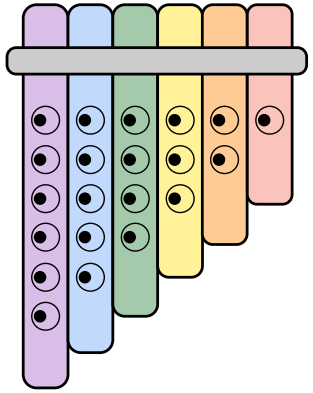
Panpipes



Panpipes

Automating Analysis of Multimodal Single-cell Datasets

Charlotte Rich-Griffin

2023-03-06

Download test data

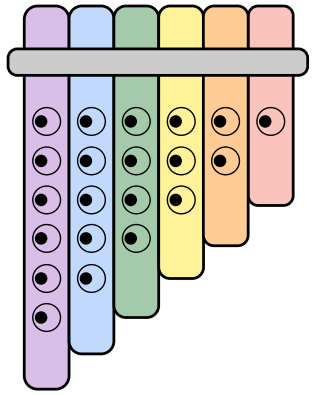
Upload the test data to bmrc using rsync or Filezilla

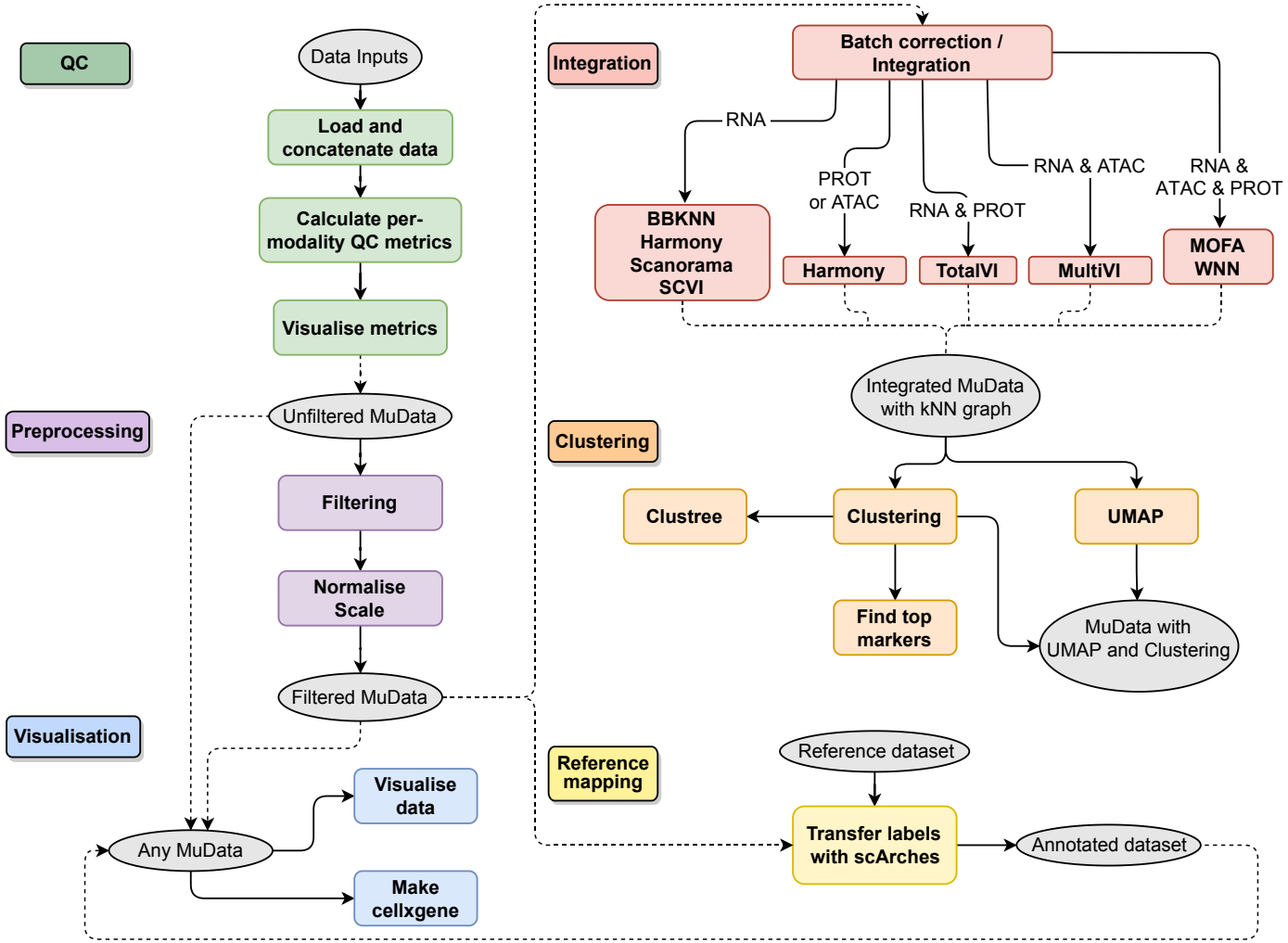
Let's take a quick tour of Panpipes:

Panpipes: Automating Analysis of Multimodal Single-cell Datasets

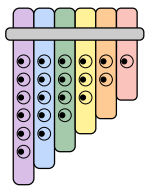
janssen Immunology | UNIVERSITY OF OXFORD

Panpipes





Panpipes

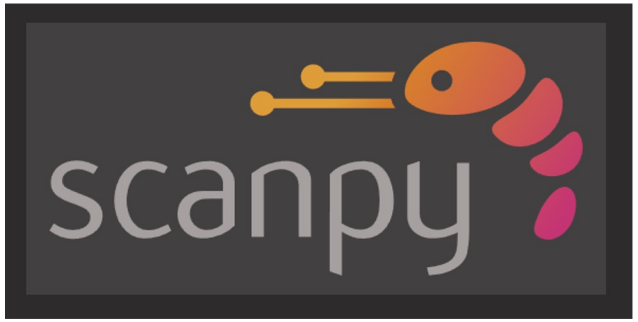


KEY

- Data structure
- Analysis step
- All modalities
- *xxx* Modality-specific step
- All modalities (between workflows)

Frameworks

- CGAT-core/ruffus framework for workflow management



- Python based ecosystem of single cell tools
- Scanpy is scalable to 1,000,000s of cells.

