4.15

Intro

In this exercise we will perform an analysis of the sequential update of the covariance and precision matrices. As stated by the author, this is a smart strategy when we must update the covariance matrix everytime we receive a new point. The exercise is divided in four itens. First, we will prove the recurrence relation that allow us to update the covariance matrix sequentially. Second, we will prove that the result has a better time complexity than the full update, which is the whole point of using the recurrence. In the third and fourth itens, we will do the same thing for the precision matrix.

Solution

a)

In this first item, we must show that the covariance can be sequentially updated as follow:

$$\begin{aligned} C_{n+1} &= \frac{n-1}{n}C_n + \frac{1}{n+1}(x_{n+1}-m_n)(x_{n+1}-m_n)^t \\ nC_{n+1}-(n-1)C_n &= \frac{n}{n+1}(x_{n+1}-m_n)(x_{n+1}-m_n)^t \end{aligned} \tag{1}$$

In (1) we wrote the equation in a more convinient form, to perform the next steps. Now, our job is to expand the LHS to arrive at the RHS. Note that the two terms on the LHS can be expanded into:

$$\begin{aligned} nC_{n+1} &= \sum_{j=1}^{n+1}(x_i-m_{n+1})(x_i-m_{n+1})^t = \sum_{j=1}^{n+1} x_j x_j^t - (n+1)m_{n+1}m_{n+1}^t \\ (n-1)C_n &= \sum_{j=1}^{n}(x_i-m_n)(x_i-m_n)^t = \sum_{j=1}^{n} x_j x_j^t - n m_n m_n^t \end{aligned} \tag{2}$$

In (2) we used $\sum_{j=1}^{n}(x_i-\mu)(x_i-\mu)^t = \sum_{j=1}^{n} x_j x_j^t - n\mu\mu^t$. Substituting (2) in the LHS of (1), we arrive at:

$$nC_{n+1}-(n-1)C_n = x_{n+1}x_{n+1}^t - (n+1)m_{n+1}m_{n+1}^t + n m_n m_n^t \tag{3}$$

Since we want the final result to be expressed in terms of x_{n+1} and m_n, we should expand m_{n+1}.

$$m_{n+1} = \frac{x_{n+1}+nm_n}{n+1} \tag{4}$$

Substituting (4) in (3) we arrive at:

$$\begin{aligned}
&nC_{n+1}-(n-1)C_n = \\
&x_{n+1}x_{n+1}^t - \frac{(n+1)}{(n+1)^2}(nm_n+x_{n+1})(nm_n+x_{n+1})^t + nm_nm_n = \\
&x_{n+1}x_{n+1}^t - \frac{1}{(n+1)}(n^2m_nm_n^t + nm_nx_{n+1}^t + nx_{n+1}m_n^t + x_{n+1}x_{n+1}^t) + nm_nm_n = \\
&\frac{n}{n+1}x_{n+1}x_{n+1}^t - \frac{n}{(n+1)}(m_nx_{n+1}^t + x_{n+1}m_n^t) + \frac{n}{n+1}m_nm_n^t = \\
&\frac{n}{n+1}(x_{n+1}-m_n)(x_{n+1}-m_n)^t
\end{aligned} \tag{5}$$

b)

Before computing the time complexity note that step $n+1$ use the mean of step n. Let's assume that this mean is computed also on step n. Thus:

$$\begin{aligned}
&C_{n+1} = \frac{n-1}{n}C_n + \frac{1}{n+1}(x_{n+1}-m_n)(x_{n+1}-m_n)^t \\
&\text{compute the mean } m_{n+1} = \frac{m_n + x_{n+1}}{n+1}\text{: } O(1) \\
&\text{outer product } (x_{n+1}-m_n)(x_{n+1}-m_n)^t : O(d^2) \\
&\text{rest of operations: } O(1) \\
&\text{total time complexity: } O(d^2)
\end{aligned} \tag{6}$$

Thus, the sequential update is n times faster than the full update.

c)

