# Integrating Different Samples in Single-cell RNA-seq

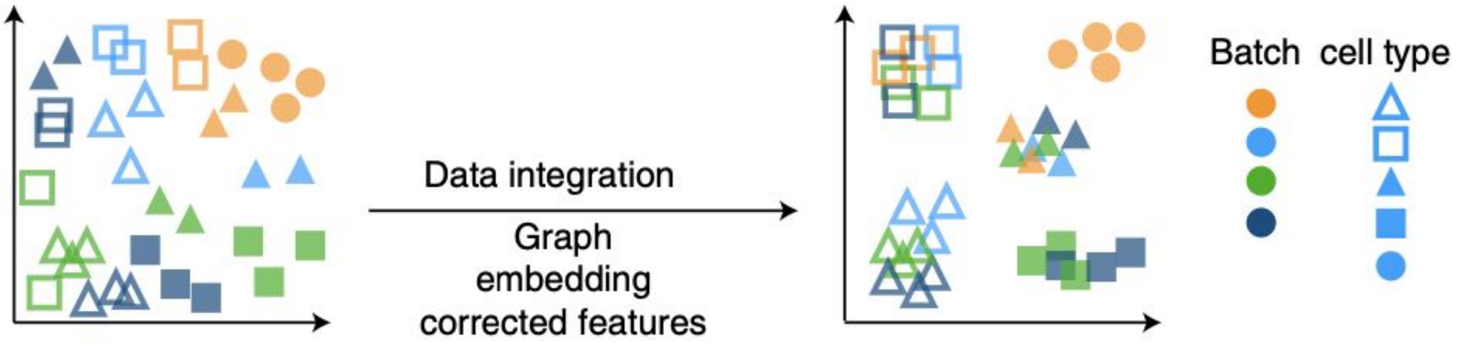
The Data Lab

# Why integrate ("batch correct") datasets?

- Analyze multiple libraries at once rather than analyze each library individually
- Integration allows us to:
    - Perform clustering on a single integrated dataset
    - Visualize integrated libraries in one space
    - Perform downstream analyses that would benefit from a larger sample size:
        - Identify marker genes
        - Cell type annotation
- Provides an integrated profile of multiple libraries, which is often represented as a reduced dimension matrix of some kind (e.g., PCA or "latent embedding")
- Note that we can expect *tradeoffs* between reducing technical variation and retaining biological variation

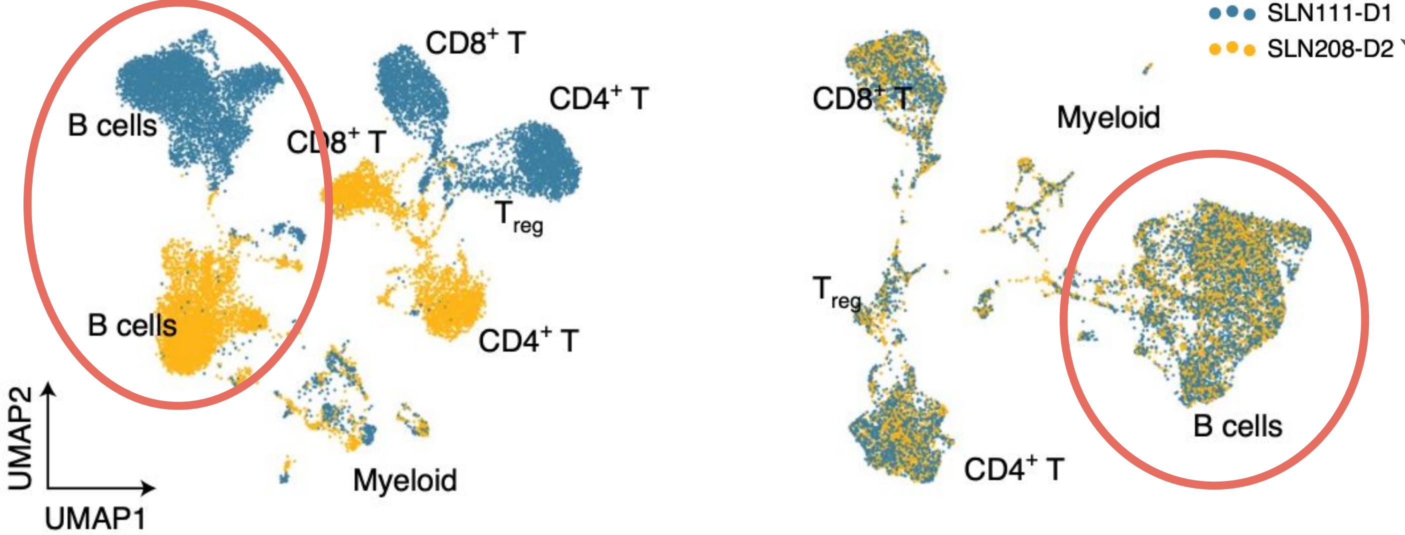
# What does "good integration" look like?