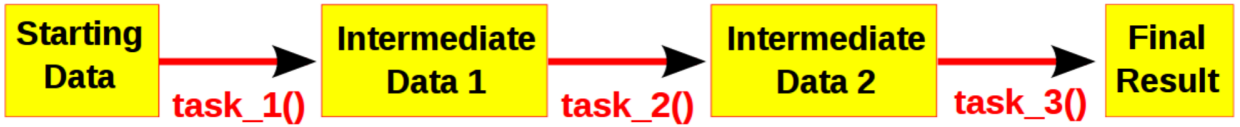
- Muon is a multi-modal extension of scanpy

- Scirpy – AnnData based repertoire analyses

Ruffus

- Open-source workflow library for Python
- **Lightweight**: Suitable for the simplest of tasks
- **Scalable**: Handles even fiendishly complicated pipelines
- **Standard python**: Uses decorators to wrap standard functions

- Flow control from one function to the next
- Allows tasks to be completed sequentially
- If the pipeline fails, ruffus will restart where it left off



http://www.ruffus.org.uk/

CGATcore

- Flexible parameterisation
- Detailed logging
- Database interaction
- Interaction with HPC clusters
- Simplified pipeline control

Thanks to David Sims for this slide

How does it actually work?

1. Configuration

   `panpipes <workflow> config`

   This creates a file called “pipeline.yml” which you editfor your project

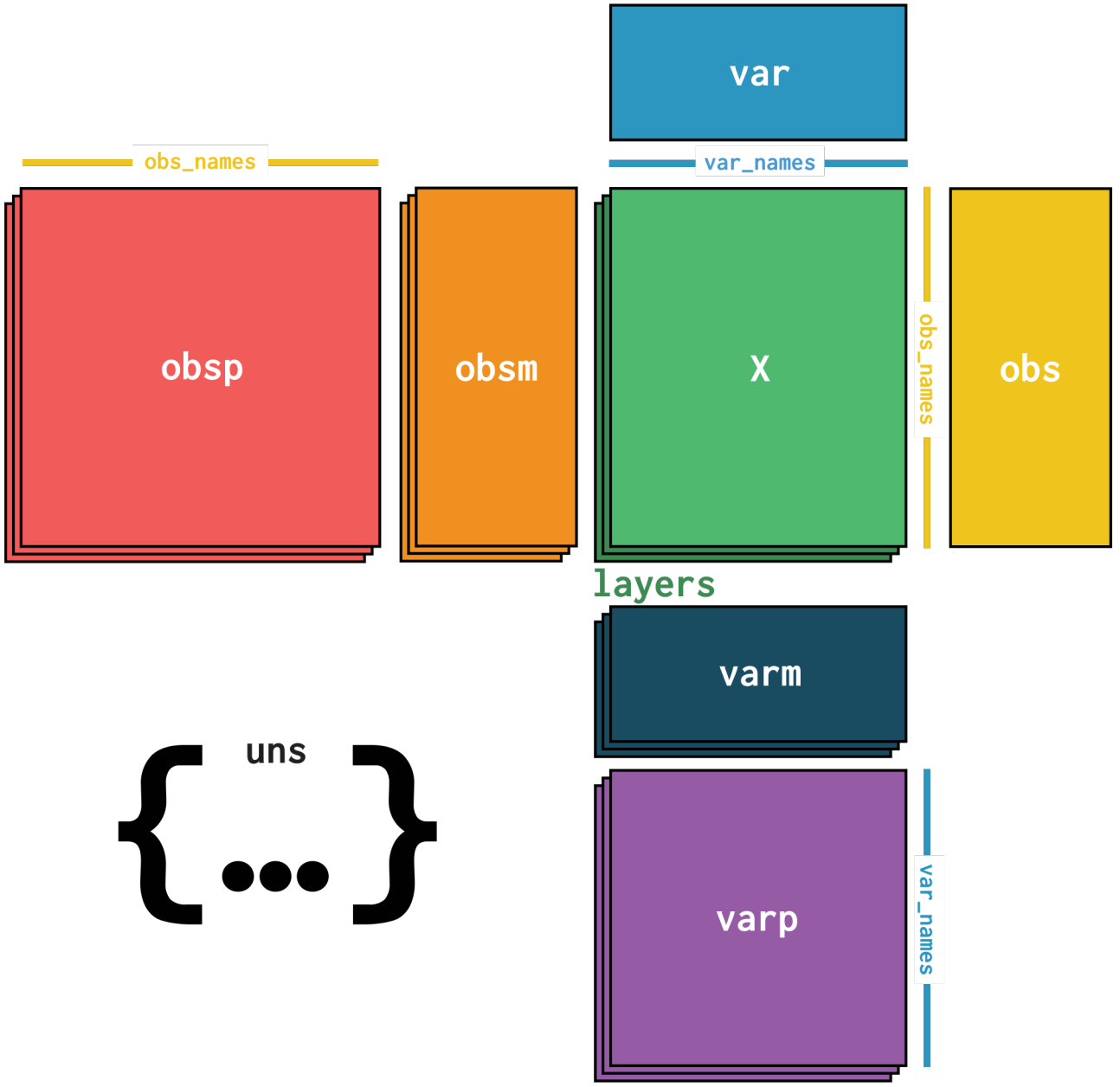
   Make sure you have all the required input files in the places that you have specified.

2. Review what steps will run: `panpipes <workflow> show full`

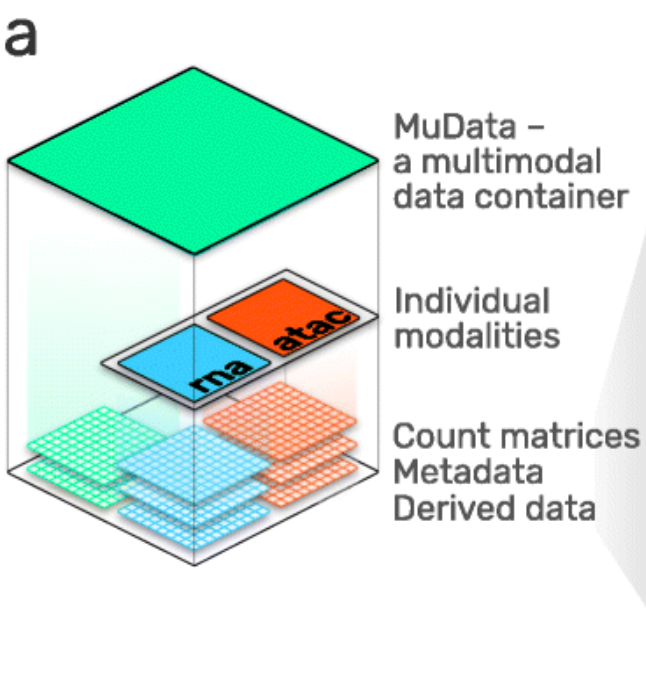
3. Run the pipeline: `panpipes <workflow> make full`

