

## Online least-squares

In our discussion of least-squares so far, we have focussed on static problems: a set of samples  $\mathbf{y} = \mathbf{A}\mathbf{x}_0 + \mathbf{e}$  comes in all at once, and we use them all to estimate  $\mathbf{x}_0$ .

In this section, we will shift our focus to **streaming problems**. We observe<sup>1</sup>

$$\begin{aligned}\mathbf{y}_0 &= \mathbf{A}_0\mathbf{x}_* + \mathbf{e}_0 \\ \mathbf{y}_1 &= \mathbf{A}_1\mathbf{x}_* + \mathbf{e}_1 \\ &\vdots \\ \mathbf{y}_k &= \mathbf{A}_k\mathbf{x}_* + \mathbf{e}_k \\ &\vdots\end{aligned}$$

At each time  $k$ , we want to form the best estimate of  $\mathbf{x}_*$  from the observations  $\mathbf{y}_0, \mathbf{y}_1, \dots, \mathbf{y}_k$  seen up to that point.

Moreover, we would like to do this in an efficient manner. The size of the problem is growing with  $k$  — rather than resolving the problem from scratch every time, we would like a principled (and fast) way to **update** the solution when a new observation is made. The idea is that our estimate is adjusted “online” as the data comes streaming by.

In our framework, the sample matrices  $\mathbf{A}_1, \mathbf{A}_2, \dots$  can be different, and can even have a different number of rows. We will work under the assumption that the total number of samples we have seen at

---

<sup>1</sup>To avoid confusion with the streaming index  $k$ , we are using the notation  $\mathbf{x}_*$  for the unknown vector; here,  $\mathbf{x}_*$  is fixed, but the  $\mathbf{y}_k, \mathbf{A}_k$ , and  $\mathbf{e}_k$  can all vary from instance to instance.

any point exceeds the number of unknowns, and if we form

$$\underline{\mathbf{A}}_k = \begin{bmatrix} \mathbf{A}_0 \\ \mathbf{A}_1 \\ \vdots \\ \mathbf{A}_k \end{bmatrix}$$

then  $\underline{\mathbf{A}}_k^T \underline{\mathbf{A}}_k$  is invertible. This assumption is not really necessary, it just makes the discussion easier; generalizing what we say to rank-deficient systems is not that hard.

In fact, we will explore on the homework how to adapt this framework to the kernel regression framework (where we never have more data than unknowns, since we are working in an infinite dimensional space.)

The key piece of mathematical technology we need is the **matrix inversion lemma**, which is also known as the Sherman-Morrison-Woodbury equation.

## The Matrix Inversion Lemma

The matrix inversion lemma shows us how the solution to a system of equations can be efficiently updated.

Let  $\mathbf{W}$ ,  $\mathbf{X}$ ,  $\mathbf{Y}$ , and  $\mathbf{Z}$  be matrices as follows:

- $\mathbf{W}$  is  $N \times N$  and invertible,
- $\mathbf{X}$  and  $\mathbf{Z}$  are  $R \times N$ , and
- $\mathbf{Y}$  is  $R \times R$  and invertible.

Then the following identity holds:

$$(\mathbf{W} + \mathbf{X}^T \mathbf{Y} \mathbf{Z})^{-1} = \mathbf{W}^{-1} - \mathbf{W}^{-1} \mathbf{X}^T (\mathbf{Y}^{-1} + \mathbf{Z} \mathbf{W}^{-1} \mathbf{X}^T)^{-1} \mathbf{Z} \mathbf{W}^{-1} \quad (1)$$

This is also known as the *Sherman-Morrison-Woodbury* identity. The point is that if  $\mathbf{W}^{-1}$  has already been calculated, then finding a solution to  $(\mathbf{W} + \mathbf{X}^T \mathbf{Y} \mathbf{Z})\mathbf{w} = \mathbf{v}$  costs  $O(N^2 R) + O(N R^2) + O(R^3)$  instead of  $O(N^3)$ . If  $R$  is very small compared to  $N$ , this can be a significant savings.

