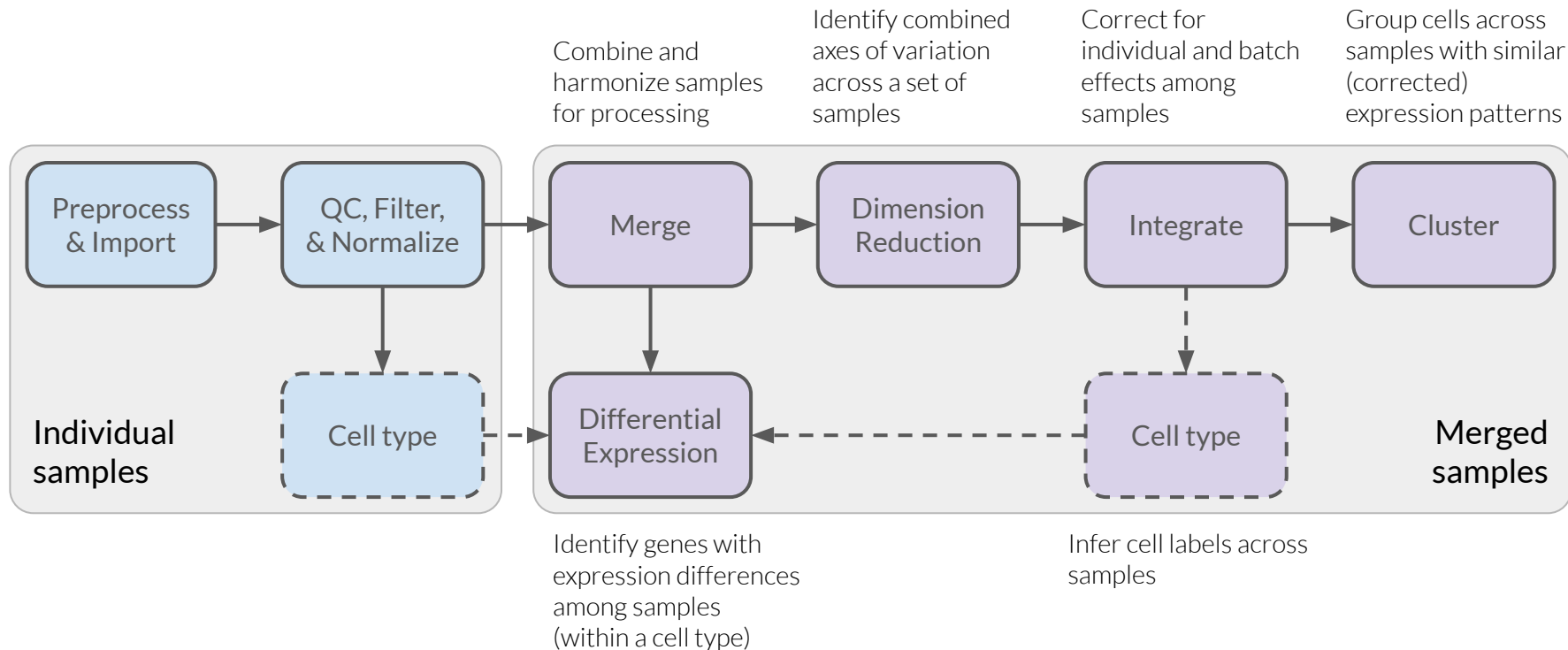




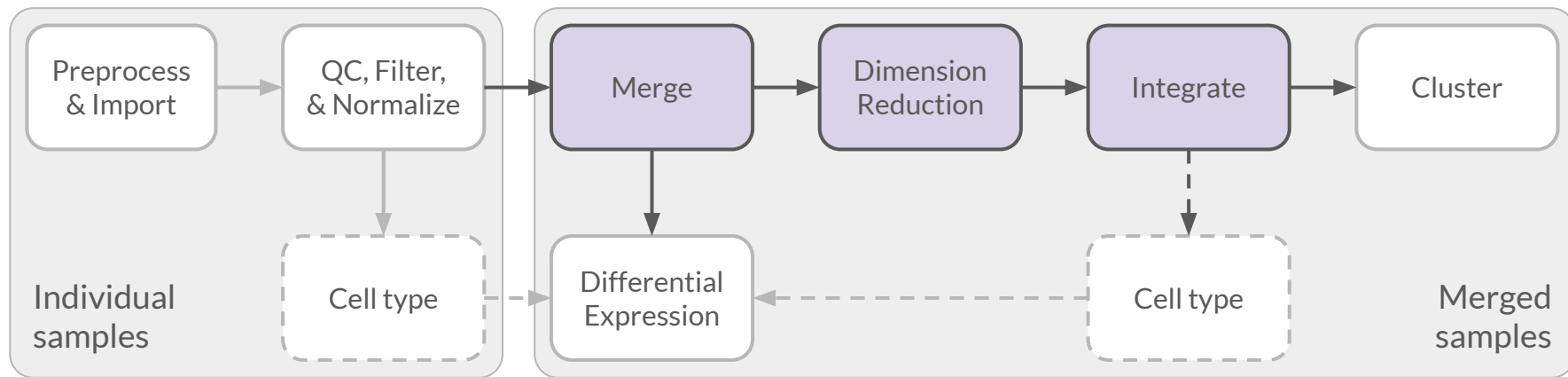
Integrating Different Samples in Single-cell RNA-seq

The Data Lab

Working with multiple samples in scRNA-seq



Integration in scRNA-seq overview



Why integrate samples?

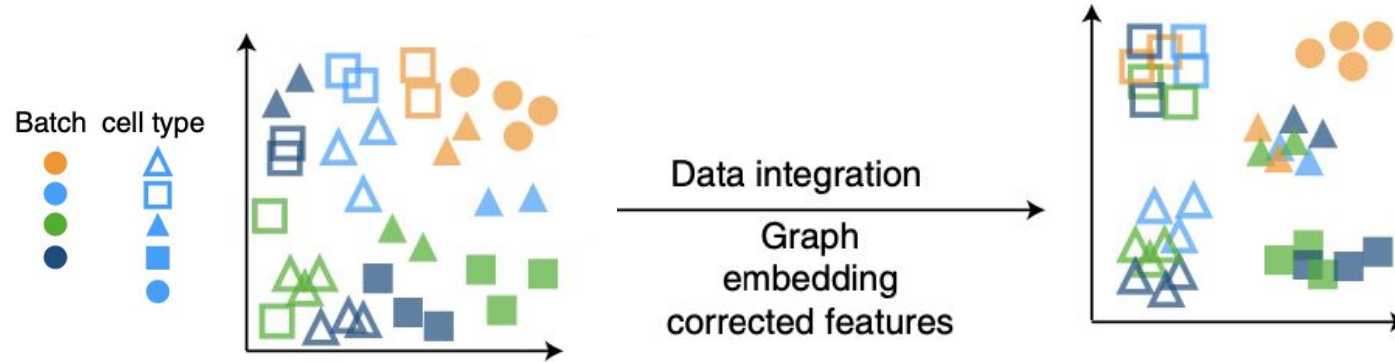
The goal of integration is to mitigate the *batch effects* caused by technical variation across samples, while still preserving biological information.

- Let's call each sample a "batch" of cells
- Cells in a given sample will share some technical variation
- This becomes a problem when we want to jointly consider several samples
 - Cells within a given sample appear more similar than they are, simply because they're from the same sample.
- To compare cells across samples, we need to remove this batch-level technical variation. Then, we can hopefully hone in on the more interesting biological variation 🕵️

What can('t) integration do for you?

- Integration is performed on reduced dimension representations (often principal components)
 - Integration also *returns* reduced dimension representations for downstream use
 - Some integration methods will "back-calculate" corrected gene expression values, but these aren't as important as you think!
 - For example, we do *not* use these for differential expression (stay tuned for more!)
 - Recommended reading on when to use, and not to use, corrected expression values: <http://bioconductor.org/books/3.20/OSCA.multisample/using-corrected-values.html>
- Integration allows us to...
 - Jointly visualize **cells** from multiple datasets
 - Jointly cluster **cells** from multiple datasets
 - Annotate or identify similar **cell types** across datasets

What does successful integration look like?