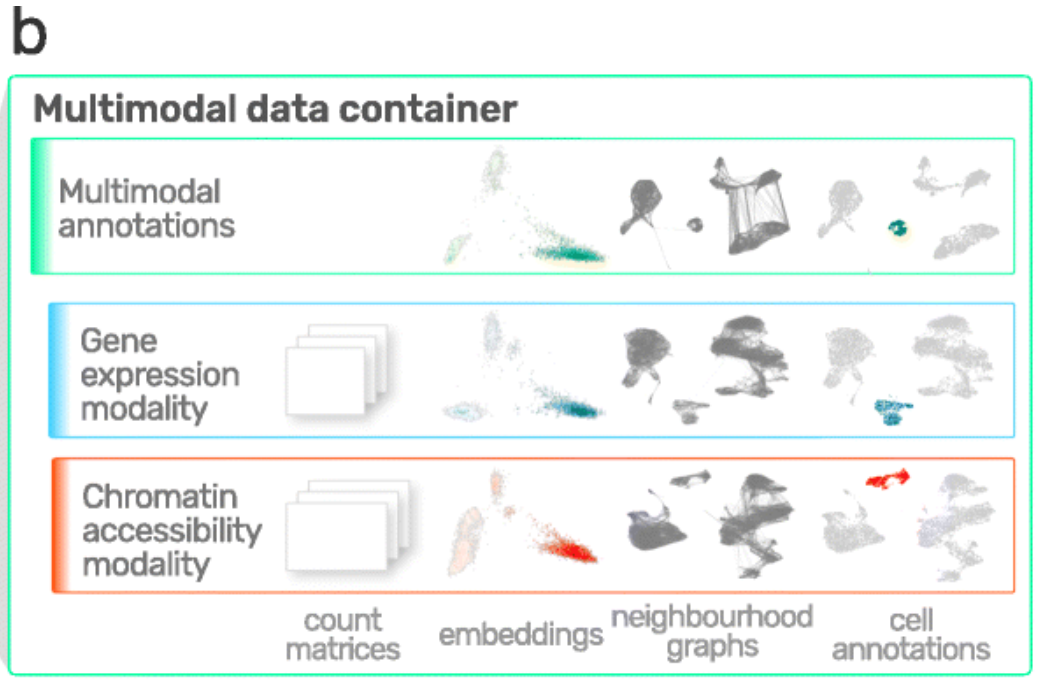
   Panpipes will submit all the jobs to the computation nodes

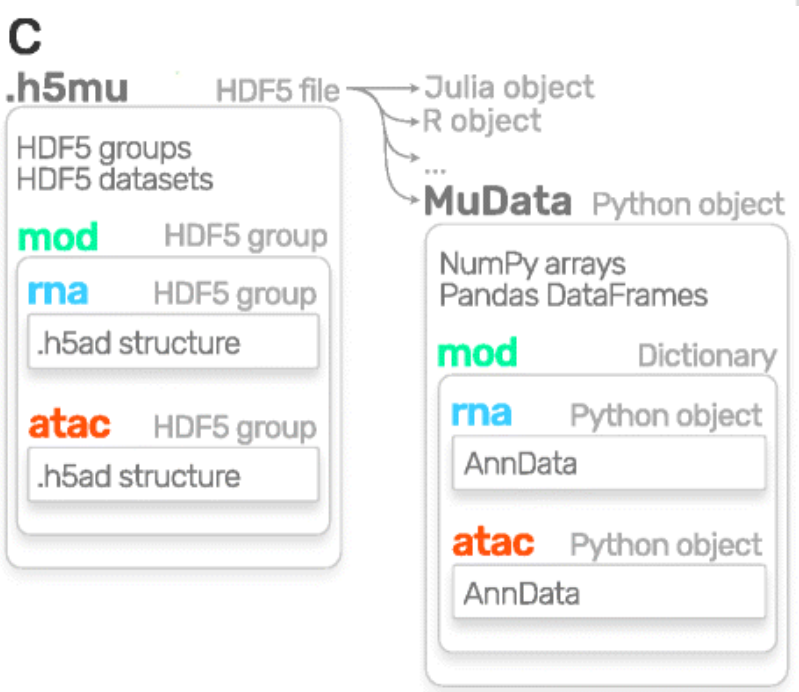
Anndata:


var
obs_names
var_names
obsp
obsm
X
obs_names
obs
layers
varm
uns
{...}
varp
var_names

MuData


a
rna
atac
MuData –
a multimodal
data container
Individual
modalities
Count matrices
Metadata
Derived data

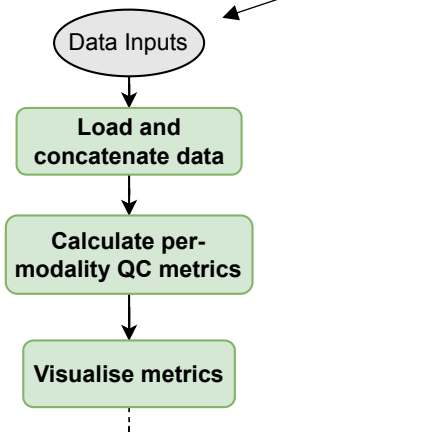

b
Multimodal data container
Multimodal
annotations
Gene
expression
modality
Chromatin
accessibility
modality
count
matrices
embeddings
neighbourhood
graphs
cell
annotations

