

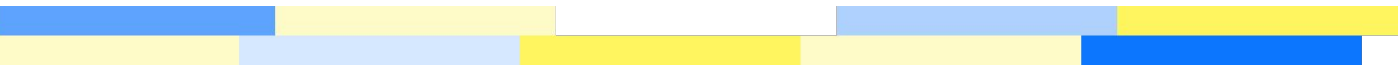


Leaving reviews and responding to reviews

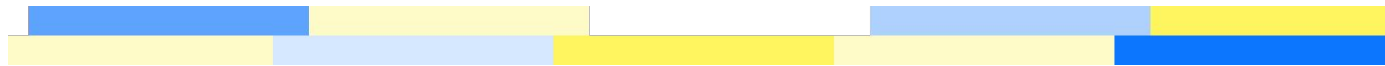
Childhood Cancer
Data Lab

x





Tips for leaving PR reviews



When leaving a review...

Start by looking through the linked issue and any associated discussion

As needed, familiarize yourself with relevant code *before* the PR code

- If you are entirely unfamiliar with the code, maybe you're not the best person to review this!

Read the PR comment carefully, including any suggestions from the author for where/how you should focus your review

- This may help you determine how to perform review: Just read through on GitHub, locally checkout out the branch and test the code, or a little of both!
- Depending on the circumstance, you may want to leave specific in-line comments or overall high-level comments, or a little of both!

We're going to see a real life PR review in a moment!





Tips for responding to PR reviews




Start by taking in the *big picture*

It's tempting to dive right in and respond to individual comments, but you'll gain more context about why the reviewer left certain comments by reading the whole review first

- For example, maybe that the reviewer misunderstood something you were trying to do, which *might happen if* you didn't clearly express the goals you had in mind for the review
- By taking in the full review, you might identify other areas to improve that weren't specifically reviewed, but conceptually mentioned

When re-requesting review...

Don't *just* re-request review - communicate with your reviewer!


- Summarize the changes you made in an overall comment, including any changes you made that the reviewer did not request
 - Bonus: Communicate specific changes you made by sharing the commit hash (*stay tuned!*)
 - Maybe there were some requested changes you didn't make - explain why, or ask the reviewer for more clarification
 - Same rules apply: Don't be a jerk. Your reviewer is on your side, don't take comments personally! It's all room for growth
- 

Make your reviewer's life a little easier

Tell them where/when you addressed their comment

- GitHub will automatically style and link commit hashes


```
bin/post_process_sce.R Outdated
103 - }
104 - metadata(sce)$scpca_filter_method <- paste0(metadata(sce)$scpca_filter_method,
105 -                                           adt_filter_string)
92 + metadata(sce)$scpca_filter_method <- metadata(sce)$scpca_filter_method
```

 **jashapiro** on Jun 2 Member ...


What is meant to be happening here?

I *think* what we want here is an accounting of the adt filter method (`adt_scpca_filter_method` ?) which should reflect whether negative controls were used.



 **sjspielman** on Jun 2 Member Author ...

What is happening here is I had to remove `paste` code but then apparently did not look at the resulting line to see... we don't need this line at all!
But yes, let's add `adt_scpca_filter_method` as a separate metadata "cleanTagCounts".



 **sjspielman** on Jun 2 Member Author ...

- Silly line removed in [39d3a43](#)
- Added adt metadata in [18783e9](#)

Make your reviewer's life a little easier


Tell them where/when you addressed their comment

- GitHub will automatically style and link commit hashes

BONUS! An example of when I (the PR author!) did not do a great job looking at my **diff** before filing the PR and/or requesting review.


Extra time was spent on review for something I could have caught before hand by just slowing down a little bit.


```
bin/post_process_sce.R Outdated
103 - }
104 - metadata(sce)$scpca_filter_method <- paste0(metadata(sce)$scpca_filter_method,
105 -                                           adt_filter_string)
92 + metadata(sce)$scpca_filter_method <- metadata(sce)$scpca_filter_method
```

 **jashapiro** on Jun 2 Member ...

What is meant to be happening here?


I *think* what we want here is an accounting of the adt filter method (`adt_scpca_filter_method` ?) which should reflect whether negative controls were used.