Before integration, the primary "clustering" is by batch

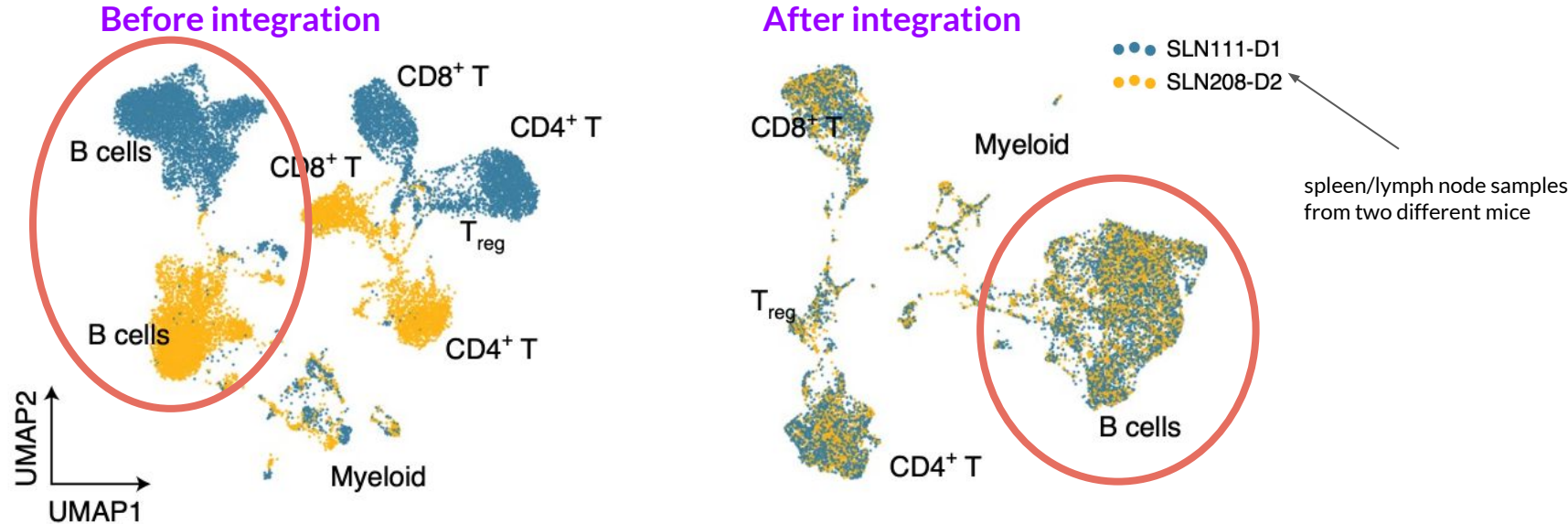
- Orange tends to group with orange, green with green, etc.

After successful integration:

- Batches show lots of mixing
- Cell types ("biology") cluster together, and do *not* show lots of mixing

Successful integration depends on *shared information* across batches.

Example of (what looks like!) successful integration



How to evaluate integration



- Compare before and after UMAP vibes
 - Before integration, batches (datasets) will mostly cluster together
 - After integration...
 - Batches should not group together but should be highly mixed across the UMAP
 - Biologically similar cells (tissue, cell type, disease vs healthy) should group together
 - Usually, when it fails, it fails.
- There are several metrics for evaluating batch correction
 - Luecken *et al.* (2022) is an excellent reference <https://doi.org/10.1038/s41592-019-0619-0>
 - Caution: Metrics do not measure "was integration successful," but other proxies which sometimes can help us tell if integration was successful (or at least not unsuccessful)

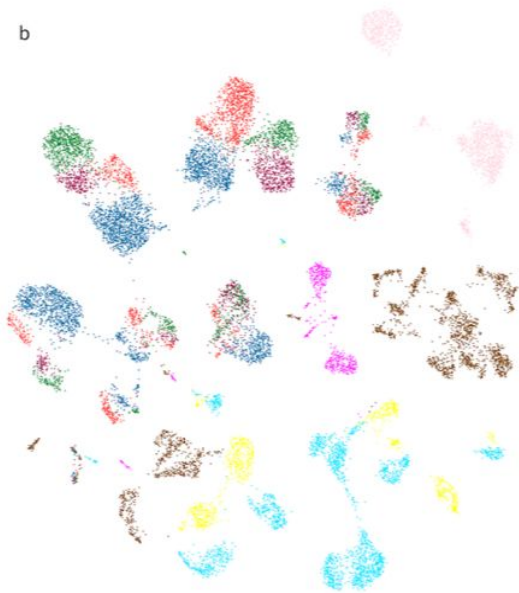
How I stopped worrying and learned to love (the) UMAPs

- Some examples from Luecken et al. (2022)
 - Luecken, M.D., Büttner, M., Chaichoompu, K. et al. Benchmarking atlas-level data integration in single-cell genomics. (2022). <https://doi.org/10.1038/s41592-021-01336-8>
 - Panels from Figure S13 are shown on the next two slides
 - https://static-content.springer.com/esm/art%3A10.1038%2Fs41592-021-01336-8/MediaObjects/41592_2021_1336_MOESM1_ESM.pdf

Top 4 "best" integration methods

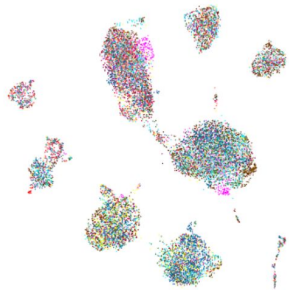
Batch

b

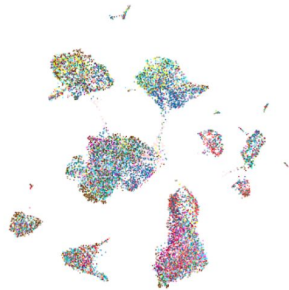


● celseq
 ● celseq2
 ● fluidigm1
 ● inDrop1
 ● inDrop2
 ● inDrop3
 ● inDrop4
 ● smarter
 ● smartseq2

Harmony (embed)
unscaled/HVG



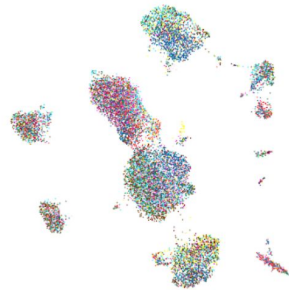
Seurat v3 RPCA (gene)
scaled/HVG



Seurat v3 CCA (gene)
scaled/HVG



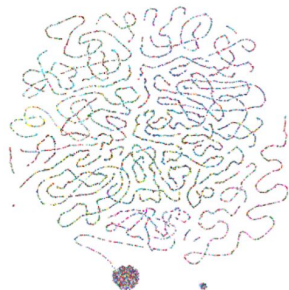
fastMNN (embed)
scaled/HVG



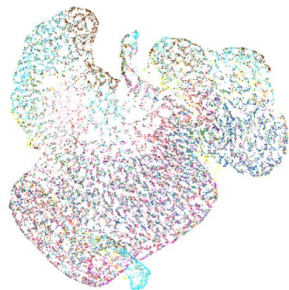
LIGER (embed)
unscaled/FULL



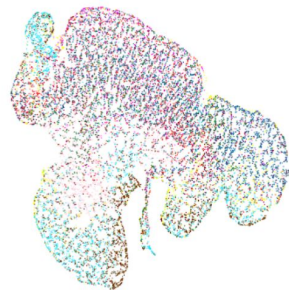
scGen* (gene)
scaled/FULL



SAUCIE (embed)
unscaled/FULL



SAUCIE (gene)
unscaled/FULL

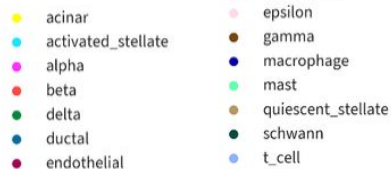
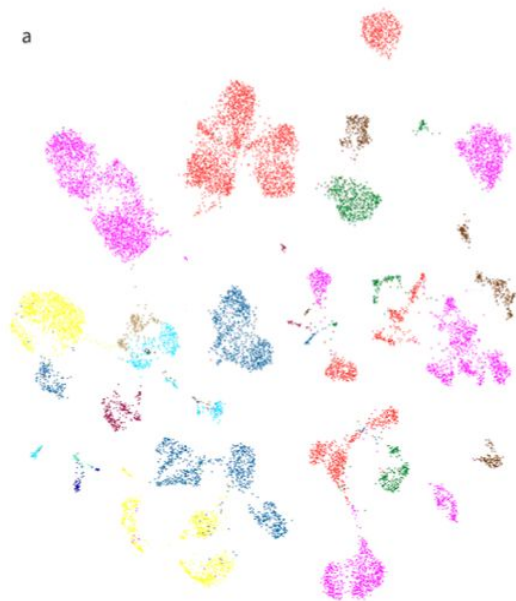