- Helps mitigate ( 🤞 ) the *batch effects* caused by variation across libraries, while hopefully ( 🤞 ) still preserving biological information
  - Technical variation can arise from *all that comes with* separate library preps and sequencing



Adapted from Lueken et al., 2022

# Example of (what looks like*) successful integration



Before integration

After integration

spleen/lymph node samples from two different mice

Gayoso *et al.*, 2021

*Measuring success is actually kind of tricky! Stay tuned...

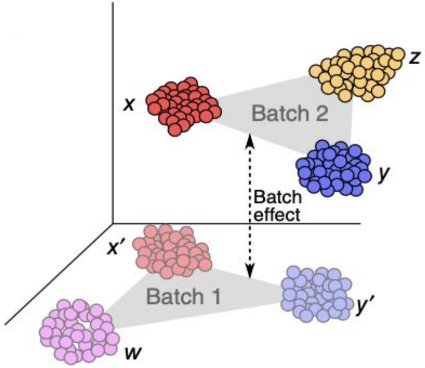
# Let's have a closer look at methods we'll be using

- **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

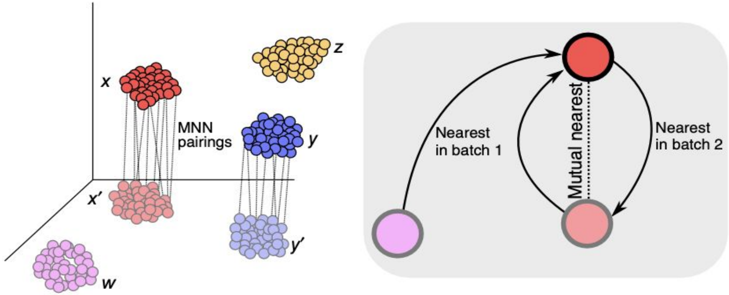
- **Harmony**
  - Korsunsky, I, Millard, N, Fan, J, *et al. Fast, sensitive and accurate integration of single-cell data with Harmony*. (2019) https://doi.org/10.1038/s41592-019-0619-0

# Mutual nearest neighbors batch correction

- Imagine we have 2 batches, each with 3 cell types
  - Red and blue are shared but yellow and pink 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

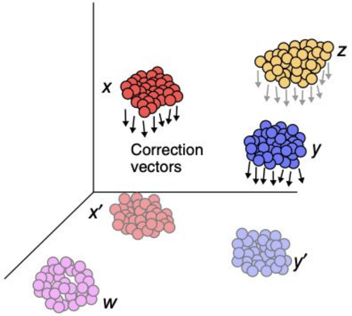


- First, we identify pairs of cells with mutually similar expression profiles
  - These are our "mutual nearest neighbors"

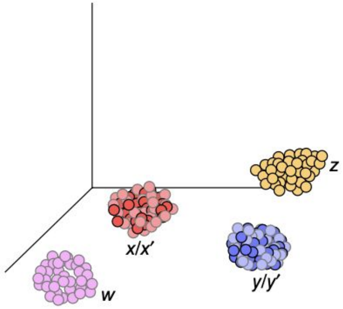


# Mutual nearest neighbors batch correction

- Next, compute a batch correction vector for each MNN pair



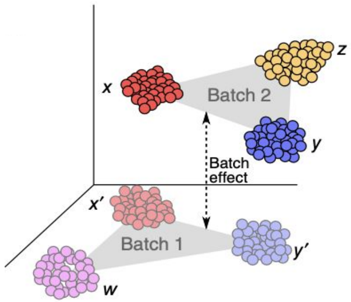
- Finally, calculate the weighted average of these vectors to get cell-specific batch corrections to perform the final integration
  - Note that **w** and **z** don't "look" as "integrated"! Why?