Now we need to prove the following expression for the sequential update of the precision matrix

$$C_{n+1}^{-1} = \frac{n}{n-1}\left[C_n^{-1} - \frac{C_n^{-1}(x_{n+1}-m_n)(x_{n+1}-m_n)^tC_n^{-1}}{\frac{n^2-1}{n} + (x_{n+1}-m_n)^tC_n^{-1}(x_{n+1}-m_n)}\right] \tag{7}$$

In order to prove this expression we will calculate the inverse of C_{n+1}^{-1} based on the expression derived in item a and the author's hint on matrix inversion lemma for rank-one updates.

$$\begin{aligned}
&C_{n+1}^{-1} = \left[\frac{n-1}{n}C_n + \frac{1}{n+1}(x_{n+1}-m_n)(x_{n+1}-m_n)^t\right]^{-1} = \\
&\frac{n}{n-1}C_n^{-1} - \frac{\frac{n}{n-1}C_n^{-1}\frac{1}{n+1}(x_{n+1}-m_n)(x_{n+1}-m_n)^t\frac{n}{n-1}C_n^{-1}}{1+\frac{1}{n+1}(x_{n+1}-m_n)^t\frac{n}{n-1}C_n^{-1}(x_{n+1}-m_n)} = \\
&\frac{n}{n-1}\left[C_n^{-1} - \frac{nC_n^{-1}(x_{n+1}-m_n)(x_{n+1}-m_n)^tC_n^{-1}}{n^2-1+n(x_{n+1}-m_n)^tC_n^{-1}(x_{n+1}-m_n)}\right] = \\
&\frac{n}{n-1}\left[C_n^{-1} - \frac{C_n^{-1}(x_{n+1}-m_n)(x_{n+1}-m_n)^tC_n^{-1}}{\frac{n^2-1}{n} + (x_{n+1}-m_n)^tC_n^{-1}(x_{n+1}-m_n)}\right]
\end{aligned} \tag{8}$$

d)

Since there is matrix multiplication and matrix inversion in the expression $(C_n^{-1}(x_{n+1}-m_n)(x_{n+1}-m_n)^t C_n^{-1})$, our first instinct it to think that the time complexity is $O(d^3)$. However, note that we do not need to compute the actual inverse C_n^{-1}, because it was determined in the last pass of the recurrence and it is stored in memory. At the same time, the matrix multiplication can be performed more quickly if we use the associative property in our favor to avoid to perform matrix/matrix multiplication. Here it is the best way of performing this multiplication:

$$\begin{aligned}
&C_n^{-1}(x_{n+1}-m_n)(x_{n+1}-m_n)^t C_n^{-1}\\
&A = C_n^{-1}(x_{n+1}-m_n)\colon\ O(d^2)\\
&B = (x_{n+1}-m_n)^t C_n^{-1}\colon\ O(d^2)\\
&C_n^{-1}(x_{n+1}-m_n)(x_{n+1}-m_n)^t C_n^{-1} = AB\colon\ O(d^2)
\end{aligned} \tag{9}$$

Note that we avoided perfoming matrix/matrix multiplication and performed matrix/vector multiplication, vector/matrix multiplication and outer product. Since the quadratic form $(x_{n+1}-m_n)^t C_n^{-1}(x_{n+1}-m_n)$ has also time complexity $O(d^2)$, the total time complexity is $O(d^2)$. Therefore, the recurrence is d times faster than the traditional approach.

Conclusion

In this exercise we performed an analysis of the sequential update of the covariance and precision matrices. We proved recurrence relations between C_n and C_{n-1}/C_n^{-1} and C_{n-1}^{-1}. As we suspected the recurrence on the covariance matrix makes the algorithm n times faster. This is very intuitive, because instead of summing n scatter matrices, we just add one scatter matrix corresponding to the new data point. This recurrence is very efficient when n start to grow to really big numbers.

As for the precision matrix, we discovered that the recurrence update is d times faster than the tradtional approach. Thus, this is specially interesting for data with high dimensionality (big d). On the other hand, this approach does not gets better as n increases.