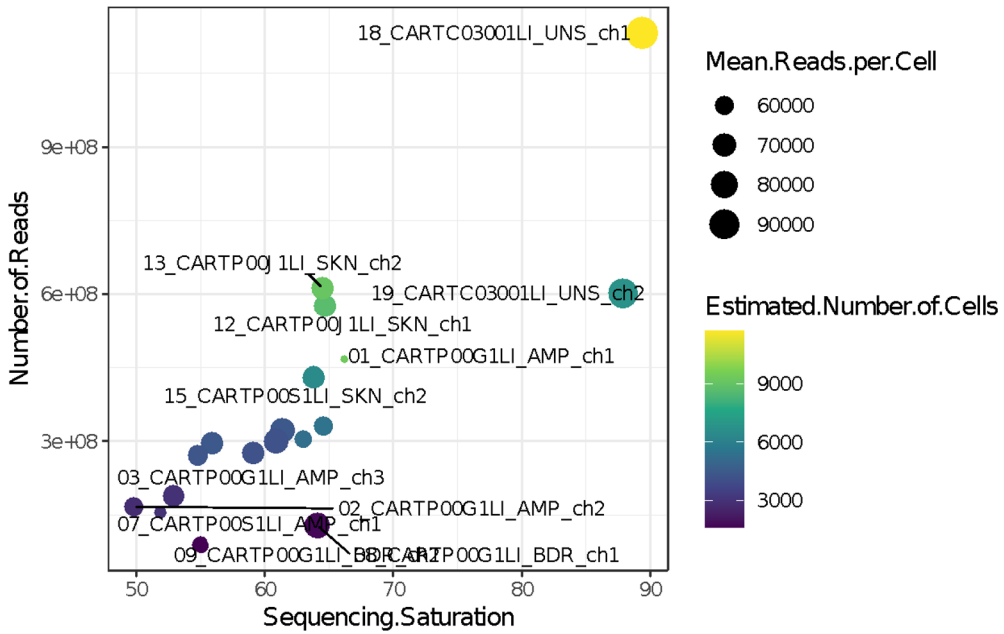

c
.h5mu
HDF5 file
Julia object
R object
...
MuData
Python object
HDF5 groups
HDF5 datasets
mod
HDF5 group
rna
HDF5 group
.h5ad structure
atac
HDF5 group
.h5ad structure
NumPy arrays
Pandas DataFrames
mod
Dictionary
rna
Python object
AnnData
atac
Python object
AnnData

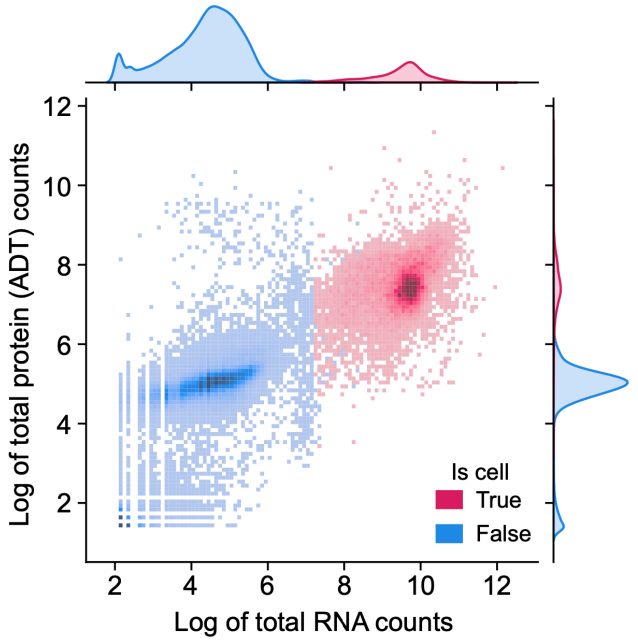
QC

Panpipes can import and process any combination:

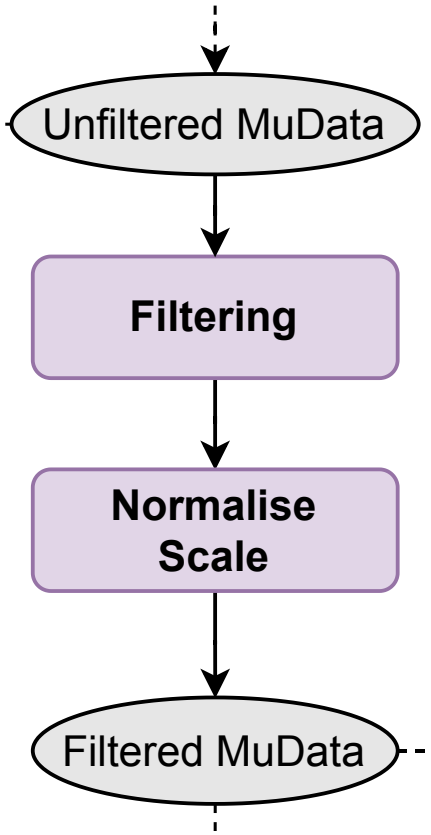
- scRNAseq
- CITEseq
- REPseq (BCR and TCR)
- scATACseq (multiome)



