Recursive least squares for online estimation

Søren Højsgaard

September 28, 2010

Contents

1 Introduction

This note describes some methods for online estimation and forecasting. We consider observations $y_1, y_2, \dots$ recorded at times $t_1, t_2, \dots$. Consider a regression problem

$$y_i = g(x_i, \theta) + v_i, \quad i = 1, \dots, n \tag{1}$$

where v_i is a random component with $E(v_i) = 0$ and $V(v_i) = \sigma^2$.

Given a batch of observations $y_1, \dots, y_n$ let $\hat{\theta}_n$ denote an estimate of θ based on these data. When a new observation y_{n+1} arrives we may find $\hat{\theta}_{n+1}$ by refitting the model to $y_1, \dots, y_n, y_{n+1}$ or we may find $\hat{\theta}_{n+1}$ by updating $\hat{\theta}_n$ (which is often computationally cheaper). This is what we mean by *online estimation*.

2 Batch estimation - iterative least squares

For later use, assume that there is a non–negative weight w_i associated with measurement y_i for $i = 1, \dots, n$. A batch estimate of θ can be obtained using for example the least squares method.

The least squares estimate for θ is the value of $\hat{\theta}$ which minimizes the objective function

$$l(\theta) = \sum_i w_i (y_i - g(x_i, \theta))^2. \tag{2}$$

When convenient we write $g_i(\theta)$ instead of $g(x_i, \theta)$. Let $y = (y_1, \dots, y_n)'$, $g = g(\theta) = (g_1(\theta), \dots, g_n(\theta))'$. Let $b_i = D_\theta g_i(\theta)$ (K–vector) and let $B = [d_1 \dots d_n]$ ($K \times n$ matrix). Let also $\Lambda = diag(w_1, \dots w_n)$ ($n \times n$ matrix) and $V = B\Lambda B'$ ($K \times K$ matrix).

To minimize $l(\theta)$ we must solve $S(\theta) = D_\theta l(\theta) = 0$ (there are K equations). We find

$$S(\theta)_k = D_{\theta_k} l(\theta) = 2 \sum_i w_i (y_i - g_i(\theta)) D_{\theta_k} g_i(\theta)$$

Hence

$$S(\theta) = 2 \sum_i w_i (y_i - g_i(\theta)) D_\theta g_i(\theta) = 2B\Lambda(y - g)$$

Further let

$$\begin{aligned} j_{rk} &= D_{\theta_r} S(\theta)_k = 2 \sum_i w_i \{(y_i - g_i(\theta)) D_{\theta_r \theta_k} g_i(\theta) - D_{\theta_r} g_i(\theta) D_{\theta_k} g_i(\theta)\} \\ u_{rk} &= E(j_{rk}) = -2 \sum_i w_i D_{\theta_r} g_i(\theta) D_{\theta_k} g_i(\theta)\} = -2(V)_{rk} \end{aligned}$$

Then the Newton step becomes as follows: Let θ^* denote the current estimate of θ and iterate the following scheme until convergence

1. Set $\theta = \theta^* + (B_{\theta^*} \Lambda B'_{\theta^*})^{-1} B_{\theta^*} \Lambda (y - g(\theta^*))$
2. Set $\theta^* = \theta$.

The linear case In the special case where $g(x_i, \theta) = x_i'\theta$ we have $b_i = x_i$ and $B = X'$ where $X' = [x_1, \dots, x_n]$. In this case the Newton step becoms

$$\theta = \theta^* + (X'\Lambda X)^{-1} X'\Lambda(y - X\theta) = (X'\Lambda X)^{-1} X'\Lambda y$$

so in this case Newton converges in one iteration. We write this out in more detail as:

$$\hat{\theta} = (\sum_i w_i x_i x_i')^{-1} (\sum_i w_i x_i y_i)$$

3 Online estimation

In the following we shall assume that $w_i = \lambda^{t_n - t_i}$ where $0 < \lambda \le 1$ is called a *forgetting factor*. Hence the weights are $(\lambda^{t_n - t_1}, \lambda^{t_n - t_2}, \dots \lambda^{t_n - t_{n-1}}, 1)$. Common choices for λ are values slightly smaller than 1, say $0.99, 0.95$ or so.