 **sjspielman** on Jun 2 Member Author ...

What is happening here is I had to remove `paste` code but then apparently did not look at the resulting line to see...we don't need this line at all!

But yes, let's add `adt_scpca_filter_method` as a separate metadata "cleanTagCounts".




 **sjspielman** on Jun 2 Member Author ...

- Silly line removed in [39d3a43](#)
- Added adt metadata in [18783e9](#)

If your reviewer left in-line suggestions...

You can either accept suggestions directly:


```
build-celltype-ref.nf
...      ...      @@ -83,6 +78,9 @@ process generate_cellassign_refs {
83      78
84      79      workflow build_celltype_ref {
85      80
81      + // read in json file with all reference paths
82      + ref_paths = Utils.getMetaVal(file(params.ref_json), "Homo_sapiens.GRCh38.104")
```


 **sjspielman** 2 days ago Member ...

Do you mean...

Suggested change

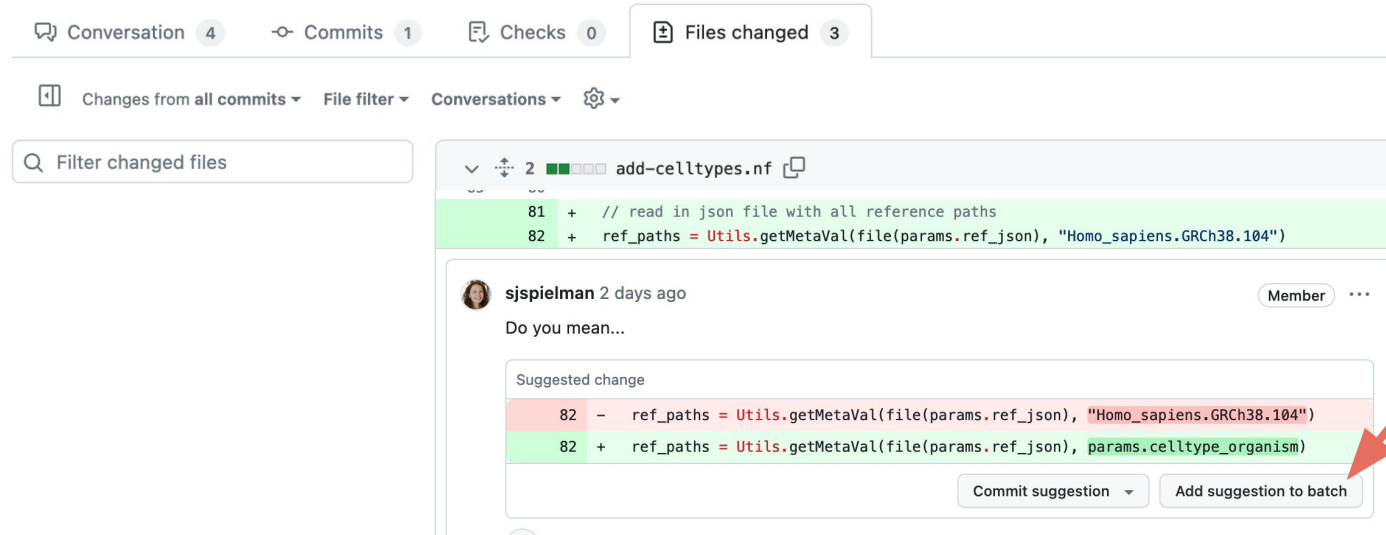
```
82 - ref_paths = Utils.getMetaVal(file(params.ref_json), "Homo_sapiens.GRCh38.104")
82 + ref_paths = Utils.getMetaVal(file(params.ref_json), params.celltype_organism)
```



 Reply...

If your reviewer left in-line suggestions...