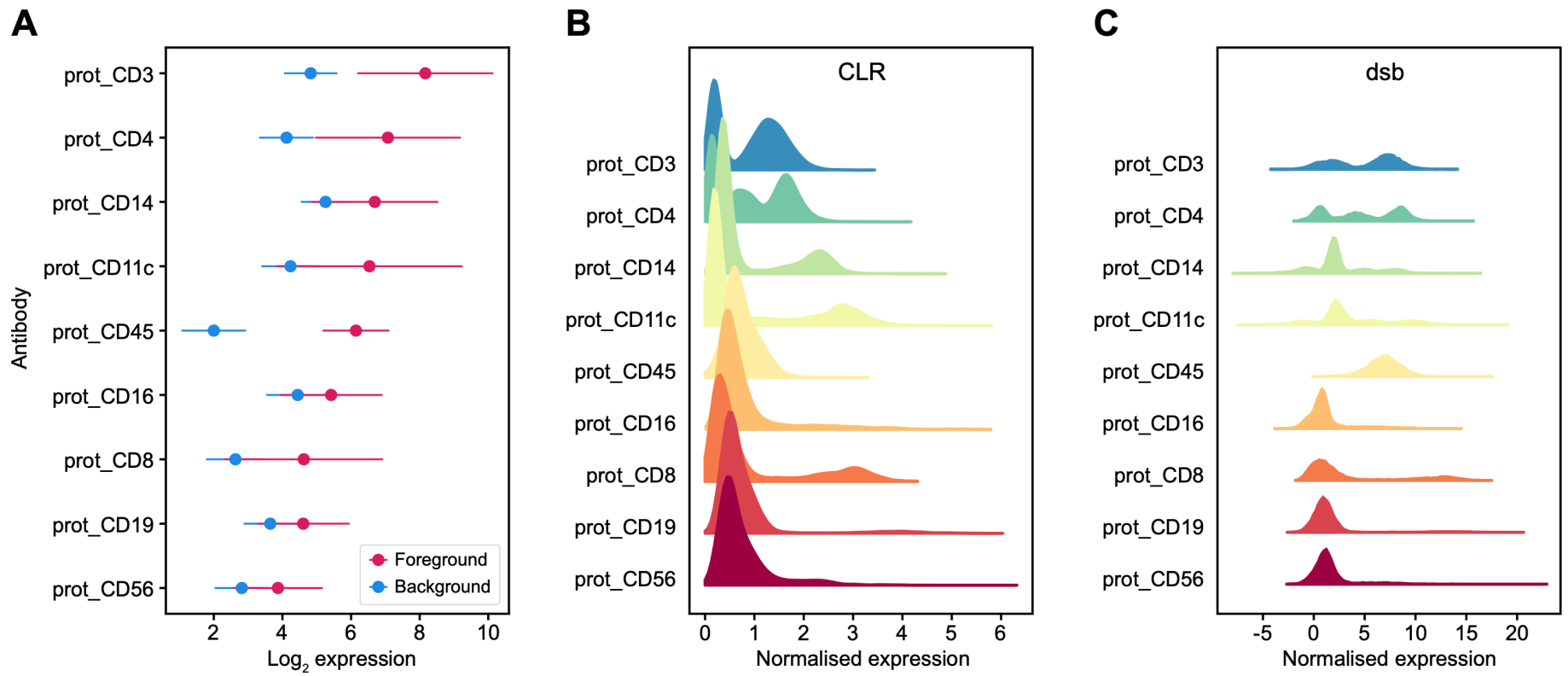




Preprocessing

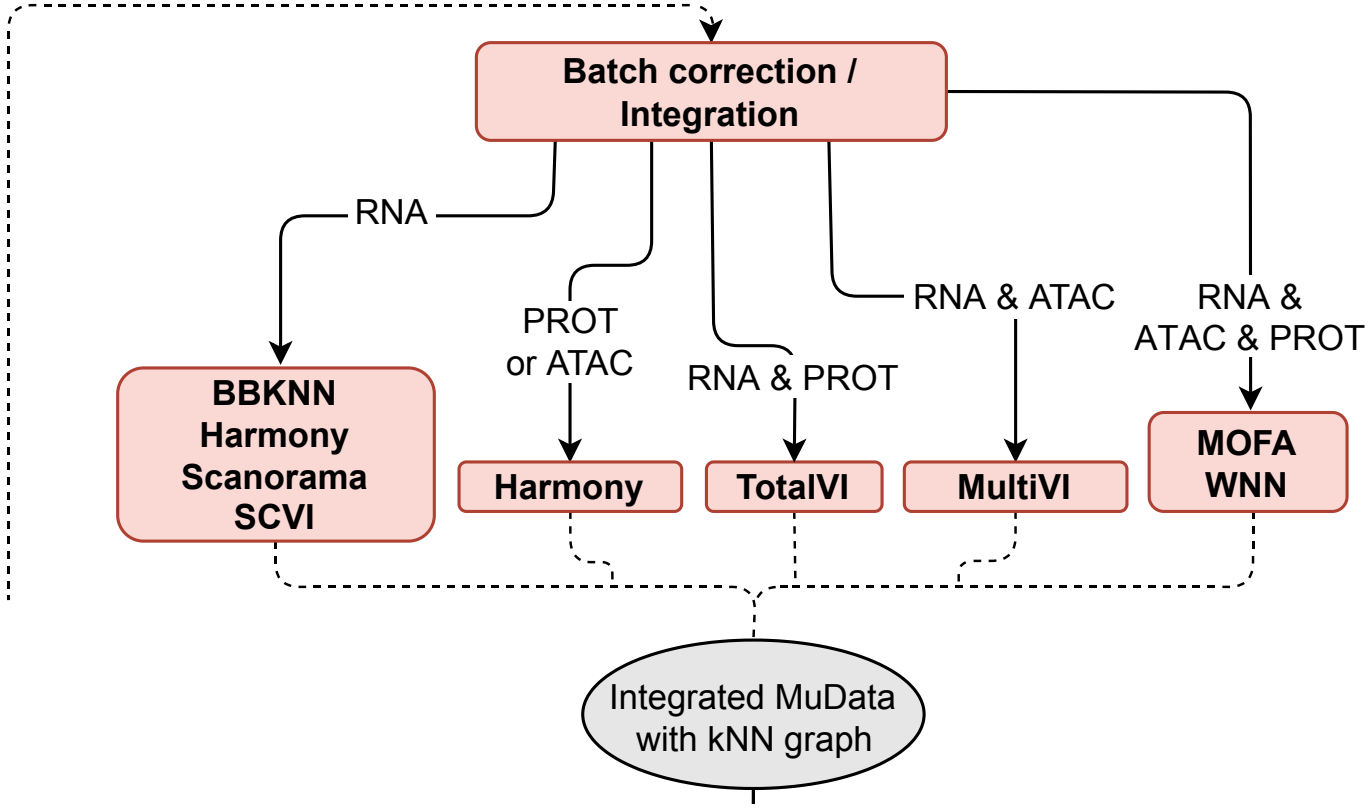


- Dynamic filtering
- Appropriate processing of all modalities