3.1 Recursive least squares – the general case

Suppose now that we have $n+1$ observations $y_1, \dots, y_{n+1}$. The objective function to minimize is

$$\tilde{l}(\theta) = \sum_{i=1}^{n+1} \lambda^{t_{n+1} - t_i} (y_i - g_i(\theta))^2$$

We extend the definitions of the previously defined quantities as follows. Let $\tilde{B} = [b_1, \dots b_n, b_{n+1}] = [B, b_{n+1}]$, $\tilde{y} = (y', y_{n+1})'$, $\tilde{g} = (g', g_{n+1})'$. Let $\gamma = \lambda^{t_{n+1}-t_n}$.

$$\begin{bmatrix} \gamma\Lambda & 0 \\ 0 & 1 \end{bmatrix}$$

Then

$$\begin{aligned} \tilde{S} &= 2\tilde{B}\tilde{\Lambda}(\tilde{y} - \tilde{g}) \\ &= 2[B, b_{n+1}] \begin{bmatrix} \gamma\Lambda(y-g) \\ y_{n+1} - g_{n+1} \end{bmatrix} \\ &= 2[\gamma B\Lambda(y-g) + b_{n+1}(y_{n+1} - g_{n+1}) \\ &= 2\gamma S + 2b_{n+1}(y_{n+1} - g_{n+1}) \end{aligned}$$

Likewise we get

$$\begin{aligned} \tilde{V} &= \tilde{B}\tilde{\Lambda}\tilde{B}' \\ &= [\gamma B\Lambda : b_{n+1}]\tilde{B}' \\ &= [\gamma B\Lambda : b_{n+1}] \begin{bmatrix} B' \\ b'_{b+1} \end{bmatrix} \\ &= \gamma B\Lambda B' + b_{n+1}b'_{n+1} \\ &= \gamma V + b_{n+1}b'_{n+1} \end{aligned}$$

So now the Newton step is: Let θ^* be the current estimate and set

1. Set $\theta = \theta^* + \tilde{V}^{-1}S$
2. Set $\theta^* = \theta$.

When y_{t+1} is observed we have an estimate $\hat{\theta}_n$ based on data $y_1, \dots, y_n$ and a corresponding matrix V. Hence $\hat{\theta}_n$ is an obvious initial value for the Newton iteration and - since V is known - $\tilde{V}$ can be inverted easily using the Woodbury identity.

In subsequent iterations $\tilde{V}^{-1}$ needs to be evaluated, but if $\tilde{V}$ does not change much between subsequent iterations we can use the following approximation: If A is invertible and R is a small relative to A then $(A+R)^{-1} \approx (A^{-1} - A^{-1}RA^{-1})$. Hence if $\tilde{V}^{-1}$ is found in the first iteration and $\check{V}$ is obtained in the next iteration then

$$\check{V}^{-1} = (V + (\check{V} - V))^{-1} \approx V^{-1} - V^{-1}(\check{V} - V)V^{-1} = 2V^{-1} - V^{-1}\check{V}V^{-1}$$

3.2 Recursive least squares – the linear case

In the linear case the objective function to minimize is $l(\theta) = \sum_i \lambda^{t_n - t_i}(y_i - x_i'\theta)^2$.

Let

$$Z_n = \sum_{i=1}^{n} \lambda^{t_n - t_i} x_i x_i', \quad P_n = Z_n^{-1}, \quad \psi_n = \sum_{i=1}^{n} \lambda^{t_n - t_i} x_i y_i$$

Then the least squares estimate minimizing (2) based on n observations is

$$\hat{\theta}_n = (\sum_i \tilde{x}_i \tilde{x}_i')^{-1}(\sum_i \tilde{x}_i \tilde{y}_i) = (\sum_i \lambda^{t_n - t_i} x_i x_i')^{-1}(\sum_i \lambda^{t_n - t_i} x_i y_i) = Z_n^{-1}\psi_n = P_n\psi_n.$$

The Recursive Least Squares (or RLS) algorithm goes as follows: Given $\hat{\theta}_{n-1}$ and P_{n-1} define

$$\begin{aligned} \gamma &= \lambda^{t_n - t_{n-1}} \\ k &= \frac{1}{\gamma + x_n' P_{n-1} x_n} P_{n-1} x_n \end{aligned} \tag{3}$$

Then

$$\hat{\theta}_n = \hat{\theta}_{n-1} + k(y_n - x_n' \hat{\theta}_{n-1}) \tag{4}$$
$$P_n = (I - k x_n') \gamma^{-1} P_{n-1} \tag{5}$$

A proof of (4) and (5) is given in Section A.

