS}(t)$ from $\hat{\Theta}_{LS}(t-1)$, φ_t, y_t, and P_t using Equation 16.

4 Uncentered Data

5 Practical Issues

[1] These sorts of issues can also be dealt with using any number of techniques from numerical linear algebra [2]

References

[1] Christopher M Bishop. *Pattern recognition and machine learning.* springer, 2006.

[2] Lloyd N Trefethen and David Bau III. *Numerical linear algebra*, volume 50. Siam, 1997.

[3] Max A Woodbury. Inverting modified matrices. 1950.