Streaming solutions to least-squares problems

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

In this section, we will shift our focus to **streaming problems**. We observe[1]

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

At each time k, we want to form the best estimate of $\boldsymbol{x}_*$ from the observations $\boldsymbol{y}_0, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_k$ 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.

We will consider two basic frameworks:

1. Recursive Least Squares (RLS):
   The vector $\boldsymbol{x}_*$ we are estimating does not vary.
2. The Kalman filter:

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

The vector $\boldsymbol{x}_k$ moves at every times step, and we have a (linear) dynamical model for how it moves.

In both of these frameworks, the measurements matrices $\boldsymbol{A}_1, \boldsymbol{A}_2, \ldots$ can be different, and can even have a different number of rows. We will work under the assumption that the total number of measurements we have seen at any point exceeds the number of unknowns, and if we form

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

then $\underline{\boldsymbol{A}}_k^{\mathrm{T}} \underline{\boldsymbol{A}}_k$ 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.

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

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

Then the following identity holds:

$$(\boldsymbol{W}+\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{Z})^{-1} = \boldsymbol{W}^{-1}-\boldsymbol{W}^{-1}\boldsymbol{X}^{\mathrm{T}}(\boldsymbol{Y}^{-1}+\boldsymbol{Z}\boldsymbol{W}^{-1}\boldsymbol{X}^{\mathrm{T}})^{-1}\boldsymbol{Z}\boldsymbol{W}^{-1} \tag{1}$$

This is also known as the *Sherman-Morrison-Woodbury* identity. The point is that if $\boldsymbol{W}^{-1}$ has already been calculated, then finding a solution to $(\boldsymbol{W}+\boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{Z})\boldsymbol{w} = \boldsymbol{v}$ costs $O(N^2R)+O(NR^2)+O(R^3)$ 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 $\boldsymbol{v} \in \mathbb{R}^N$, we would like to solve

$$(\boldsymbol{W} + \boldsymbol{X}^{\mathrm{T}}\boldsymbol{Y}\boldsymbol{Z})\boldsymbol{w} = \boldsymbol{v} \tag{2}$$

for $\boldsymbol{w}$. Set

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

We now have the set of two equations

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

Manipulating the first equation yields

$$\boldsymbol{w} = \boldsymbol{W}^{-1}(\boldsymbol{v} - \boldsymbol{X}^{\mathrm{T}}\boldsymbol{z}), \tag{3}$$

and then plugging this into the second equation gives us

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

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

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

As this holds for any right-hand side $\boldsymbol{v}$, 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 measurements become available.

Suppose we have observed

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

and have formed the least-squares estimate[2]

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

Now we observe

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

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

$$\begin{bmatrix}\boldsymbol{y}_0\\ \boldsymbol{y}_1\end{bmatrix} = \begin{bmatrix}\boldsymbol{A}_0\\ \boldsymbol{A}_1\end{bmatrix}\boldsymbol{x}_* + \begin{bmatrix}\boldsymbol{e}_1\\ \boldsymbol{e}_2\end{bmatrix},$$

[2]Our estimate will change with k, so we denote the least-squares estimate of $\boldsymbol{x}_*$ at time k as $\hat{\boldsymbol{x}}_k$

resulting in

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

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

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

and so

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

Then using the matrix inversion lemma with

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

gives us the update

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

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

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

and

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

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

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

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

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

In general (for M_1 new measurements), we have

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

and since

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

this implies

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

where $\boldsymbol{K}_1$ is the "gain matrix"

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

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

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

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

- $O(M_1N^2)$ to compute $\boldsymbol{U} = \boldsymbol{P}_0\boldsymbol{A}_1^{\mathrm{T}}$,
- $O(M_1^2N)$ to compute $\boldsymbol{A}_1\boldsymbol{U}$,
- $O(M_1^3)$ to invert[3] $(\mathbf{I} + \boldsymbol{A}_1\boldsymbol{U})^{-1}$,
- $O(M_1^2N)$ to compute $(\mathbf{I} + \boldsymbol{A}_1\boldsymbol{U})^{-1}\boldsymbol{U}^{\mathrm{T}}$,
- $O(M_1N^2)$ to take the result of the last step and apply $\boldsymbol{U}$,
- $O(N^2)$ to subtract the result of the last step from $\boldsymbol{P}_0$.

So assuming that $M_1 < N$, the overall cost is $O(M_1N^2)$, which is on the order of M_1 vector-matrix multiplies.

Recursive Least Squares (RLS)

Given

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

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

[3] In practice, it is probably more stable to find and update a factorization of this matrix. But the cost is the same.

Recursive Least Squares

Initialize: ($\boldsymbol{y}_0$ appears)

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

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

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

($\boldsymbol{y}_k$ appears)

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

$$\boldsymbol{K}_k = \boldsymbol{P}_k\boldsymbol{A}_k^{\mathrm{T}}$$

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

end for

RLS and the BLUE

Correlation information about the observation noise can easily be incorporated into RLS.