3.3 Initialization

One way of starting the RLS algorithm is to first process a batch of the B first observations to obtain $\hat{\theta}_B$ and P_B and then continue the RLS algorithm (4) and (5) from there. Here B must be chosen large enough to ensure that P_B is invertible.

An approximate initialization is based on the following: Classical least squares theory gives $\mathbb{V}\mathrm{ar}(\hat{\theta}_n) = \sigma^2 P_n$ meaning that the covariance of $\hat{\theta}_n$ is proportional to P_n. At $n = 0$, the prior knowledge of θ is often sparse and this suggests to take P_0 to be "large", for example $P_0 = \delta I$ for some large value δ. For a large dataset, the initial value of P_0 is not important because it will be downweighted because of the exponential forgetting factor λ. If a prior choice of θ is available, then this value can of course be used in the initialization. Otherwise a typical initialization would be to set $\theta_0 = 0$.

3.4 Comments on the RLS

Notice that the introduction of the forgetting factor corresponds to making the assumption that $\mathbb{V}\mathrm{ar}(y_i) = \frac{\sigma^2}{\lambda^{t_n - t_i}} = \frac{\lambda^{t_i}\sigma^2}{\lambda^{t_n}}$ such that 1) observations in the far past will have a much larger variance than those close to t_n and 2) the variance of previous observations will increase as n increases. The latter point is less appealing.

Notice also that if $\hat{\theta}_{n-1}$ is an unbiased estimate of θ then $\hat{\theta}_n$ is also unbiased. However $\hat{\theta}_n$ is not in general consistent. Consider for example the model $y_i = \theta + e_i$. In this case $Z_n = \sum_{i=1}^{n} \lambda^{t_n - t_i}$. If $t_i = i$ for all t_i then Z_n is a convergent series when $\lambda < 1$ which implies that $P_n = Z_n^{-1}$ does not go to zero as n goes to infinity.

4 Least Mean Squares Algorithm

An approximation to the RLS algorithm can be obtained by defining $\tilde{k} = \rho x_n$ for a small value of ρ. Given $\hat{\theta}_{n-1}$ set $\hat{\theta}_n^0 = \hat{\theta}_{n-1}$ and do the following iteration until (hopefully) convergence

$$\hat{\theta}_n^j = \hat{\theta}_n^{j-1} + \rho x_n' (y_n - x_n' \hat{\theta}_n^{j-1}). \tag{6}$$

The scheme (6) is called a *least mean squares* (or *LMS*) algorithm. A problem with the LMS algorithm is that the step sizes depends on the scaling of the covariates n_n. The *normalized least mean squares* (or *NLMS*) algorithm tries to remedy this problem by normalizing the covariates to have length 1:

$$\hat{\theta}_n^j = \hat{\theta}_n^{j-1} + \rho \frac{1}{x_n' x_n} x_n' (y_n - x_n' \hat{\theta}_n^{j-1}). \tag{7}$$