# Some assumptions that MNN makes

- At least one cell population is present in both batches

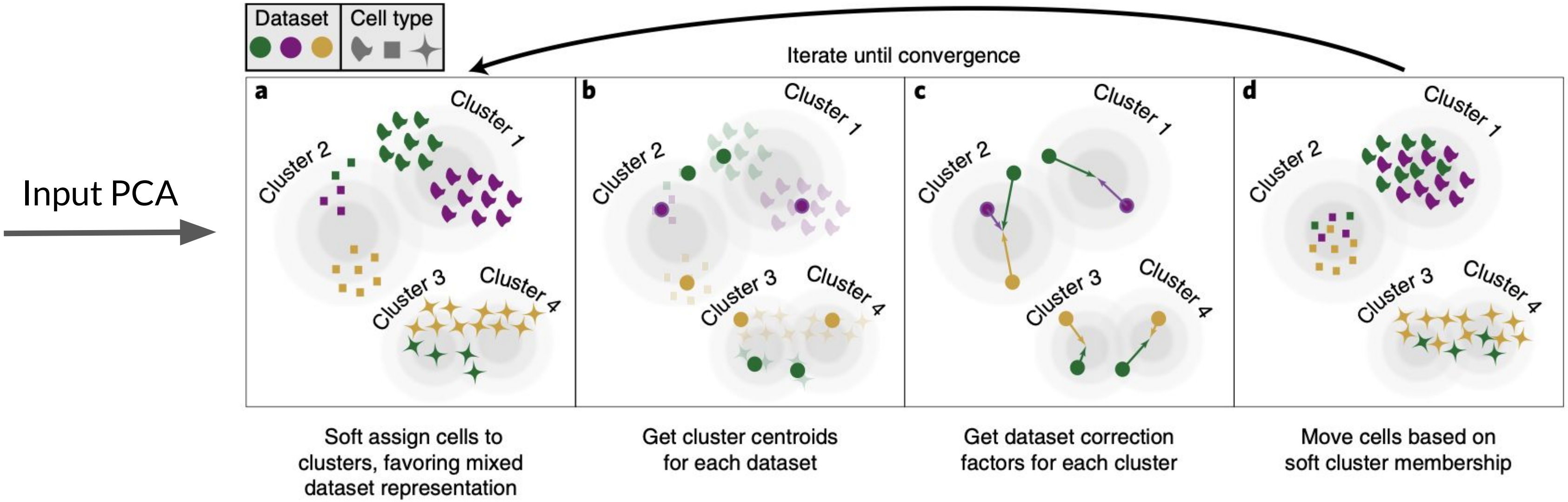
- The batch effect is almost orthogonal to the biological effects
  - Roughly means, batches and biology are expected to have *separate variation*



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

# Harmony batch correction

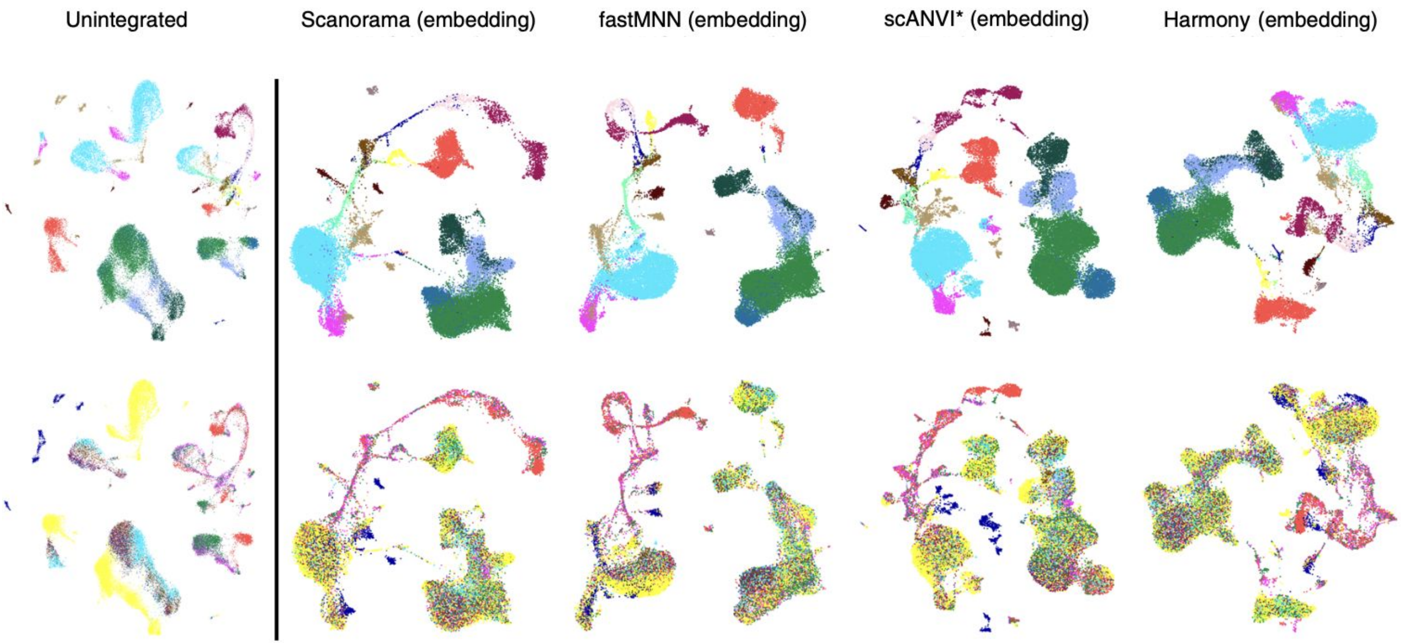
- "Soft k-means clustering algorithm"


Input PCA
Dataset
Cell type
Iterate until convergence
a
Cluster 1
Cluster 2
Cluster 3
Cluster 4
b
Cluster 1
Cluster 2
Cluster 3
Cluster 4
c
Cluster 1
Cluster 2
Cluster 3
Cluster 4
d
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Soft assign cells to clusters, favoring mixed dataset representation
Get cluster centroids for each dataset
Get dataset correction factors for each cluster
Move cells based on soft cluster membership

# Evaluating integration: What counts as "good"?

- Before and after UMAP vibes
    - We expect that batches cluster less after integration due to removed technical variation
    - But "biology" across batches should still cluster together
        - Biology = cell types (which may not be known!), tissue type, donor, etc.