Bottom 4 "worst" integration methods

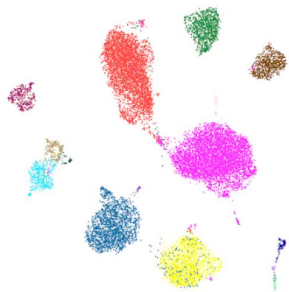
Top 4 "best" integration methods

Cell Type

a



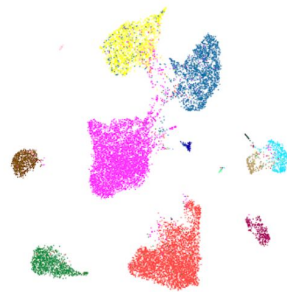
Harmony (embed)
unscaled/HVG



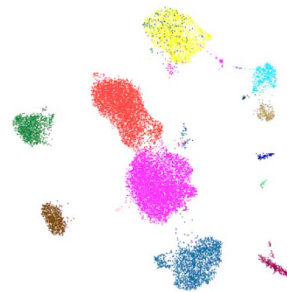
Seurat v3 RPCA (gene)
scaled/HVG



Seurat v3 CCA (gene)
scaled/HVG



fastMNN (embed)
scaled/HVG



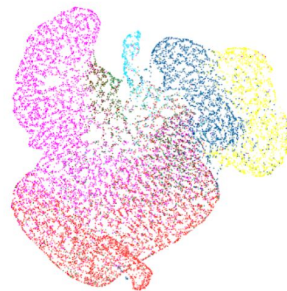
LIGER (embed)
unscaled/FULL



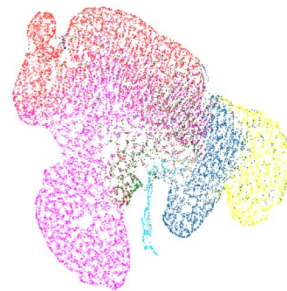
scGen* (gene)
scaled/FULL



SAUCIE (embed)
unscaled/FULL

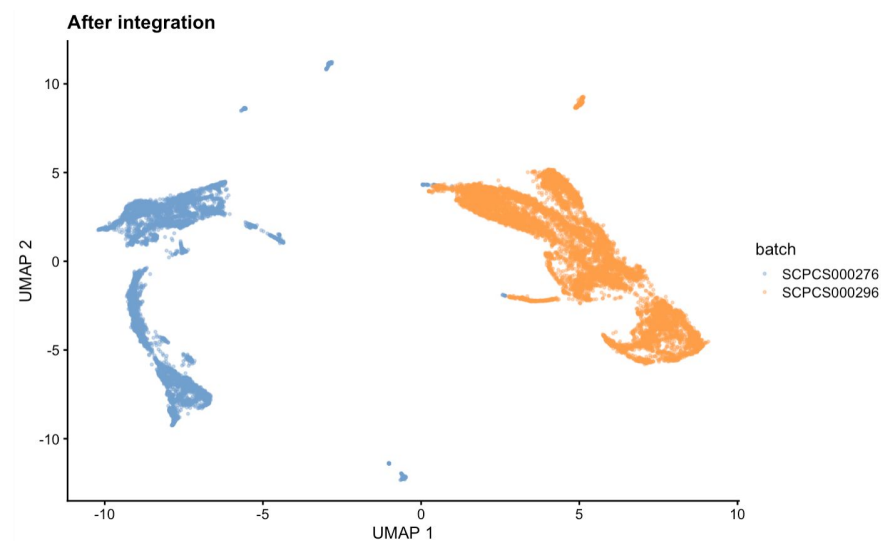
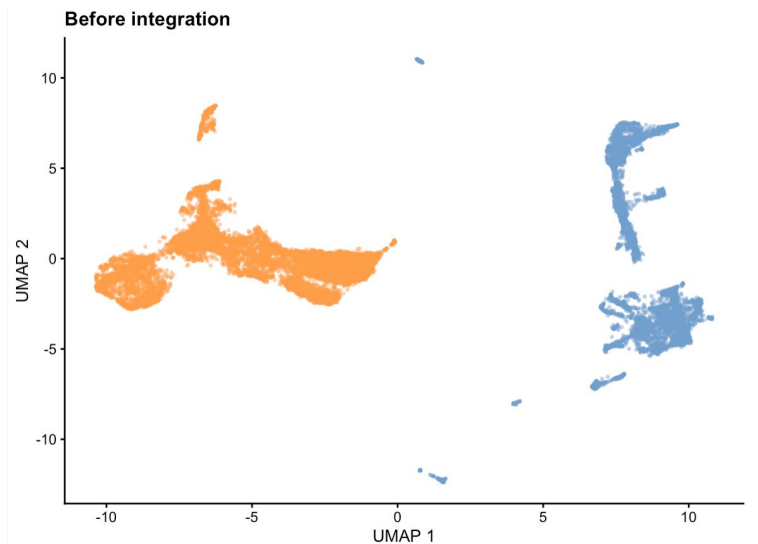