The proof of (1) is straightforward. Given any right hand side  $\mathbf{v} \in \mathbb{R}^N$ , we would like to solve

$$(\mathbf{W} + \mathbf{X}^T \mathbf{Y} \mathbf{Z})\mathbf{w} = \mathbf{v} \quad (2)$$

for  $\mathbf{w}$ . Set

$$\mathbf{z} = \mathbf{Y} \mathbf{Z} \mathbf{w} \quad \Rightarrow \quad \mathbf{Y}^{-1} \mathbf{z} = \mathbf{Z} \mathbf{w}.$$

We now have the set of two equations

$$\begin{aligned} \mathbf{W} \mathbf{w} + \mathbf{X}^T \mathbf{z} &= \mathbf{v} \\ \mathbf{Z} \mathbf{w} - \mathbf{Y}^{-1} \mathbf{z} &= \mathbf{0}. \end{aligned}$$

Manipulating the first equation yields

$$\mathbf{w} = \mathbf{W}^{-1}(\mathbf{v} - \mathbf{X}^T \mathbf{z}), \quad (3)$$

and then plugging this into the second equation gives us

$$\begin{aligned} \mathbf{Z} \mathbf{W}^{-1} \mathbf{v} - \mathbf{Z} \mathbf{W}^{-1} \mathbf{X}^T \mathbf{z} - \mathbf{Y}^{-1} \mathbf{z} &= \mathbf{0} \\ \Rightarrow \mathbf{z} &= (\mathbf{Y}^{-1} + \mathbf{Z} \mathbf{W}^{-1} \mathbf{X}^T)^{-1} \mathbf{Z} \mathbf{W}^{-1} \mathbf{v}. \end{aligned} \quad (4)$$

So then given any  $\mathbf{v} \in \mathbb{R}^N$ , we can solve for  $\mathbf{w}$  in (2) by combining (3) and (4) to get

$$\mathbf{w} = \mathbf{W}^{-1}\mathbf{v} - \mathbf{W}^{-1}\mathbf{X}^T(\mathbf{Y}^{-1} + \mathbf{Z}\mathbf{W}^{-1}\mathbf{X}^T)^{-1}\mathbf{Z}\mathbf{W}^{-1}\mathbf{v}.$$

As this holds for any right-hand side  $\mathbf{v}$ , this establishes (1).

## Updating least-squares solutions

We can apply the matrix inversion lemma to efficiently update the solution to least-squares problems as new samples become available.

Suppose we have observed

$$\mathbf{y}_0 = \mathbf{A}_0\mathbf{x}_* + \mathbf{e}_0$$

and have formed the least-squares estimate<sup>2</sup>

$$\hat{\mathbf{x}}_0 = (\mathbf{A}_0^T\mathbf{A}_0)^{-1}\mathbf{A}_0^T\mathbf{y}_0.$$

Now we observe

$$\mathbf{y}_1 = \mathbf{A}_1\mathbf{x}_* + \mathbf{e}_1,$$

where  $\mathbf{A}_1$  is an  $M_1 \times N$  matrix with  $M_1 \ll N$ . Given  $\mathbf{y}_0$  and  $\mathbf{y}_1$ , the full least-squares estimate is formed from the system of equations

$$\begin{bmatrix} \mathbf{y}_0 \\ \mathbf{y}_1 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_0 \\ \mathbf{A}_1 \end{bmatrix} \mathbf{x}_* + \begin{bmatrix} \mathbf{e}_0 \\ \mathbf{e}_1 \end{bmatrix},$$

resulting in

$$\hat{\mathbf{x}}_1 = \left( \mathbf{A}_0^T\mathbf{A}_0 + \mathbf{A}_1^T\mathbf{A}_1 \right)^{-1} (\mathbf{A}_0^T\mathbf{y}_0 + \mathbf{A}_1^T\mathbf{y}_1).$$

---

<sup>2</sup>Our estimate will change with  $k$ , so we denote the least-squares estimate of  $\mathbf{x}_*$  at time  $k$  as  $\hat{\mathbf{x}}_k$

Now let  $\mathbf{P}_k$  be the aggregated system we would like to solve at each step:

$$\begin{aligned}\mathbf{P}_0 &= (\mathbf{A}_0^T \mathbf{A}_0)^{-1} \\ \mathbf{P}_1 &= (\mathbf{A}_0^T \mathbf{A}_0 + \mathbf{A}_1^T \mathbf{A}_1)^{-1},\end{aligned}$$

and so

$$\begin{aligned}\mathbf{P}_1^{-1} &= \mathbf{A}_0^T \mathbf{A}_0 + \mathbf{A}_1^T \mathbf{A}_1 \\ &= \mathbf{P}_0^{-1} + \mathbf{A}_1^T \mathbf{A}_1.\end{aligned}$$