# Your mileage may vary across methods!



Batches

Cell types

Luecken, M.D., Büttner, M., Chaichoompu, K. et al. *Benchmarking atlas-level data integration in single-cell genomics.* (2022). https://doi.org/10.1038/s41592-019-0619-0

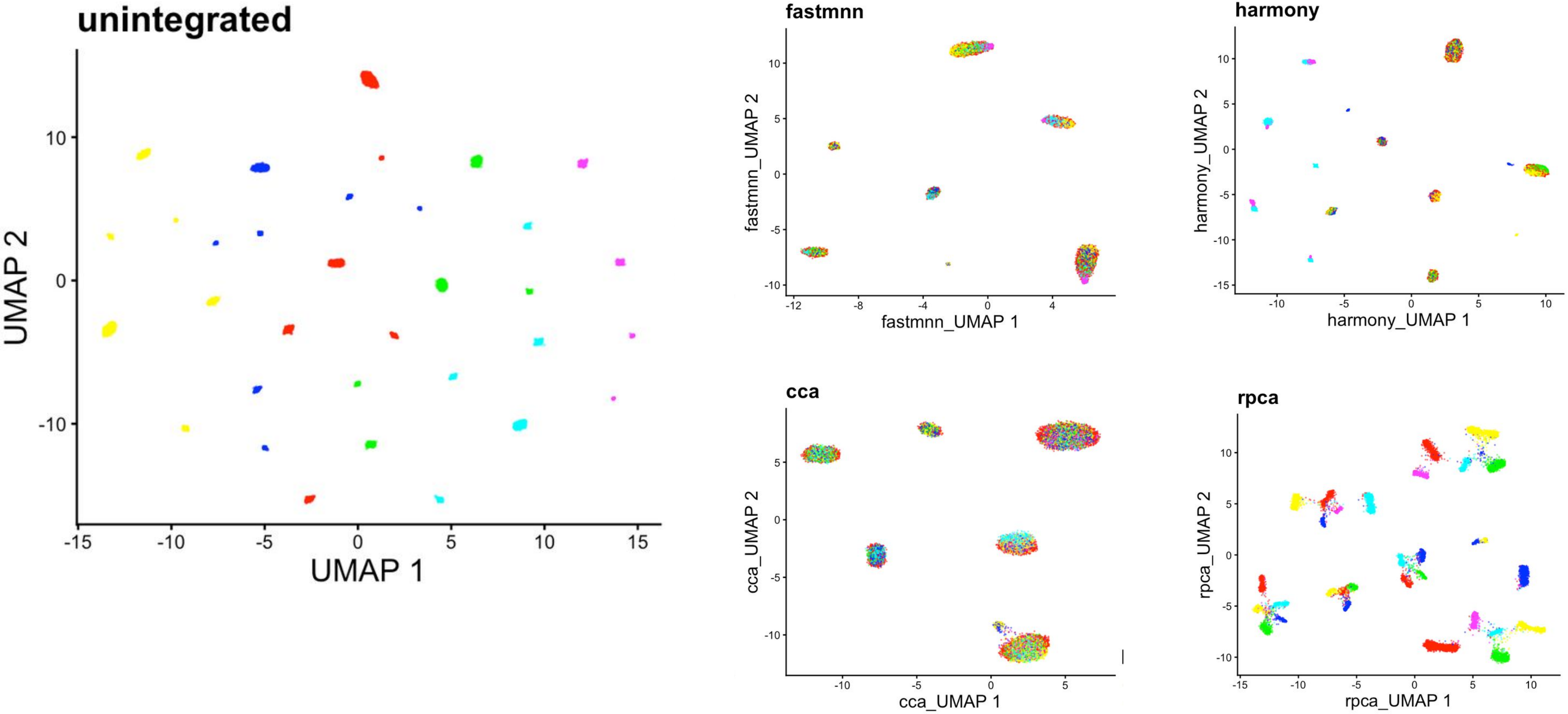
# We performed some benchmarking on simulated data from Luecken *et al.*