SAUCIE (gene)
unscaled/FULL

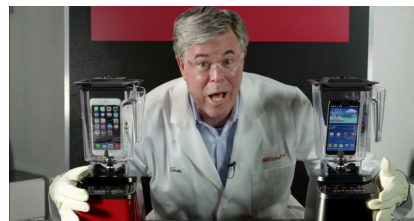


Bottom 4 "worst" integration methods

An example of failed integration

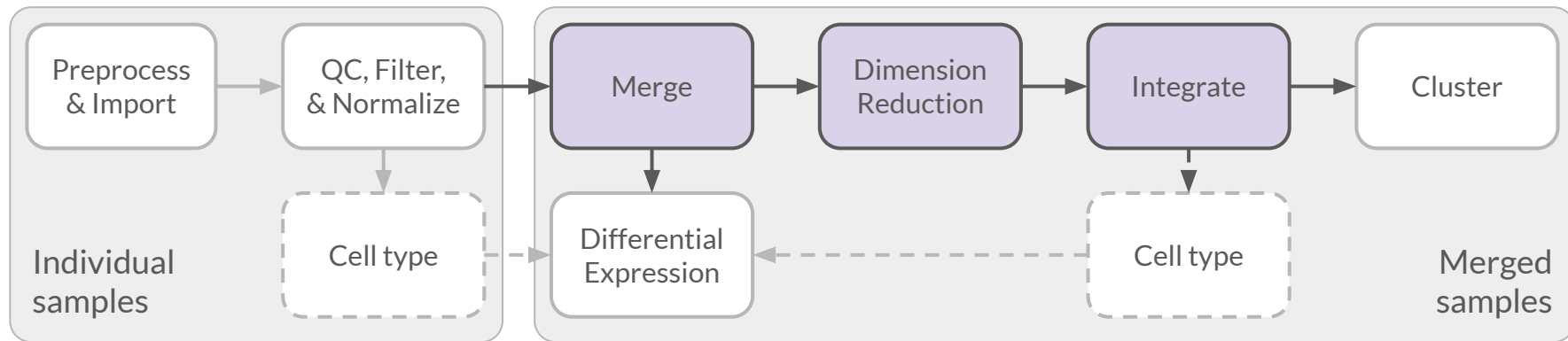


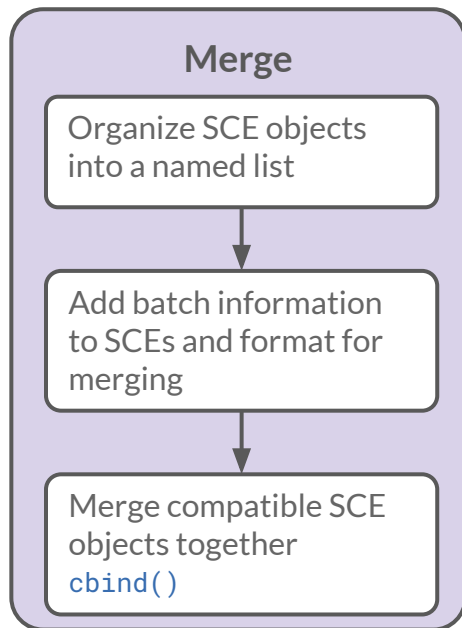
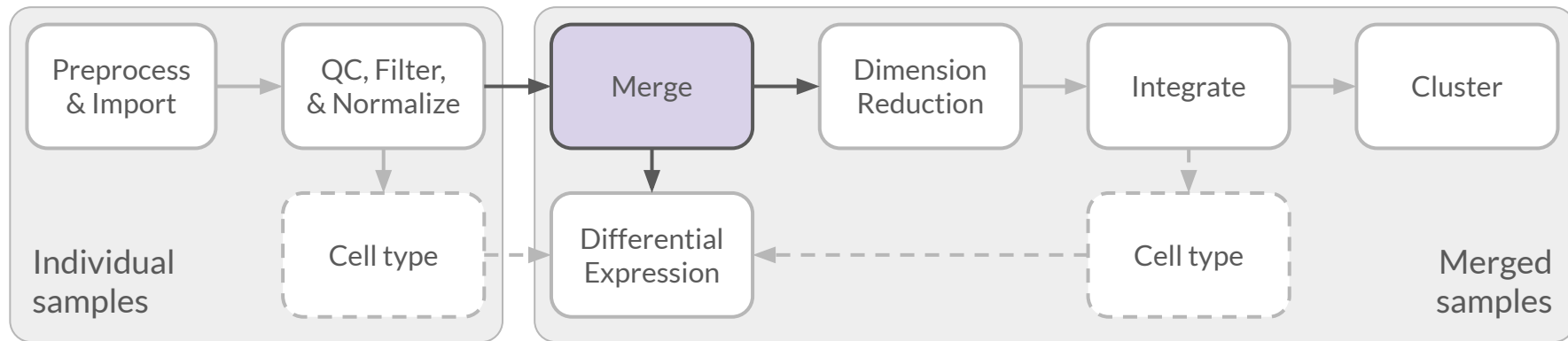
Will it integrate?



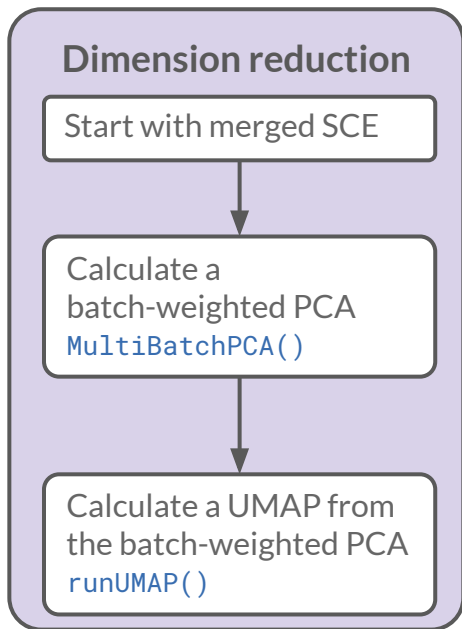
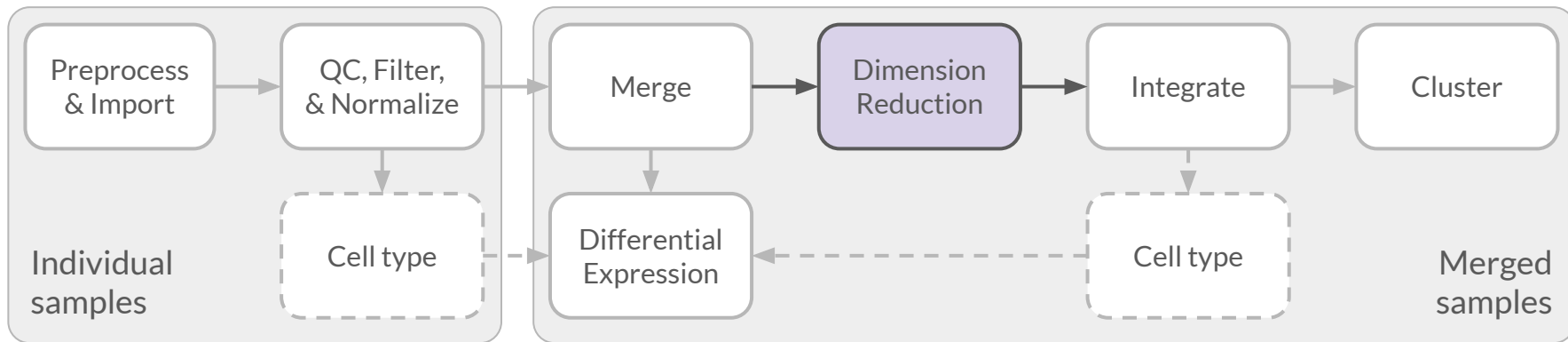
- Datasets that don't have shared cell types or states will be hard to integrate
 - Patient and xenograft
 - Healthy and normal
 - Data from different tissue types
 - Data from different organisms
- The extent of "overlap" among datasets may also influence which integration method you should use, along with the results themselves

Integration in scRNA-seq overview



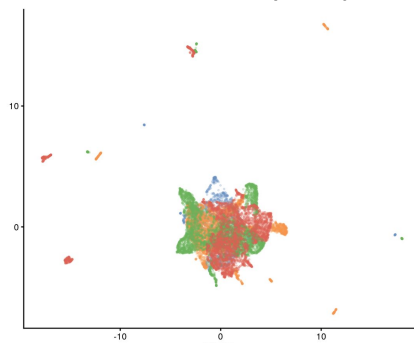


- It's useful to *merge SCEs together* for many downstream analyses, but this merging requires some bookkeeping:
 - After merging, how can we still tell which batch (sample) each cell came from?
 - We need to add this information into SCEs
 - Are SCEs formatted such that R will let us merge them?
 - They need to have compatible column and row names