Or, in particular if there are *a lot* of suggestions, use "Add suggestion to batch" in the "Files Changed" tab



The screenshot shows a GitHub pull request interface. At the top, there are tabs for "Conversation" (4), "Commits" (1), "Checks" (0), and "Files changed" (3). Below the tabs, there are filters for "Changes from all commits", "File filter", "Conversations", and a settings icon. A search bar on the left is labeled "Filter changed files". The main content area shows a file named "add-celltypes.nf" with two lines of code:

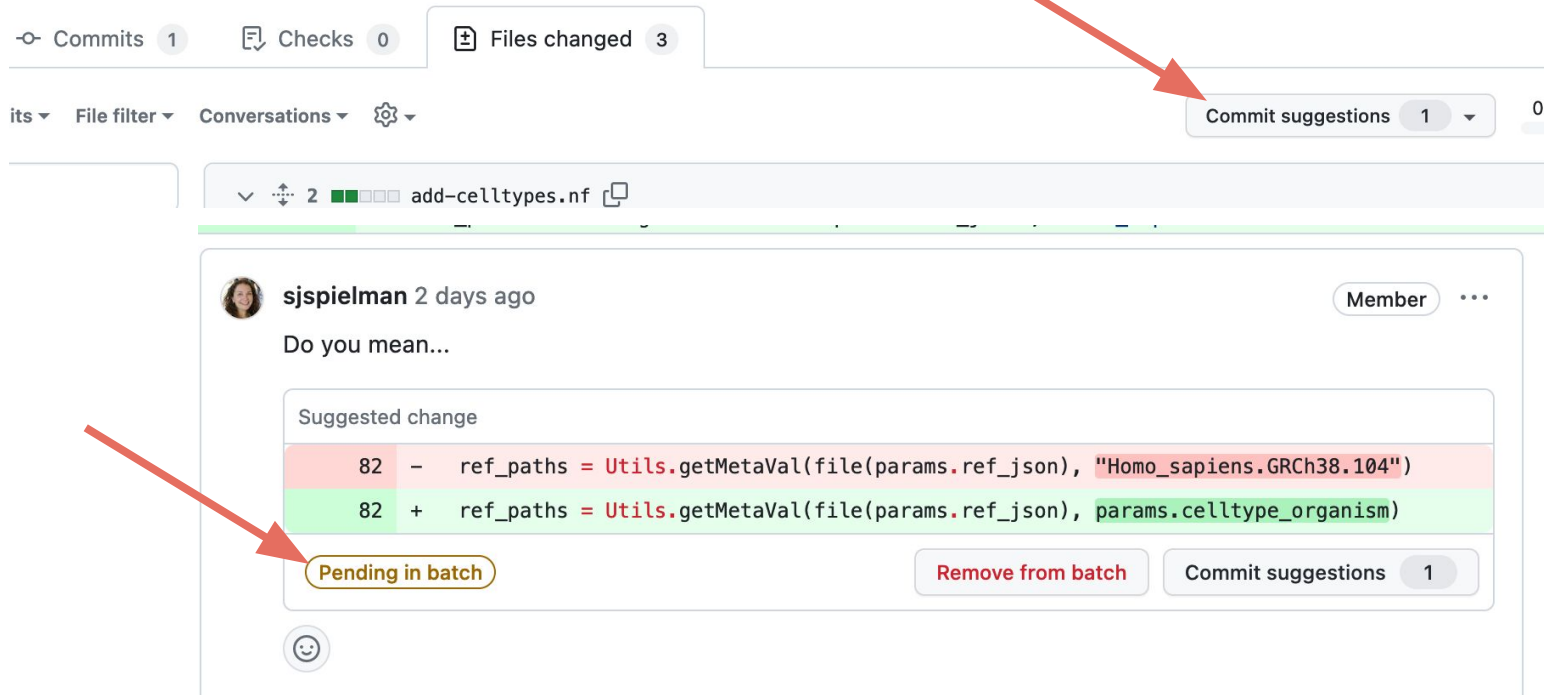
```
81 + // read in json file with all reference paths
82 + ref_paths = Utils.getMetaVal(file(params.ref_json), "Homo_sapiens.GRCh38.104")
```

Below the code, a reviewer named "sjspielman" (Member) has commented "Do you mean..." and provided a "Suggested change". The suggested change shows two lines of code:

```
82 - ref_paths = Utils.getMetaVal(file(params.ref_json), "Homo_sapiens.GRCh38.104")
82 + ref_paths = Utils.getMetaVal(file(params.ref_json), params.celltype_organism)
```

At the bottom of the suggested change, there are two buttons: "Commit suggestion" and "Add suggestion to batch". A red arrow points to the "Add suggestion to batch" button.