- 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

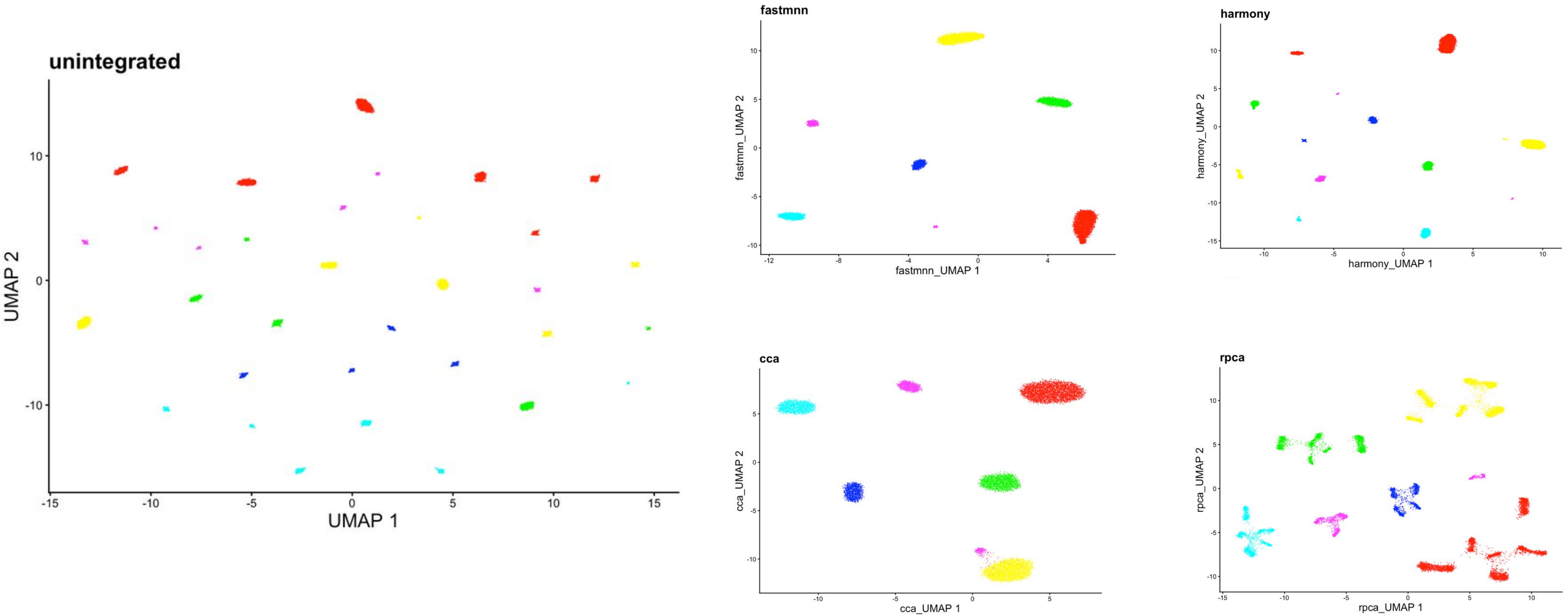
# Scenario 1: All cell types are present in all batches

**UMAPs colored by Batch**



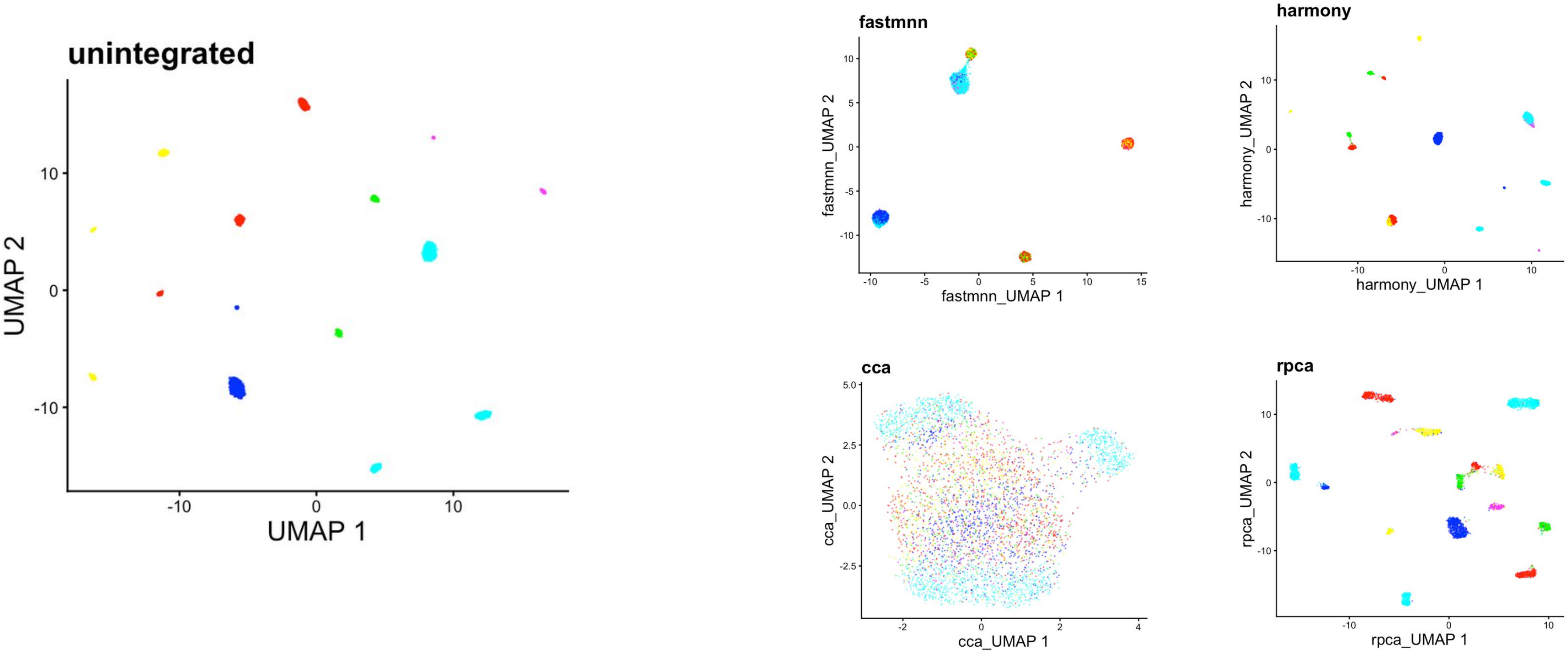
# Scenario 1: All cell types are present in all batches

