Partitioning the recording and estimate Q separately at each part doesn’t add a lot, but makes things more complicated. Let’s forget about it.

Basically, some thoughts from following investigation:

1. **When “window” is used, it is somewhat robust to Q, if Q is not too small.**
2. Since (1), adding covariance or not doesn’t influence results a lot (if optimization of Q with covariance intercept is fine)
3. Optimization of Q with covariance seems non-convex. That makes things more complicated, but doesn’t add too much as mentioned in (2).
4. **Again since (1), when the optimized Q is too weird/ too slow, we can just set Q to something like 1e-3, and optimize the window size alone?**

# Optimize Q, when Q is non-diagonal

Here, I still use previous interior-point method (i.e. assume convexity), for simplicity of the first trial. The bound for covariance intercept is set as [-1e-3, 1e-3].

## Simulation on Q-tune

The code can be found in “~\demo\single\demo\_QTune\_cov.m” and “~\demo\single\demo\_QTune\_cov\_v2.m”. The difference between these 2 is that v2 has not constraint on diagonal Q, while v1 is more structured (same non-intercept Q within group).

True Q: diag([1e-4 1e-5])

Optimized Q:

Qoptmatrix =

1.0e-04 \*

0.8241 -0.0130

-0.0130 0.0734

OK, looks fine.

## Simulation on 3 cases

The code can be found in “~\demo\single\demo\_newCases\_fisher\_window\_v4.m”

**Case 3:**

Qoptmatrix =

1.0e-03 \*

1.0000 0.1139

0.1139 0.0500

|  |  |  |
| --- | --- | --- |
| Log-likelikhood againt to window size | **Current fitting (with covariance)** | Previous fitting (no covariance) |
|  |  |  |

With or without covariance term, the results are similar.

Let’s try the annoying case 2.

**Case 2:**

Qoptmatrix =

1.0e-03 \*

0.9673 0.3659

0.3659 0.1383

Under this Q, the algorithm will often go singular. Let’s see the performance under different window sizes.

|  |  |
| --- | --- |
| K = 10 | K = 50 |
|  |  |

Previous results:

|  |  |
| --- | --- |
| No Q tune, no window. Q = diag([1e-3 1e-3]) | Llhd vs. window size |
|  |  |
| No Q tune: Q = diag([1e-3 1e-3]), k = 5 (optimized window size for filtering likelihood) | No Q tune: Q = diag([1e-3 1e-3]), k = 15 |
|  |  |
| No Q tune: Q = diag([1e-3 1e-3]), k = 100 | Q tune: Q = diag([2.5e-6 2.4e-7]), k = 400 |
|  |  |

Err… It seems the optimized Q is not correct… This may suggest the non-convexity of the objective function. But it seems diagonal Q works well enough for “window”. Moreover, the “window” is robust to diagonal Q if it’s not super small.

Personally, I think diagonal assumption is fine. Anyway, I will try global search algorithm for Q later if necessary. **Do you have some suggested global search (non-convex) algorithm**? Many of them are pretty cumbersome, as far as I know…

# V1 data, window version

The code can be found in “\demo\v1\v1\_single\_demo\_window.m”

I still use neuron 13 at this point.

Let’s first see the performance with no Q-tuning, but with tuning of window size.

|  |  |  |
| --- | --- | --- |
| Q = diag([repmat(1e-3,nknots+1,1); repmat(1e-3,Gnknots + 1,1)]); | Q = diag([repmat(1e-4,nknots+1,1); repmat(1e-4,Gnknots + 1,1)]); | Q = diag([repmat(1e-5,nknots+1,1); repmat(1e-5,Gnknots + 1,1)]); |
|  |  |  |
| K = 10 | K = 20 | K = 40 |
|  |  |  |
|  |  |  |

It seems that the window is kind of robust to Q, if it is not too small (e.g. Q = 1e-5\*… is somewhat too small)

Let’s see the trace of window size under Q = 1e-3\*… and Q = 1e-4\*…

|  |  |  |
| --- | --- | --- |
|  | Q = 1e-3\*… | Q = 1e-4\*… |
| K=1 |  |  |
| K=5 |  |  |
| K=10 |  |  |
| K=20 |  |  |
| K=50 |  |  |
| K=100 |  |  |

It seems when the “window” is turned on, it is robust to Q to some degree

OK, let’s turn on the Q-tuning.

|  |  |  |  |
| --- | --- | --- | --- |
| Diagonal, no constraint |  |  |  |
| Diagonal, same Q for non-intercept within group |  |  |  |
| Add intercept in covariance, no constraint on diagonal (very hard to optimize) | | | |
| Add intercept in covariance, same Q for non-intercept within group on diagonal (not stable…) | | |  |

Again, in this case, it seems adding covariance makes things more complicated, but it doesn’t improve things a lot. If necessary, I will try the non-convex optimization of Q later.

Also, we can see that the Q-tuned results is very similar to Q = 1e-3\*… and Q = 1e-4\*… at the top of this section. That again suggests that the “window” is robust to Q, if it is not too small.

**Maybe if** **the optimized Q is too weird/ too slow, we can just set Q to something like 1e-3, and optimize the window size alone?**