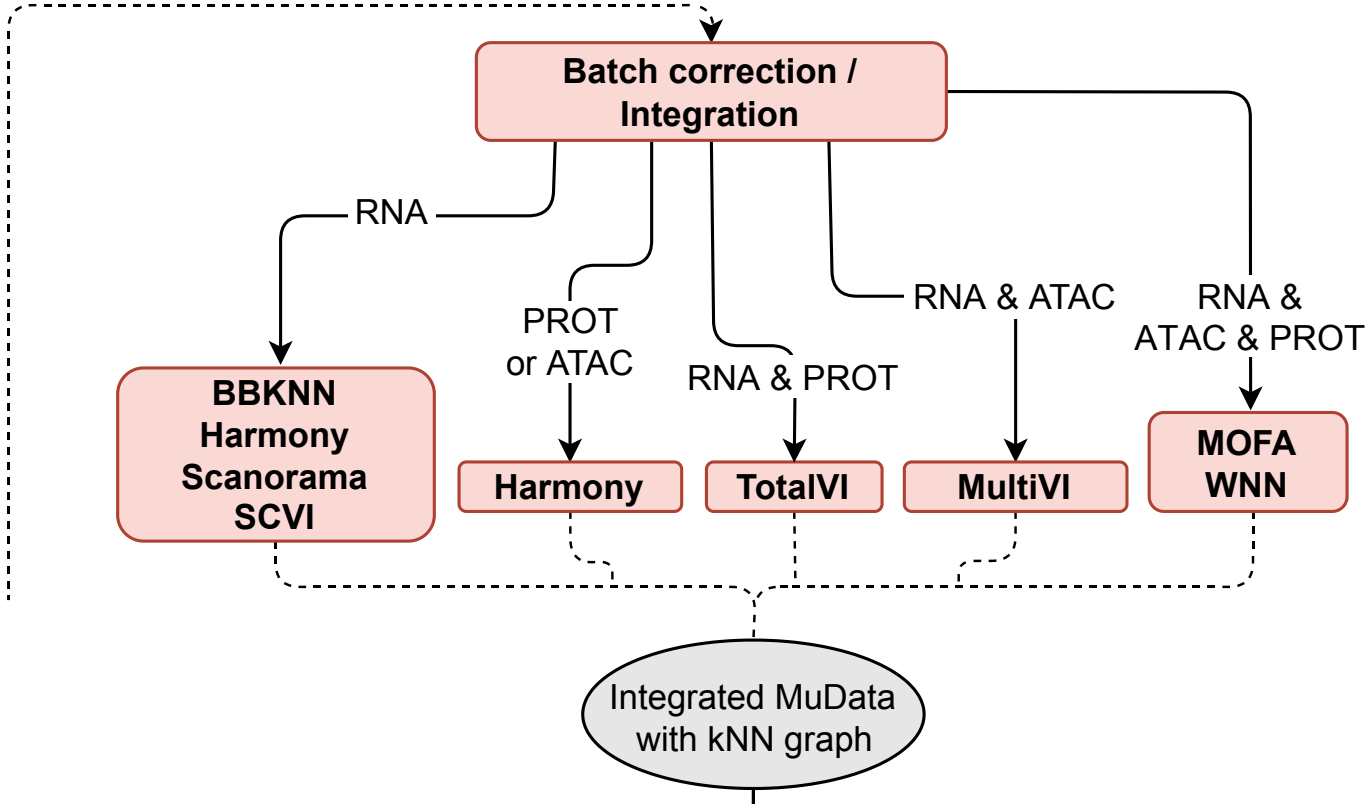


- Comparing the counts in cell-containing foreground to empty droplets in the background
- Panpipes provides 2 options for ADT normalization.