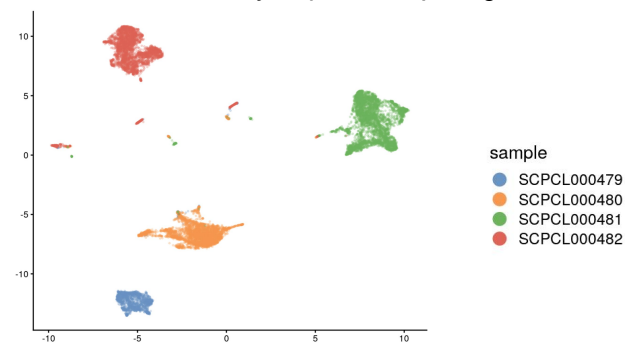


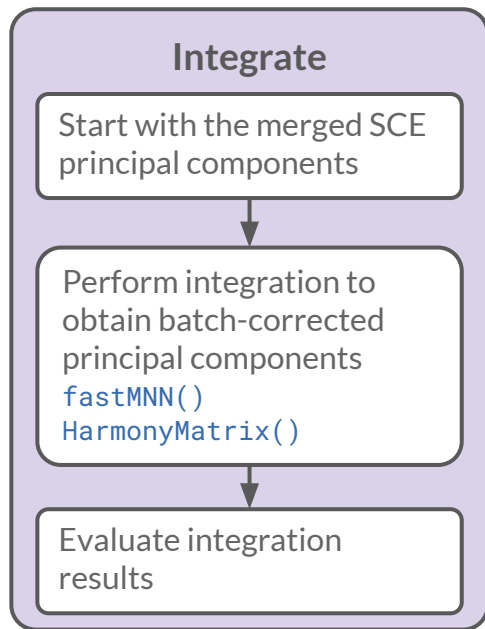
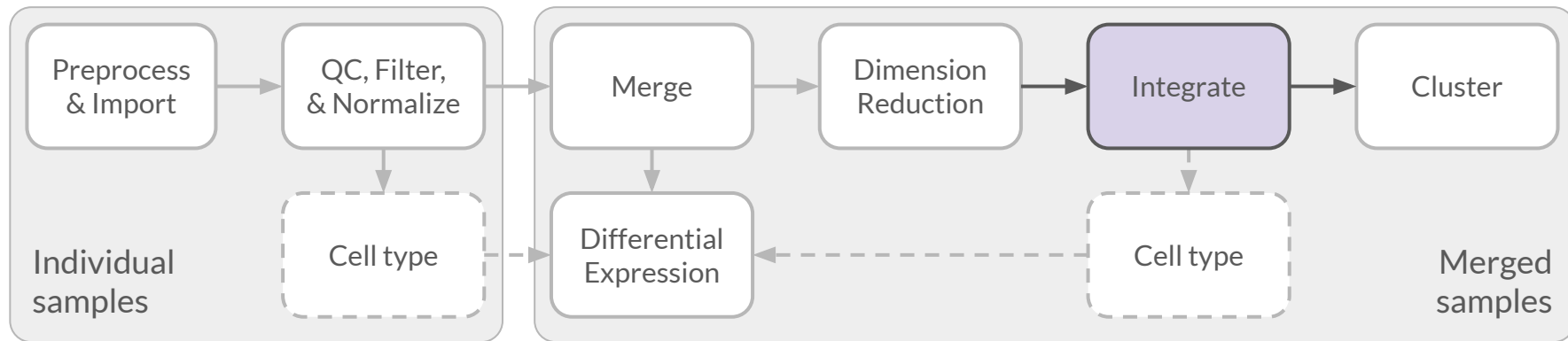
- Dimension reduction techniques like PCA and UMAP start by scaling data to be centered at 0.
- To use PCA/UMAP across samples, we need to calculate the variation jointly

PCA/UMAP calculated separately on each sample



PCA/UMAP calculated jointly on all samples together






- We can evaluate results by...
 - Comparing before/after UMAPs
 - Calculating metrics that tell us how well cells and batches mix
- If integration is successful, we should see...
 - Batches are well-mixed across the UMAP
 - Cell types (or similar biological grouping, if known) group together separately
 - Remember: Success depends on overlap among batches

Let's have a closer look at methods we'll be using

- **MNN: Mutual nearest neighbors**

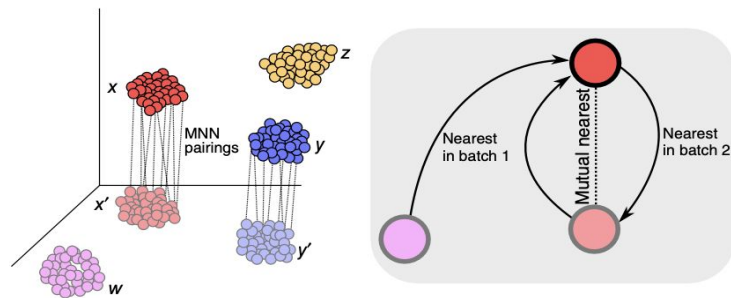
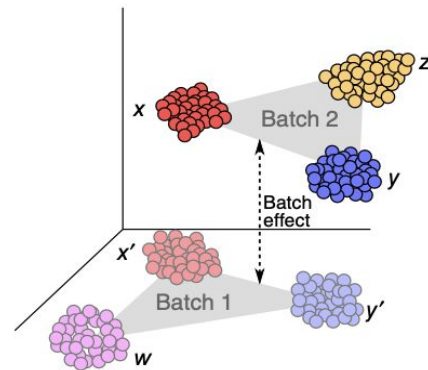
- Specifically, we'll use FastMNN 
- Haghverdi, L, Lun, A, Morgan, M, et al. *Batch effects in single-cell RNA-sequencing data are corrected by matching mutual nearest neighbors.* (2018) <https://doi.org/10.1038/nbt.4091>

- **Harmony**

- Korsunsky, I, Millard, N, Fan, J, et al. *Fast, sensitive and accurate integration of single-cell data with Harmony.* (2019) <https://doi.org/10.1038/s41592-019-0619-0>

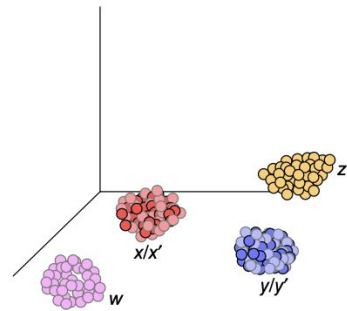
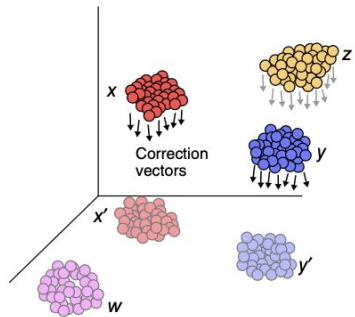
Mutual nearest neighbors batch correction

- Imagine we have 2 batches, each with 3 cell types
 - Red (x) and blue (y) are shared but pink (w) and yellow (z) are not!
 - Before beginning integration, cosine distances are first calculated among pairs of cells *within each sample*
 - This enables expression profile comparisons and sets up the data for integration
- First, we identify pairs of cells with mutually similar expression profiles
 - These are our "mutual nearest neighbors"



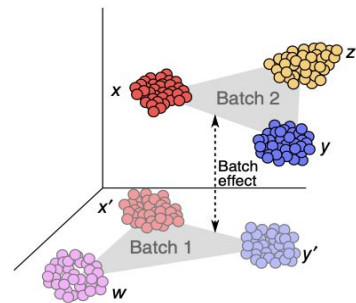
Mutual nearest neighbors batch correction

- Next, compute a batch correction vector for each MNN pair
- Finally, calculate the weighted average of these vectors to get cell-specific batch corrections to perform the final integration
 - Note that **w** and **z** don't "look" as "integrated"! Why?



Some assumptions that MNN makes

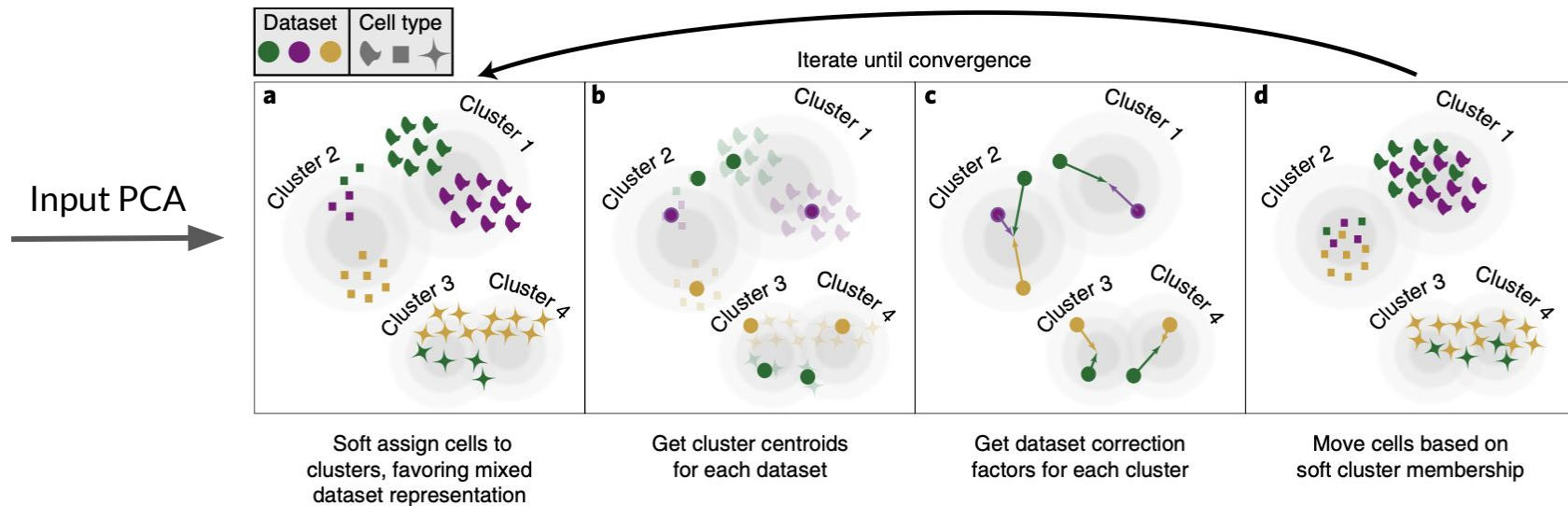
- At least one cell population is present in both batches
- The batch effect is almost orthogonal to the biological effects
 - Roughly means, batches and biology are expected to have *separate variation*