Then using the matrix inversion lemma with

$$\mathbf{W} = \mathbf{A}_0^T \mathbf{A}_0 = \mathbf{P}_0^{-1}, \quad \mathbf{X} = \mathbf{Z} = \mathbf{A}_1, \quad \mathbf{Y} = \mathbf{I},$$

gives us the update

$$\mathbf{P}_1 = \mathbf{P}_0 - \mathbf{P}_0 \mathbf{A}_1^T (\mathbf{I} + \mathbf{A}_1 \mathbf{P}_0 \mathbf{A}_1^T)^{-1} \mathbf{A}_1 \mathbf{P}_0.$$

When the number of new samples (rows in  $\mathbf{A}_1$ ) is small, then the system of equations  $\mathbf{I} + \mathbf{A}_1 \mathbf{P}_0 \mathbf{A}_1^T$  can be much easier to handle than  $\mathbf{A}_0^T \mathbf{A}_0 + \mathbf{A}_1^T \mathbf{A}_1 = \mathbf{P}_1^{-1}$ . For example, suppose we see just one new measurement, so the matrix  $\mathbf{A}_1$  has just one row:  $\mathbf{A}_1 = \mathbf{a}_1^T$ ,  $\mathbf{a}_1 \in \mathbb{R}^N$ . Then

$$y_1 = \mathbf{a}_1^T \mathbf{x}_* + e_1,$$

and

$$\hat{\mathbf{x}}_1 = [\mathbf{P}_0 - \mathbf{P}_0 \mathbf{a}_1 (1 + \mathbf{a}_1^T \mathbf{P}_0 \mathbf{a}_1)^{-1} \mathbf{a}_1^T \mathbf{P}_0] (\mathbf{A}_0^T \mathbf{y}_0 + y_1 \mathbf{a}_1).$$

Set  $\mathbf{u} = \mathbf{P}_0 \mathbf{a}_1$ . Then

$$\begin{aligned}\hat{\mathbf{x}}_1 &= \hat{\mathbf{x}}_0 + y_1 \mathbf{u} - \frac{\mathbf{a}_1^T \hat{\mathbf{x}}_0}{1 + \mathbf{a}_1^T \mathbf{u}} \mathbf{u} - \frac{y_1 \cdot \mathbf{a}_1^T \mathbf{u}}{1 + \mathbf{a}_1^T \mathbf{u}} \mathbf{u} \\ &= \hat{\mathbf{x}}_0 + \left( \frac{1}{1 + \mathbf{a}_1^T \mathbf{u}} \right) (y_1 - \mathbf{a}_1^T \hat{\mathbf{x}}_0) \mathbf{u}.\end{aligned}$$

Thus we can update the solution with one vector-matrix multiply (which has cost  $O(N^2)$ ) and two inner products (with cost  $O(N)$ ).

In addition, we can carry forward the “information matrix” using the update

$$\mathbf{P}_1 = \mathbf{P}_0 - \frac{1}{1 + \mathbf{a}_1^T \mathbf{u}} \mathbf{u} \mathbf{u}^T.$$

In general (for  $M_1$  new measurements), we have

$$\begin{aligned} \hat{\mathbf{x}}_1 &= \mathbf{P}_1 (\mathbf{A}_0^T \mathbf{y}_0 + \mathbf{A}_1^T \mathbf{y}_1) \\ &= \mathbf{P}_1 (\mathbf{P}_0^{-1} \hat{\mathbf{x}}_0 + \mathbf{A}_1^T \mathbf{y}_1), \end{aligned}$$

and since

$$\mathbf{P}_0^{-1} = \mathbf{P}_1^{-1} - \mathbf{A}_1^T \mathbf{A}_1,$$

this implies

$$\begin{aligned} \hat{\mathbf{x}}_1 &= \mathbf{P}_1 \left( \mathbf{P}_1^{-1} \hat{\mathbf{x}}_0 - \mathbf{A}_1^T \mathbf{A}_1 \hat{\mathbf{x}}_0 + \mathbf{A}_1^T \mathbf{y}_1 \right) \\ &= \hat{\mathbf{x}}_0 + \mathbf{K}_1 (\mathbf{y}_1 - \mathbf{A}_1 \hat{\mathbf{x}}_0), \end{aligned}$$