Integration



Integration



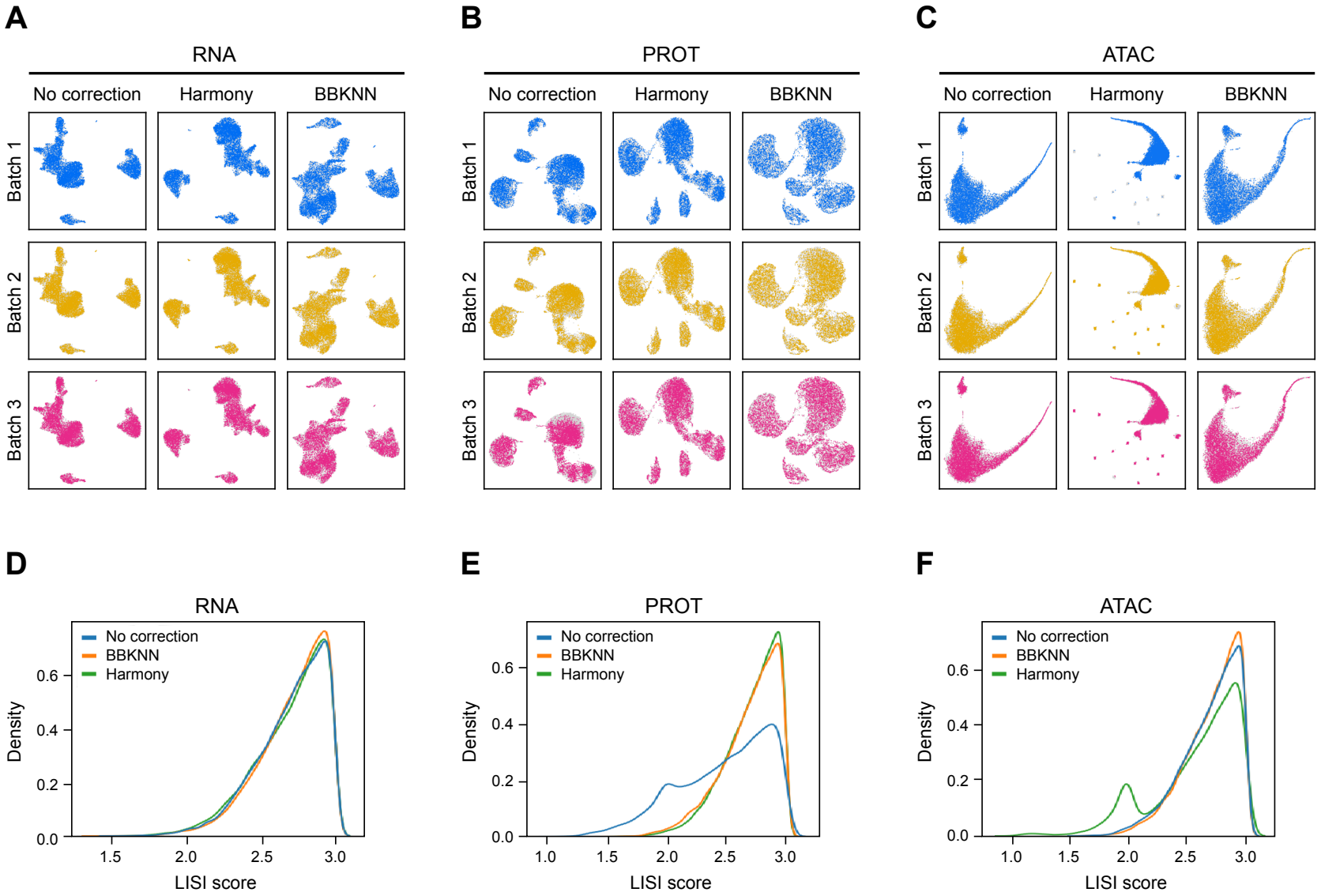
Graph based methods:
- BBKNN
- WNN

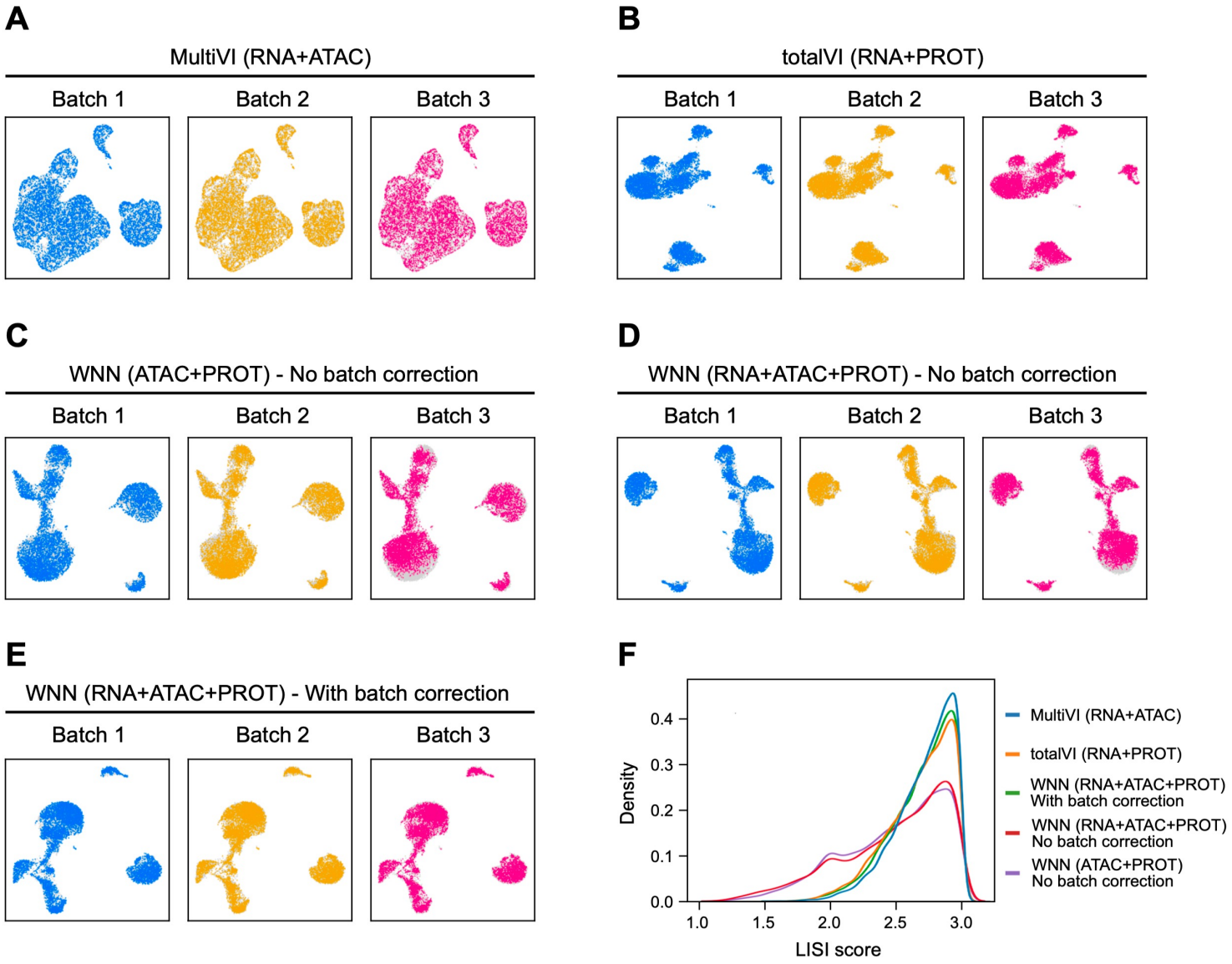
Non-linear reduction based methods:
- Harmony
- Scanorama