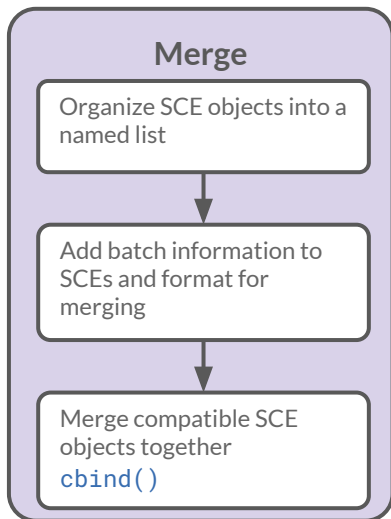
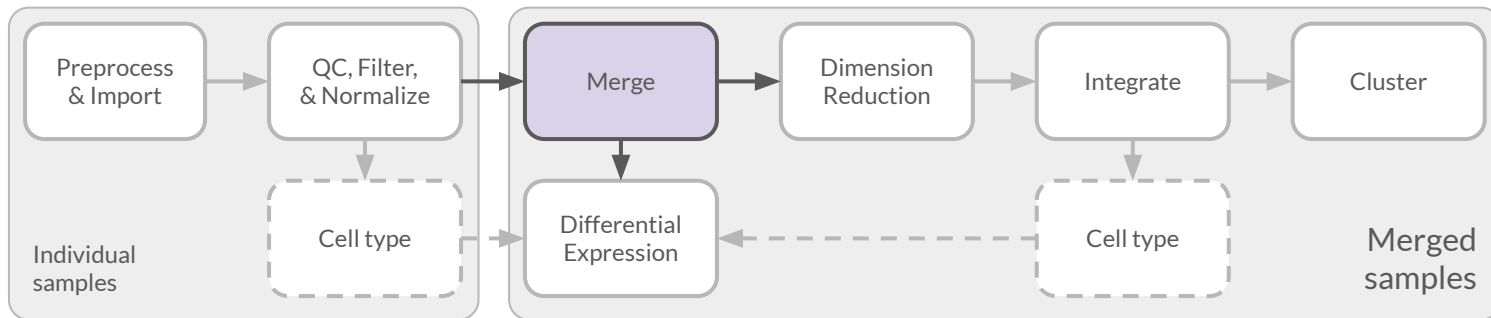
- The batch-effect variation is much smaller than the biological-effect variation across cell types

Harmony batch correction

- "Soft k-means clustering algorithm"



Before we dive in, let's see how some of this sausage gets made

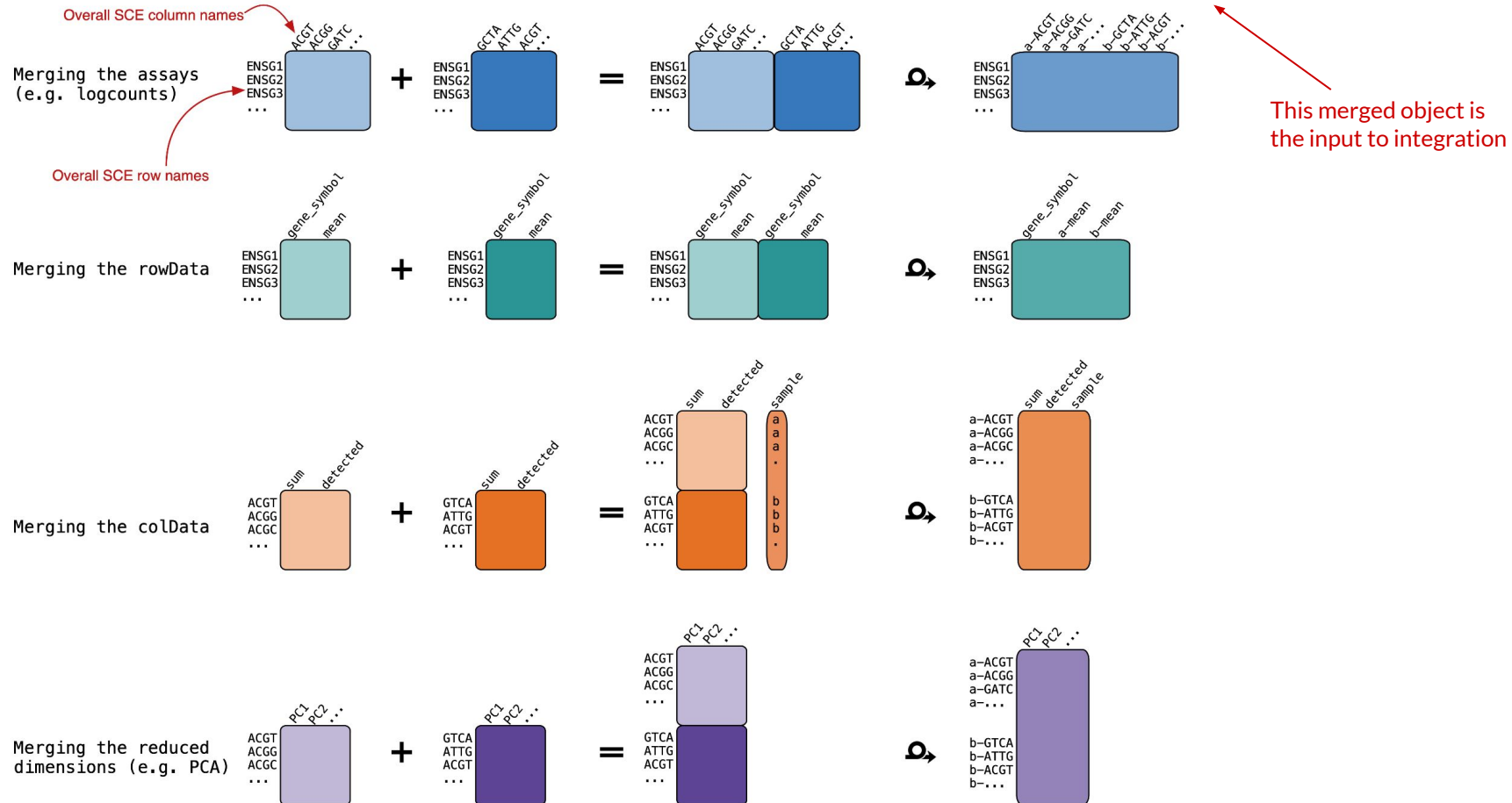


- It's useful to *merge SCEs together* for many downstream analyses, but this merging requires some bookkeeping:
 - After merging, how can we still tell which batch (sample) each cell came from?
 - We need to add this information into SCEs
 - Are SCEs formatted such that R will let us merge them?
 - They need to have compatible column and row names

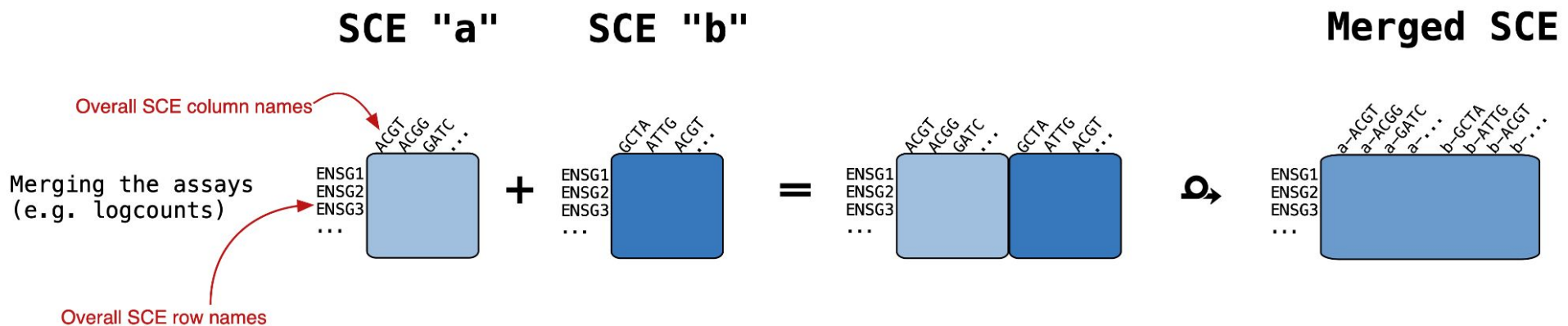
SCE "a"