where  $\mathbf{K}_1$  is the “gain matrix”

$$\mathbf{K}_1 = \mathbf{P}_1 \mathbf{A}_1^T.$$

The update for  $\mathbf{P}_1$  is

$$\begin{aligned} \mathbf{P}_1 &= \mathbf{P}_0 - \mathbf{P}_0 \mathbf{A}_1^T (\mathbf{I} + \mathbf{A}_1 \mathbf{P}_0 \mathbf{A}_1^T)^{-1} \mathbf{A}_1 \mathbf{P}_0 \\ &= \mathbf{P}_0 - \mathbf{U} (\mathbf{I} + \mathbf{A}_1 \mathbf{U})^{-1} \mathbf{U}^T, \end{aligned}$$

where  $\mathbf{U} = \mathbf{P}_0 \mathbf{A}_1^T$  is an  $N \times M_1$  matrix, and  $\mathbf{I} + \mathbf{A}_1 \mathbf{U}$  is  $M_1 \times M_1$ . So the cost of the update is

- $O(M_1 N^2)$  to compute  $\mathbf{U} = \mathbf{P}_0 \mathbf{A}_1^T$ ,
- $O(M_1^2 N)$  to compute  $\mathbf{A}_1 \mathbf{U}$ ,
- $O(M_1^3)$  to invert<sup>3</sup>  $(\mathbf{I} + \mathbf{A}_1 \mathbf{U})^{-1}$ ,
- $O(M_1^2 N)$  to compute  $(\mathbf{I} + \mathbf{A}_1 \mathbf{U})^{-1} \mathbf{U}^T$ ,
- $O(M_1 N^2)$  to take the result of the last step and apply  $\mathbf{U}$ ,
- $O(N^2)$  to subtract the result of the last step from  $\mathbf{P}_0$ .

So assuming that  $M_1 < N$ , the overall cost is  $O(M_1 N^2)$ , which is on the order of  $M_1$  vector-matrix multiplies.

## Online Least Squares (or Recursive Least Squares (RLS))

Given

$$\begin{aligned}
 \mathbf{y}_0 &= \mathbf{A}_0 \mathbf{x}_* + \mathbf{e}_0 \\
 \mathbf{y}_1 &= \mathbf{A}_1 \mathbf{x}_* + \mathbf{e}_1 \\
 &\vdots \\
 \mathbf{y}_k &= \mathbf{A}_k \mathbf{x}_* + \mathbf{e}_k \\
 &\vdots,
 \end{aligned}$$

RLS is an **online algorithm** for computing the best estimate for  $\mathbf{x}_*$  from all the measurements it has seen up to the current time.

---

<sup>3</sup>In practice, it is probably more stable to find and update a factorization of this matrix. But the cost is the same.

## Online/Recursive Least Squares

Initialize: ( $\mathbf{A}_0$ ,  $\mathbf{y}_0$  appear)

$$\mathbf{P}_0 = (\mathbf{A}_0^T \mathbf{A}_0)^{-1}$$

$$\hat{\mathbf{x}}_0 = \mathbf{P}_0 (\mathbf{A}_0^T \mathbf{y}_0)$$

**for**  $k = 1, 2, 3, \dots$  **do**

    ( $\mathbf{A}_k$ ,  $\mathbf{y}_k$  appear)

$$\mathbf{P}_k = \mathbf{P}_{k-1} - \mathbf{P}_{k-1} \mathbf{A}_k^T (\mathbf{I} + \mathbf{A}_k \mathbf{P}_{k-1} \mathbf{A}_k^T)^{-1} \mathbf{A}_k \mathbf{P}_{k-1}$$

$$\mathbf{K}_k = \mathbf{P}_k \mathbf{A}_k^T$$

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_{k-1} + \mathbf{K}_k (\mathbf{y}_k - \mathbf{A}_k \hat{\mathbf{x}}_{k-1})$$

**end for**