**UMAPs colored by Cell Type**



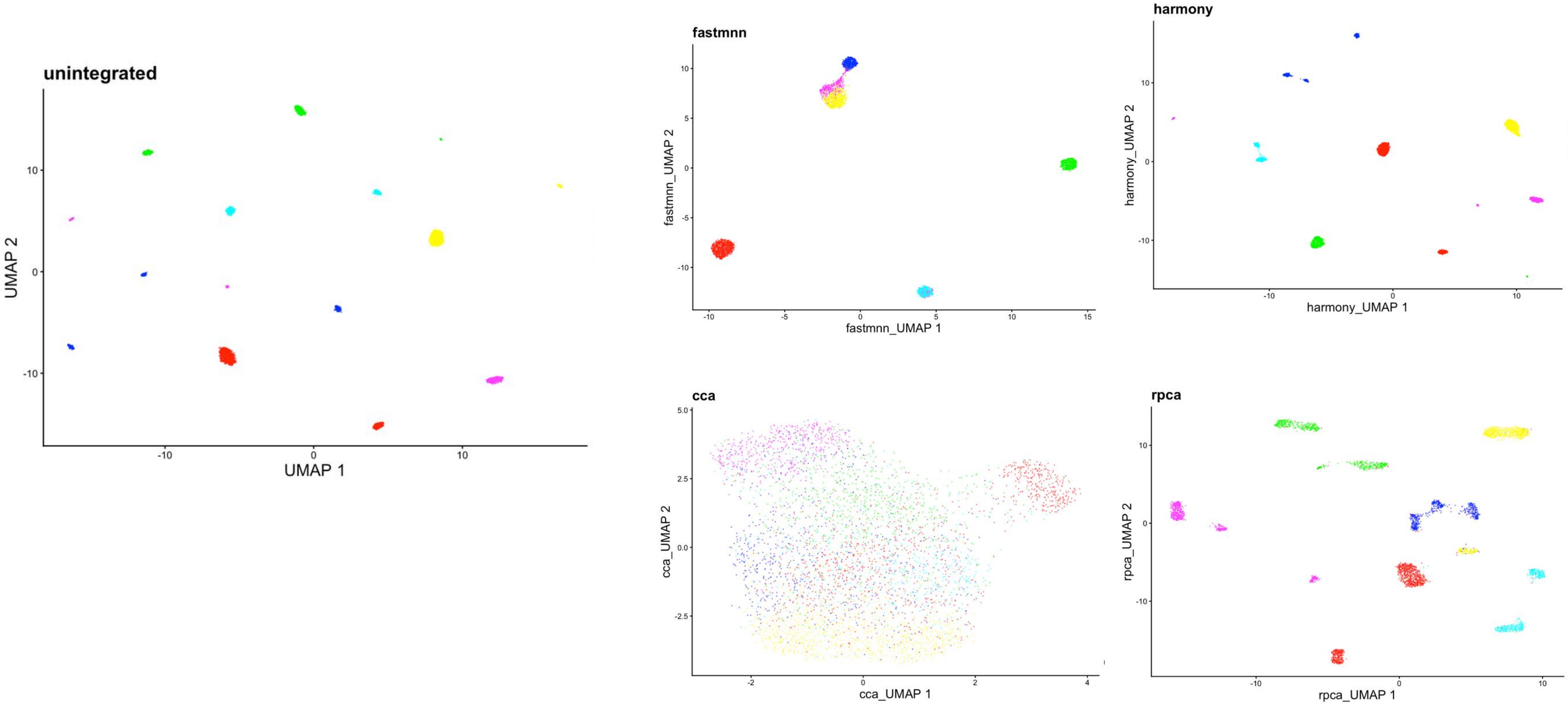
# Scenario 2: Cell types are not present in all batches, and not all batches have cells in common

**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**



# Evaluating integration: What counts as "good"?

- Before and after UMAP vibes

  - We expect that batches cluster less after integration due to removed technical variation
  - But "biology" across batches should still cluster together
    - Biology = cell types (which may not be known!), tissue type, donor, etc.