SCE "b"

Merged SCE

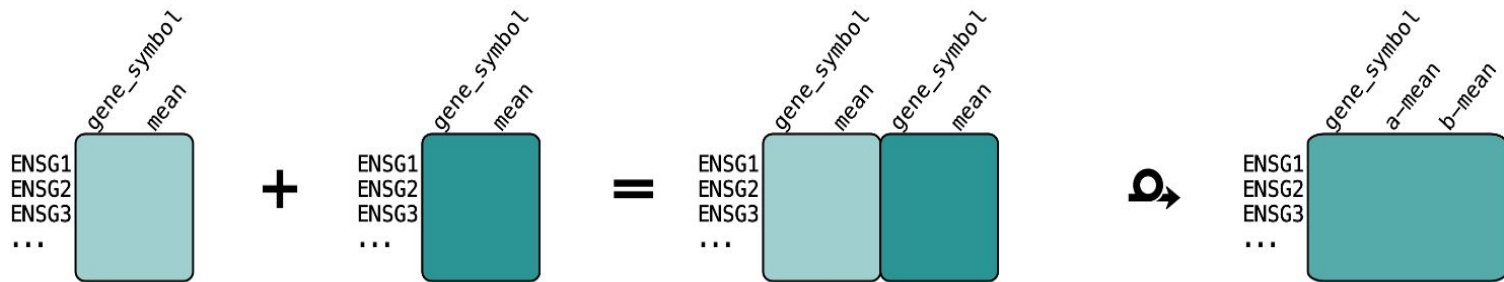


Merging SCE assays



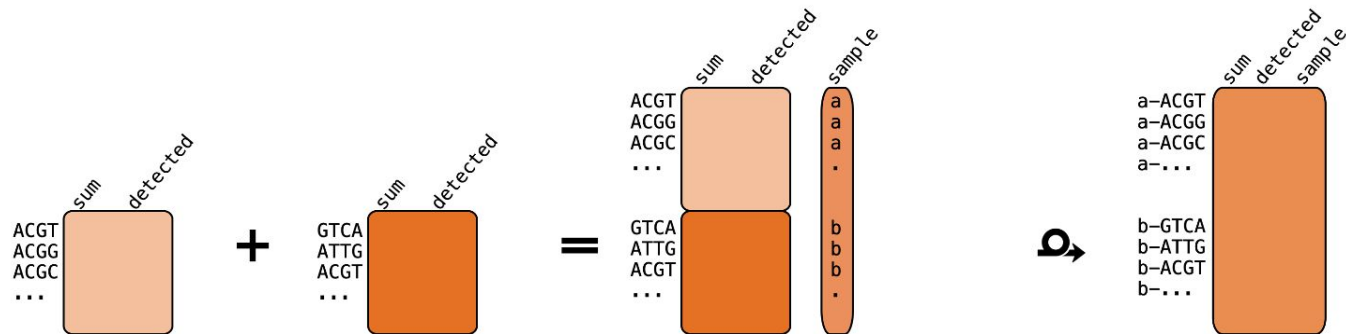
Per-gene (feature) data: Each row is a gene

Merging the rowData

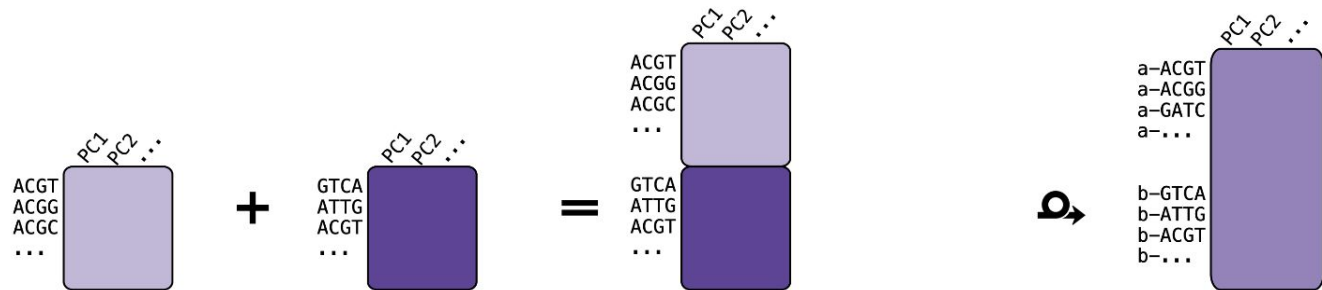


Per-cell data: Each row is a cell

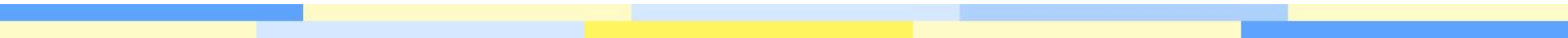
Merging the colData



Merging the reduced dimensions (e.g. PCA)



The following slides show some results we obtained benchmarking integration algorithms.

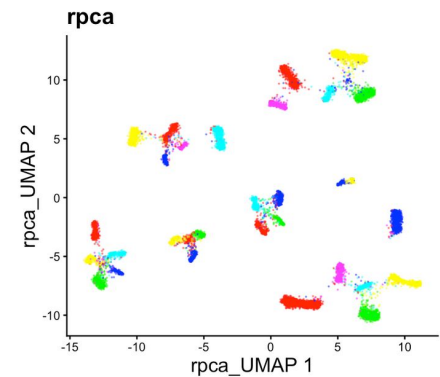
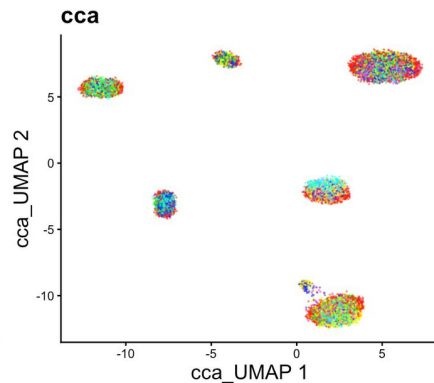
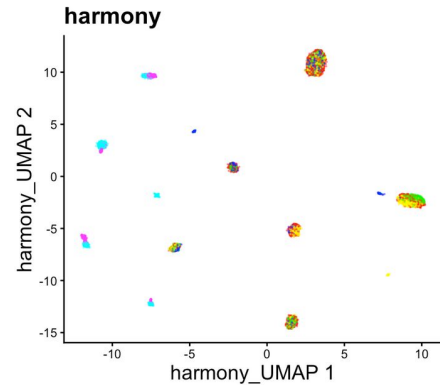
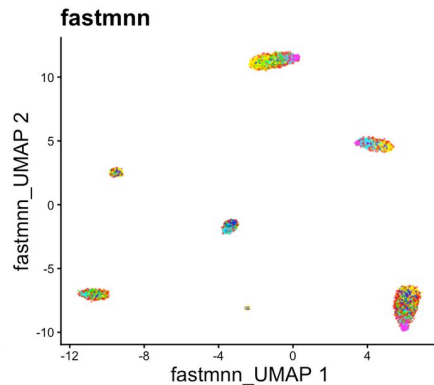
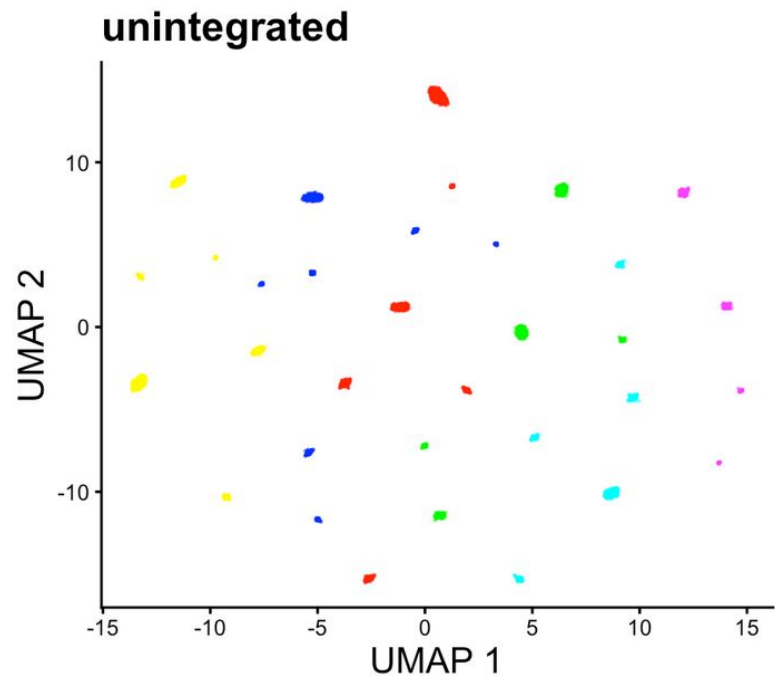