We observe

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

The vectors $\boldsymbol{e}_0, \boldsymbol{e}_1, \ldots, \boldsymbol{e}_k, \ldots$ are random error vectors with zero mean and covariance matrices

$$\boldsymbol{R}_k = \mathrm{E}[\boldsymbol{e}_k\boldsymbol{e}_k^{\mathrm{T}}].$$

We will assume that error vectors at different times are uncorrelated:

$$\mathrm{E}[\boldsymbol{e}_k \boldsymbol{e}_j^{\mathrm{T}}] = \boldsymbol{0}, \quad \forall j \neq k.$$

Given the first set of measurements

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

we estimate $\boldsymbol{x}_*$ using the BLUE

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

Now we define the "information matrix" $\boldsymbol{P}_0$ as

$$\boldsymbol{P}_0 = (\boldsymbol{A}_0^{\mathrm{T}} \boldsymbol{R}_0^{-1} \boldsymbol{A}_0)^{-1},$$

and so

$$\hat{\boldsymbol{x}}_0 = \boldsymbol{P}_0 \boldsymbol{A}_0^{\mathrm{T}} \boldsymbol{R}_0^{-1} \boldsymbol{y}_0.$$

Now we observe another set of observations

$$\boldsymbol{y}_1 = \boldsymbol{A}_1 \boldsymbol{x}_* + \boldsymbol{e}_1.$$

The naive way to solve for the optimal $\hat{\boldsymbol{x}}_1$ using both $\boldsymbol{y}_0$ and $\boldsymbol{y}_1$ is to form

$$\underline{\boldsymbol{y}} = \begin{bmatrix} \boldsymbol{y}_0 \\ \boldsymbol{y}_1 \end{bmatrix}, \quad \underline{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A}_0 \\ \boldsymbol{A}_1 \end{bmatrix}, \quad \underline{\boldsymbol{R}} = \begin{bmatrix} \boldsymbol{R}_0 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}_1 \end{bmatrix}$$

and then solve

$$\hat{\boldsymbol{x}}_1 = (\underline{\boldsymbol{A}}^{\mathrm{T}} \underline{\boldsymbol{R}}^{-1} \underline{\boldsymbol{A}})^{-1} \underline{\boldsymbol{A}}^{\mathrm{T}} \underline{\boldsymbol{R}}^{-1} \underline{\boldsymbol{y}}.$$

Of course as we receive more and more data, this gets harder and harder to do.

But just as before, we can compute $\hat{\boldsymbol{x}}_1$ recursively. We have

$$\hat{\boldsymbol{x}}_1 = \boldsymbol{P}_1 \begin{bmatrix} \boldsymbol{A}_0^{\mathrm{T}} & \boldsymbol{A}_1^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \boldsymbol{R}_0 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}_1 \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{y}_0 \\ \boldsymbol{y}_1 \end{bmatrix}.$$

Since the "big" correlation matrix is block diagonal,

$$\begin{bmatrix} \boldsymbol{R}_0 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}_1 \end{bmatrix}^{-1} = \begin{bmatrix} \boldsymbol{R}_0^{-1} & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{R}_1^{-1} \end{bmatrix},$$

and so

$$\begin{aligned} \hat{\boldsymbol{x}}_1 &= \boldsymbol{P}_1(\boldsymbol{A}_0^{\mathrm{T}}\boldsymbol{R}_0^{-1}\boldsymbol{y}_0 + \boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{R}_1^{-1}\boldsymbol{y}_1) \\ &= \boldsymbol{P}_1(\boldsymbol{P}_0^{-1}\hat{\boldsymbol{x}}_0 + \boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{R}_1^{-1}\boldsymbol{y}_1). \end{aligned}$$

It is also true that

$$\begin{aligned} \boldsymbol{P}_1^{-1} &= \boldsymbol{A}_0^{\mathrm{T}}\boldsymbol{R}_0^{-1}\boldsymbol{A}_0 + \boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{R}_1^{-1}\boldsymbol{A}_1 \\ &= \boldsymbol{P}_0^{-1} + \boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{R}_1^{-1}\boldsymbol{A}_1, \end{aligned}$$

and so

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

where $\boldsymbol{K}_1 = \boldsymbol{P}_1\boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{R}_1^{-1}$ is the "gain matrix".

As before, we can repeat this process as more data arrives, giving us the following algorithm:

RLS BLUE

Initialize: ($\boldsymbol{y}_0$ appears)

$$\boldsymbol{P}_0 = (\boldsymbol{A}_0^{\mathrm{T}} \boldsymbol{R}_0^{-1} \boldsymbol{A}_0)^{-1}$$

$$\hat{\boldsymbol{x}}_0 = \boldsymbol{P}_0 (\boldsymbol{A}_0^{\mathrm{T}} \boldsymbol{R}_0^{-1} \boldsymbol{y}_0)$$

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

($\boldsymbol{y}_k$ appears)

$$\boldsymbol{P}_k = \boldsymbol{P}_{k-1} - \boldsymbol{P}_{k-1} \boldsymbol{A}_k^{\mathrm{T}} (\boldsymbol{R}_k + \boldsymbol{A}_k \boldsymbol{P}_{k-1} \boldsymbol{A}_k^{\mathrm{T}})^{-1} \boldsymbol{A}_k \boldsymbol{P}_{k-1}$$

$$\boldsymbol{K}_k = \boldsymbol{P}_k \boldsymbol{A}_k^{\mathrm{T}} \boldsymbol{R}_k^{-1}$$

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

end for