When all suggestions are added, commit them all at once



The screenshot shows a GitHub pull request interface. At the top, there are tabs for 'Commits 1', 'Checks 0', and 'Files changed 3'. Below these, there are navigation options: 'its', 'File filter', 'Conversations', and a settings icon. On the right side, there is a 'Commit suggestions 1' button with a dropdown arrow and a '0' indicator. A red arrow points from the top right towards this button.

The main content area shows a commit suggestion from user 'sjspielman' 2 days ago, with a 'Member' badge. The suggestion text is 'Do you mean...'. Below this is a 'Suggested change' box containing two lines of code:


```
82 - ref_paths = Utils.getMetaVal(file(params.ref_json), "Homo_sapiens.GRCh38.104")
82 + ref_paths = Utils.getMetaVal(file(params.ref_json), params.celltype_organism)
```

Below the code, there is a 'Pending in batch' button, a 'Remove from batch' button, and a 'Commit suggestions 1' button. A red arrow points from the left towards the 'Pending in batch' button.

Suggestions are fantastic, but *always check them*

Suggestions are usually not pieces of tested code - you will want to make sure they actually work as expected when responding to review (aka, run the suggested code!)

Always check the spacing! Suggestions can sometimes mess this up in unanticipated ways...

 **Caution!** Accepting a suggestion will resolve the review comment, which may not be what you want to do!

- Protip: Limit suggestion comments to suggestions, and leave other comments elsewhere

You can open new issues based on PR comments

- Keeps the project moving along - otherwise, we could be here *a while*
- Keeps each PR in a reasonable "bite-sized" scope

```
templates/qc_report/cite_qc.rmd
```

```
42 - arrange(desc(mean)) |>
42 + # ensure `target` is the first level of target_type
43 + mutate(target_type = forcats::fct_relevel(target_type, "target")) |>
44 + arrange(target_type, desc(mean)) |>
```



jashapiro last month

Member ...

Oh, one more comment, which I don't want to make hold this up: When we have a lot of ADTs, this table gets kind of unreasonably long. We could limit this to a set number of ADTs, or collapse the table, but I'm not really sure what the best option is here. Maybe a JS table like <https://glin.github.io/reactable/> ? This would be a whole separate thing, and I don't think it is worth doing in this PR, or for this release.



sjspielman last month

Member Author ...

When we have a lot of ADTs, this table gets kind of unreasonably long.

yes, *yes it does*.

I'll open an issue for circling back to this.



Update QC report tables #371

Open sjspielman opened this issue last month · 0 comments



sjspielman commented last month

Member ...

From this [comment](#)