We performed some benchmarking on simulated data from Luecken *et al.*

- We evaluated several methods, four of which we'll show here:
 - **FastMNN**
 - **Harmony**
 - **Seurat using CCA** (canonical correlation analysis)
 - **Seurat using RPCA** (reciprocal PCA)
 - (We'll note that we also looked at scVI, which seemed to work well but is slow on CPU and it's in Python which is beyond the scope of our workshop!)
- We chose these methods based on performance in Luecken *et al.* and their usability
- See <https://github.com/AlexsLemonade/sc-data-integration> for our benchmarking code

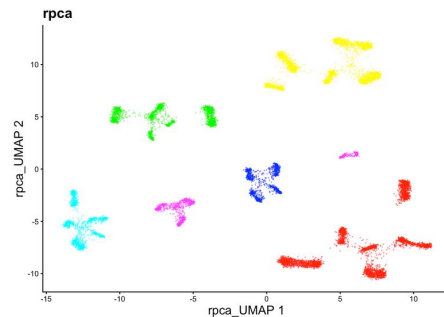
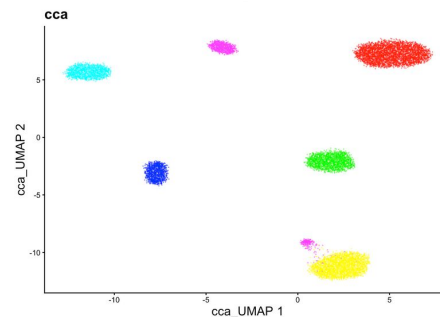
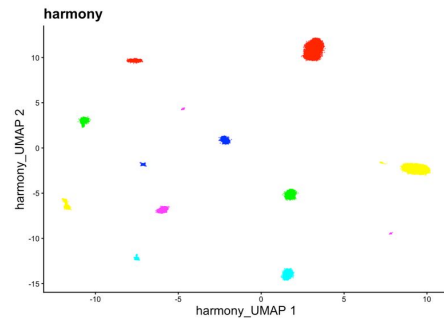
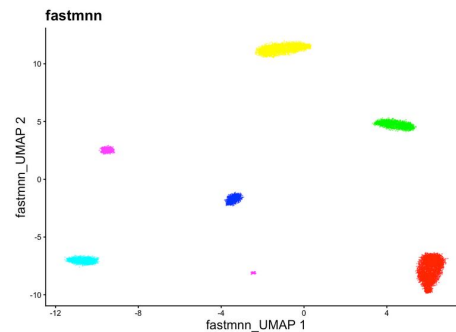
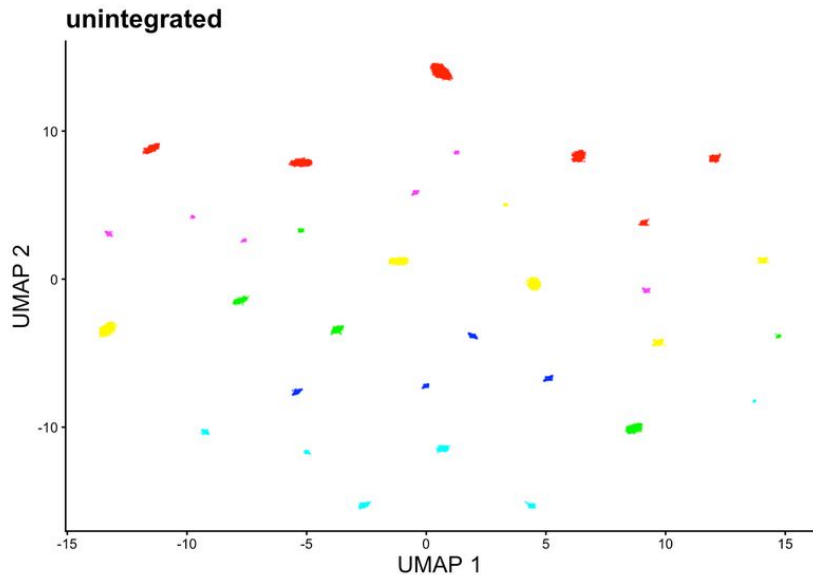
Scenario 1: All cell types are present in all batches

UMAPs colored by Batch



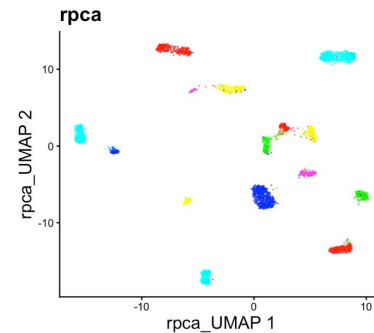
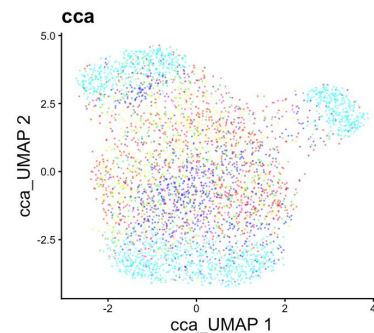
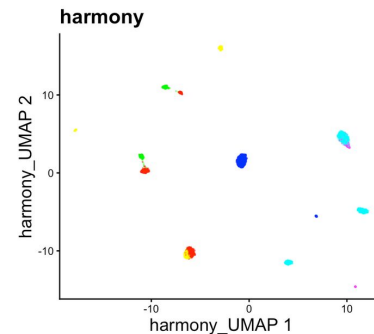
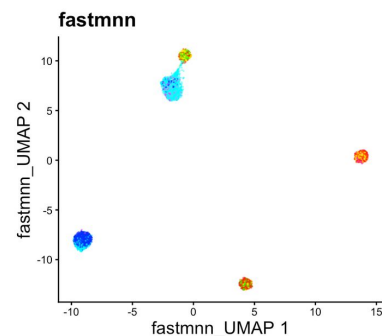
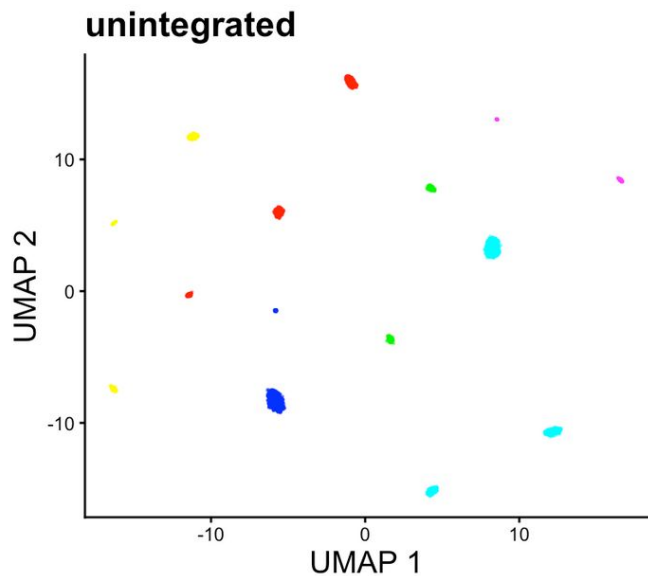
Scenario 1: All cell types are present in all batches

UMAPs colored by Cell Type



Scenario 2: Cell types are not present in all batches, and not all batches have cells in common

UMAPs colored by Batch



Scenario 2: Cell types are not present in all batches, and not all batches have cells in common

UMAPs colored by Cell Type