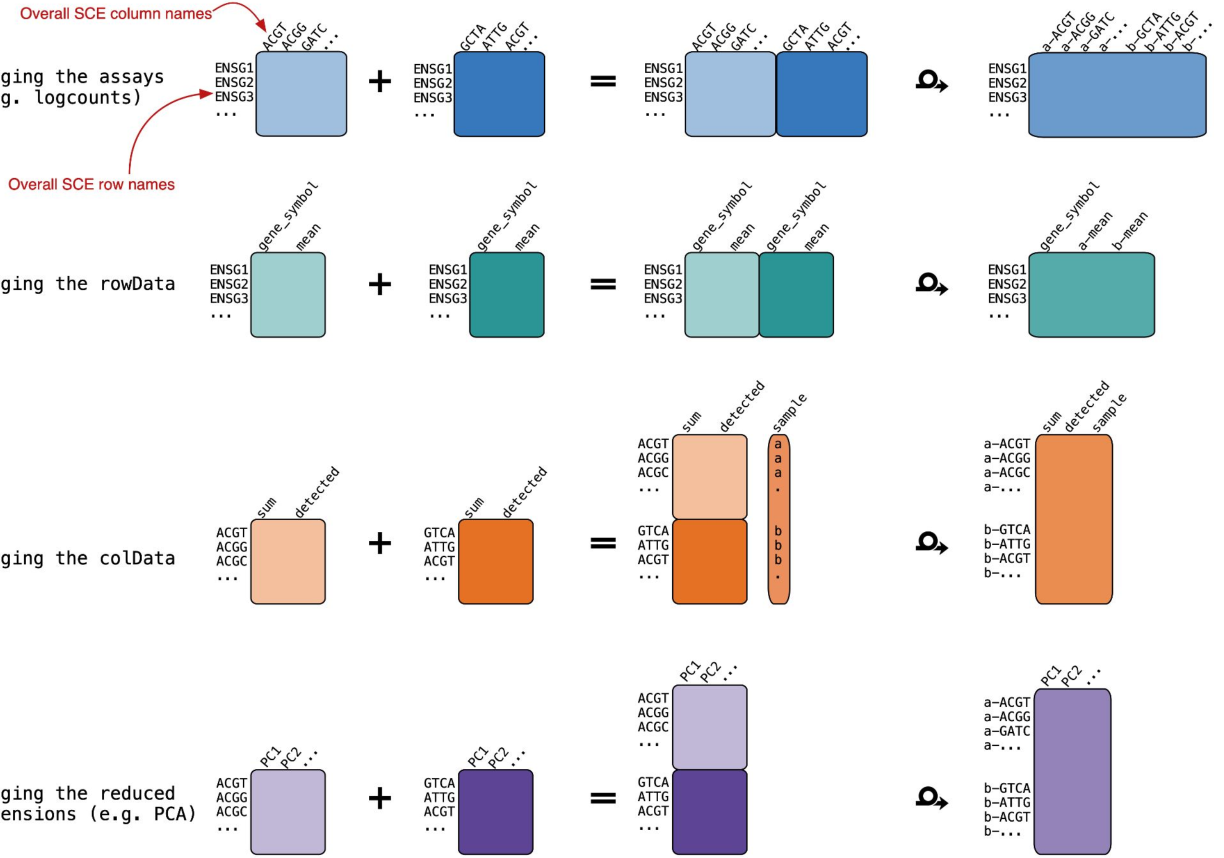
- There are several metrics for evaluating batch correction
  - Luecken *et al.* (2022) is an excellent reference https://doi.org/10.1038/s41592-019-0619-0
  - Caution: Metrics generally do not measure "was integration successful," but other proxies which *sometimes can help us tell* if integration was successful (or at least not unsuccessful)

# Performing integration: Bookkeeping

- As input, many methods (in R!) require you to merge all SCEs into one *unintegrated SCE object*, which can then be integrated
    - *Key point: Combining is NOT integrating*

- This means SCEs need to be able to be merged, which may require us to manipulate SCE objects first for compatibility!

SCE "a" SCE "b" Merged SCE



Overall SCE column names

Overall SCE row names

Merging the assays (e.g. logcounts)