A Proof of RLS updates

To prove (4) first notice that we get the following recurrence relations for W_n and ψ_n:

$$\begin{aligned} W_n &= \sum_{i=1}^{n} \lambda^{t_n - t_i} x_i x_i' = x_n x_n' + \sum_{i=1}^{n-1} \lambda^{t_n - t_i} x_i x_i' \\ &= x_n x_n' + \lambda^{t_n - t_{n-1}} \sum_{i=1}^{n-1} \lambda^{t_{n-1} - t_i} x_i x_i' = x_n x_n + \lambda^{t_n - t_{n-1}} W_{n-1} \end{aligned} \tag{8}$$

and similarly

$$\psi_n = x_n y_n + \lambda^{t_n - t_{n-1}} \psi_{n-1} \tag{9}$$

We need a recurrence relation for $P_n = W_n^{-1}$ as well. Let $\gamma = \lambda^{t_n - t_{n-1}}$. Using the Woodbury identity (12) we find

$$\begin{aligned} P_n &= (\gamma W_{n-1} + x_n x_n')^{-1} \\ &= \gamma^{-1} W_{n-1}^{-1} - \frac{1}{1 + \gamma^{-1} x_n' W_{n-1}^{-1} x_n} \gamma^{-1} W_{n-1}^{-1} x_n x_n' \gamma^{-1} W_{n-1}^{-1} \\ &= \gamma^{-1} P_{n-1} - \frac{1}{1 + \gamma^{-1} x_n' P_{n-1} x_n} \gamma^{-1} P_{n-1} x_n x_n' \gamma^{-1} P_{n-1} \end{aligned} \tag{10}$$

To avoid making the notation too cumbersome let $Q = P_{n-1}$ and $\phi = \psi_{n-1}$. Then

$$P_n = \gamma^{-1} Q - \frac{1}{1 + \gamma^{-1} x_n' Q x_n} \gamma^{-1} Q x_n x_n' \gamma^{-1} Q.$$

The least squares estimate of θ based on the first $n-1$ observations is consequently

$$\hat{\theta}_{n-1} = P_{n-1} \psi_{n-1} = Q\phi.$$

For later use define $c = -\frac{1}{1+\gamma^{-1} x_n' Q x_n}$ and notice that $\gamma^{-1} x_n' Q x_n = -(1 + \frac{1}{c})$. Now it is all straight forward:

$$\begin{aligned} \hat{\theta}_n &= P_n \psi_n \\ &= (\gamma^{-1} Q + c\gamma^{-1} Q x_n x_n' \gamma^{-1} Q)(\gamma\phi + x_n y_n) \\ &= \gamma^{-1} Q \gamma \phi + c\gamma^{-1} Q x_n x_n' \gamma^{-1} Q \gamma \phi + \gamma^{-1} Q x_n y_n + c\gamma^{-1} Q x_n x_n' \gamma^{-1} Q x_n y_n \\ &= \hat{\theta}_{n-1} + c\gamma^{-1} Q x_n x_n' \hat{\theta}_{n-1} + \gamma^{-1} Q x_n y_n + c\gamma^{-1} Q x_n x_n' \gamma^{-1} Q x_n y_n \\ &= \hat{\theta}_{n-1} + c\gamma^{-1} Q x_n x_n' \hat{\theta}_{n-1} + \gamma^{-1} Q x_n y_n - c\gamma^{-1} Q x_n (1 + \frac{1}{c}) y_n \\ &= \hat{\theta}_{n-1} + c\gamma^{-1} Q x_n x_n' \hat{\theta}_{n-1} - c\gamma^{-1} Q x_n y_n \\ &= \hat{\theta}_{n-1} - c\gamma^{-1} Q x_n (y_n - x_n' \hat{\theta}_{n-1}) \end{aligned}$$

With k defined as in (3) we get

$$\hat{\theta}_n = \hat{\theta}_{n-1} + k(y_n - x_n' \hat{\theta}_{n-1}) \tag{11}$$

which proves (4). To prove (5) notice that

$$\begin{aligned} P_n &= \gamma^{-1} Q - \frac{1}{1 + \gamma^{-1} x_n' Q x_n} \gamma^{-1} Q x_n x_n' \gamma^{-1} Q = \gamma^{-1} Q - k x_n' \gamma^{-1} Q \\ &= (I - k x_n') \gamma^{-1} Q = (I - k x_n') \gamma^{-1} P_{n-1}. \end{aligned}$$

B Woodbury identity

Woodbury identity Let A be and invertible matrix and a a vector. Then

$$(A + aa')^{-1} = A^{-1} - \frac{1}{1 + a'A^{-1}a} A^{-1}aa'A^{-1}. \tag{12}$$