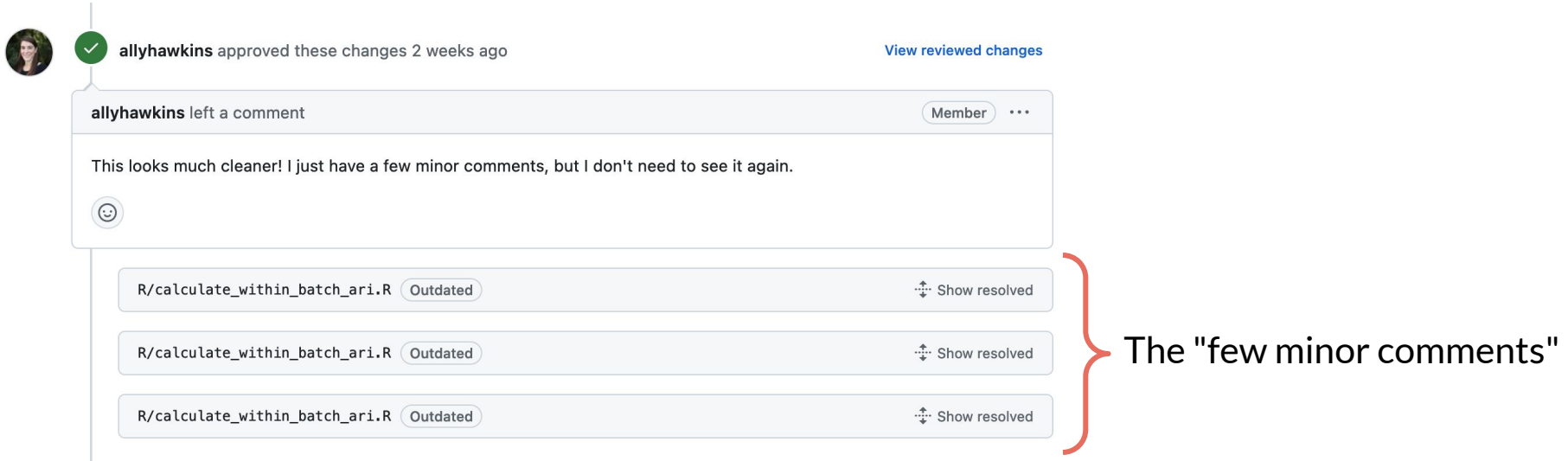
Oh, one more comment, which I don't want to make hold this up: When we have a lot of ADTs, this table gets kind of unreasonably long. We could limit this to a set number of ADTs, or collapse the table, but I'm not really sure what the best option is here. Maybe a JS table like <https://glin.github.io/reactable/> ?

We should take some time to revisit tables overall, including but not limited to excessively long tables. Another lightweight (aka no additional dependencies) approach could be `DT::datatable()`, but we should probably explore a couple options.



Don't rush to merge when PR has been approved

You still need to read the review comments! For example, depending on a team's policy/culture, it may be acceptable for a reviewer to approve **under the assumption you'll make a couple more small changes.**



The screenshot shows a GitHub pull request review interface. At the top, a user named allyhawkins has approved the changes, with a green checkmark icon and the text "allyhawkins approved these changes 2 weeks ago". To the right of this is a link that says "View reviewed changes". Below the approval is a comment box from allyhawkins that says "This looks much cleaner! I just have a few minor comments, but I don't need to see it again." Below the comment box are three separate review comments, each with the text "R/calculate_within_batch_ari.R" and a status of "Outdated". To the right of each comment is a "Show resolved" button. A red bracket on the right side of the image groups these three outdated comments together, with the text "The 'few minor comments'" next to it.

allyhawkins approved these changes 2 weeks ago [View reviewed changes](#)

allyhawkins left a comment Member ...

This looks much cleaner! I just have a few minor comments, but I don't need to see it again.

R/calculate_within_batch_ari.R Outdated [Show resolved](#)

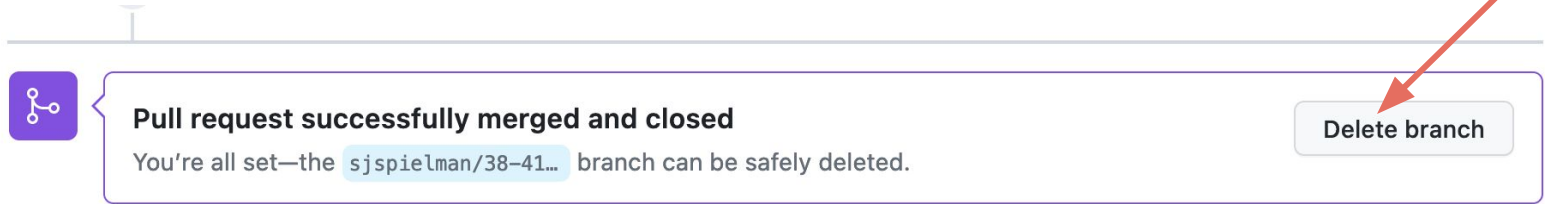
R/calculate_within_batch_ari.R Outdated [Show resolved](#)

R/calculate_within_batch_ari.R Outdated [Show resolved](#)

The "few minor comments"

Don't forget to delete your branch after merging!

Especially if the branch was stacked!



(Note that this won't delete your *local* copy of the branch.)



Last but not least, let's review a PR