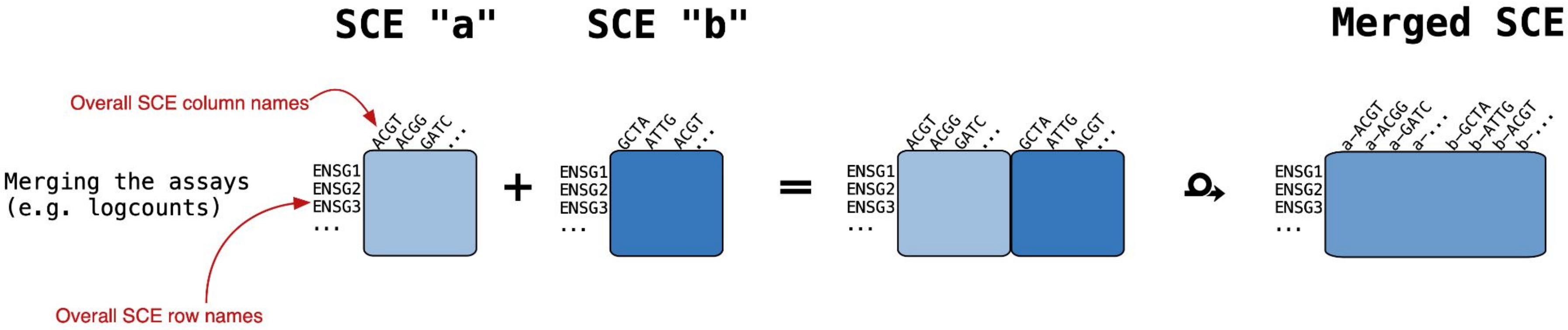
Merging the rowData

Merging the colData

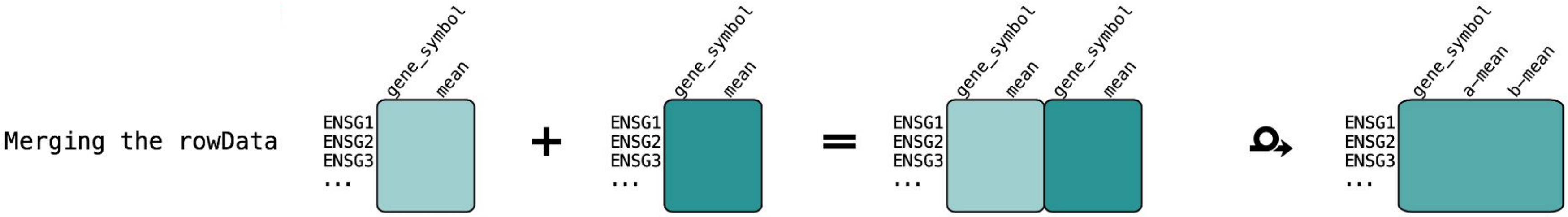
Merging the reduced dimensions (e.g. PCA)

This merged object is the input to integration

# Merging SCE assays

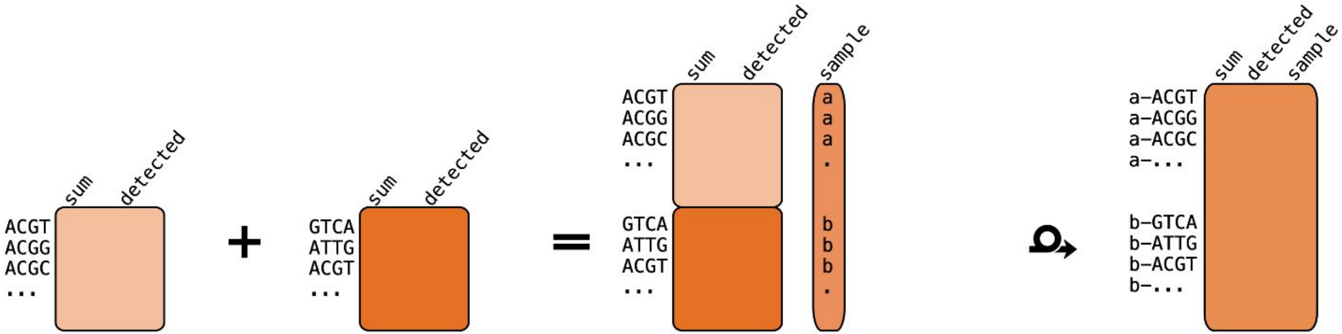


# Per-gene (feature) data: Each row is a gene

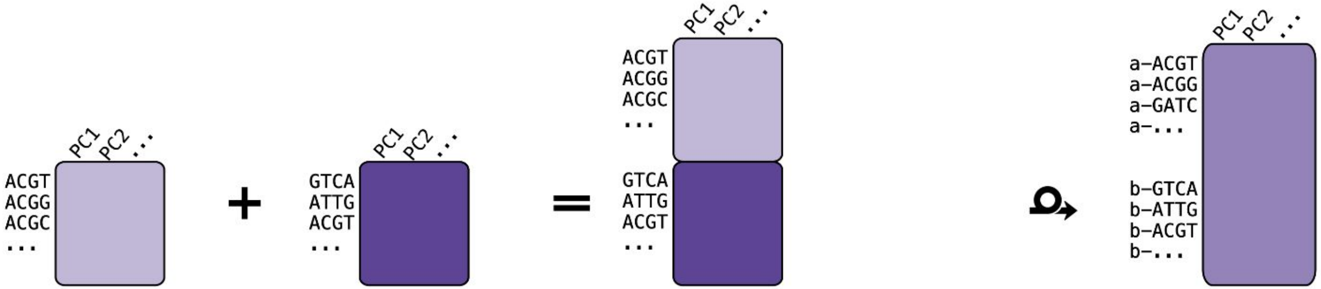


# Per-cell data: Each row is a cell

Merging the colData



Merging the reduced dimensions (e.g. PCA)


PC1
PC2
...
ACGT
ACGG
ACGC
...
+
PC1
PC2
...
GTCA
ATTG
ACGT
...
=
PC1
PC2
...
ACGT
ACGG
ACGC
...
GTCA
ATTG
ACGT
...
PC1
PC2
...
a-ACGT
a-ACGG
a-GATC
a-...
b-GTCA
b-ATTG
b-ACGT
b-...