Machine Learning methods:
- scVI
- TotalVI
- MultiVI

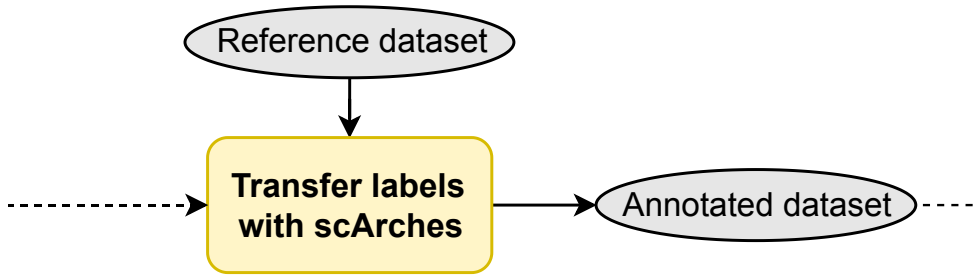
Factor based method:
- MOFA

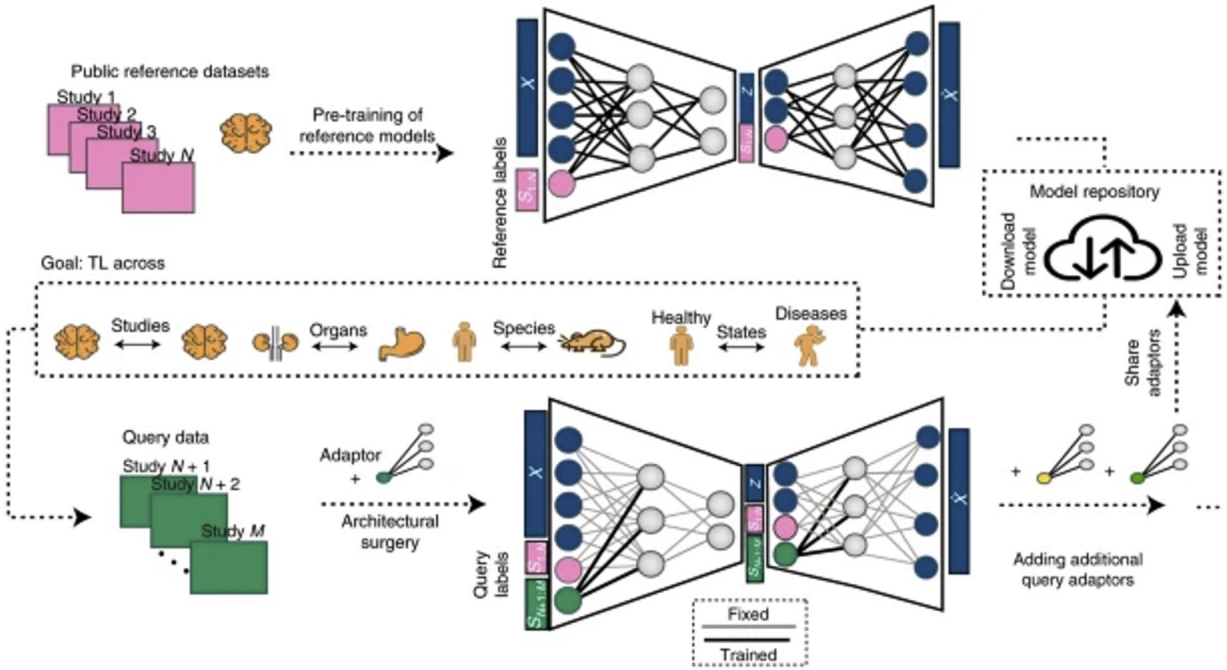
Figure S1


A
RNA
No correction
Harmony
BBKNN
Batch 1
Batch 2
Batch 3
B
PROT
No correction
Harmony
BBKNN
Batch 1
Batch 2
Batch 3
C
ATAC
No correction
Harmony
BBKNN
Batch 1
Batch 2
Batch 3
D
RNA
No correction
BBKNN
Harmony
Density
0.0
0.2
0.4
0.6
1.5
2.0
2.5
3.0
LISI score
E
PROT
No correction
BBKNN
Harmony
Density
0.0
0.2
0.4
0.6
1.0
1.5
2.0
2.5
3.0
LISI score
F
ATAC
No correction
BBKNN
Harmony
Density
0.0
0.2
0.4
0.6
1.0
1.5
2.0
2.5
3.0
LISI score

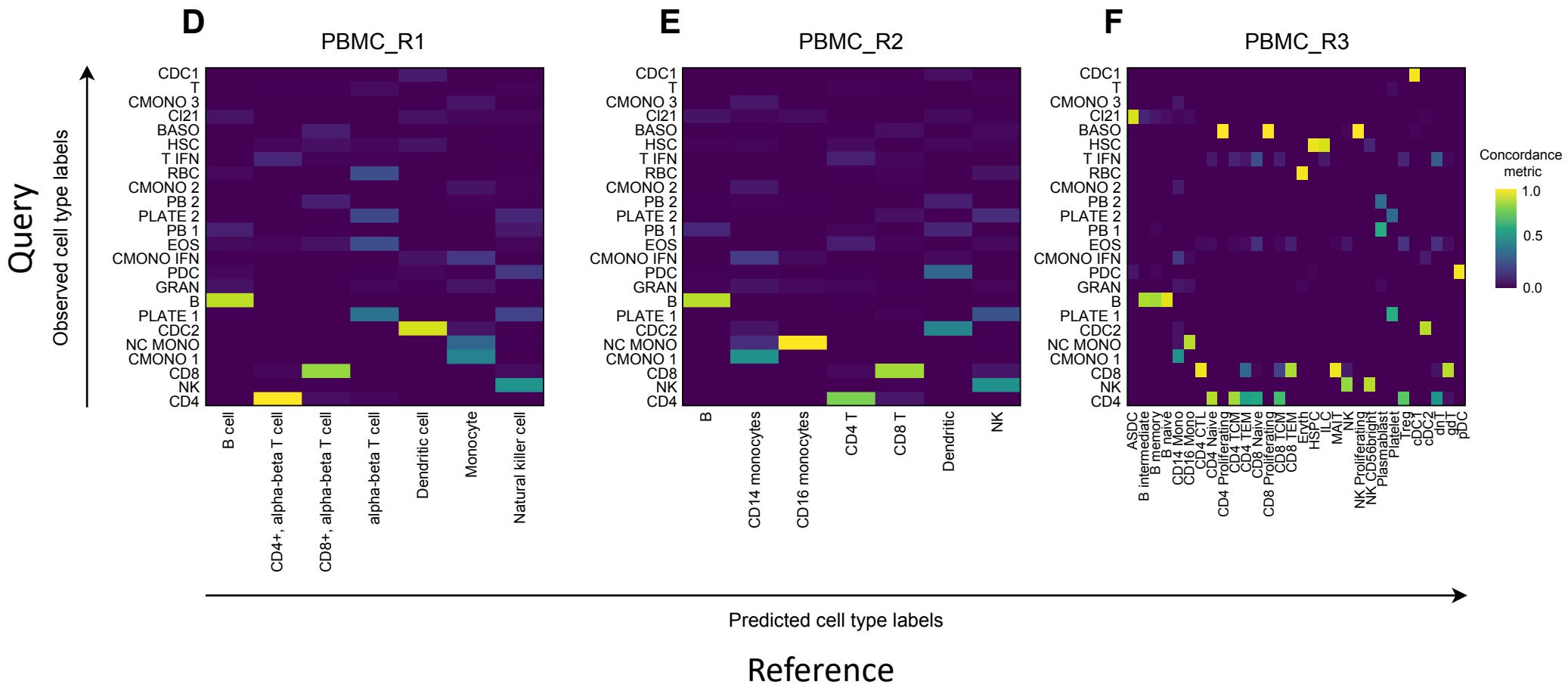

A
MultiVI (RNA+ATAC)
Batch 1
Batch 2
Batch 3
B
totalVI (RNA+PROT)
Batch 1
Batch 2
Batch 3
C
WNN (ATAC+PROT) - No batch correction
Batch 1
Batch 2
Batch 3
D
WNN (RNA+ATAC+PROT) - No batch correction
Batch 1
Batch 2
Batch 3
E
WNN (RNA+ATAC+PROT) - With batch correction
Batch 1
Batch 2
Batch 3
F
Density
0.0
0.1
0.2
0.3
0.4
1.0
1.5
2.0
2.5
3.0
LISI score
MultiVI (RNA+ATAC)
totalVI (RNA+PROT)
WNN (RNA+ATAC+PROT)
With batch correction
WNN (RNA+ATAC+PROT)
No batch correction
WNN (ATAC+PROT)
No batch correction

Reference mapping

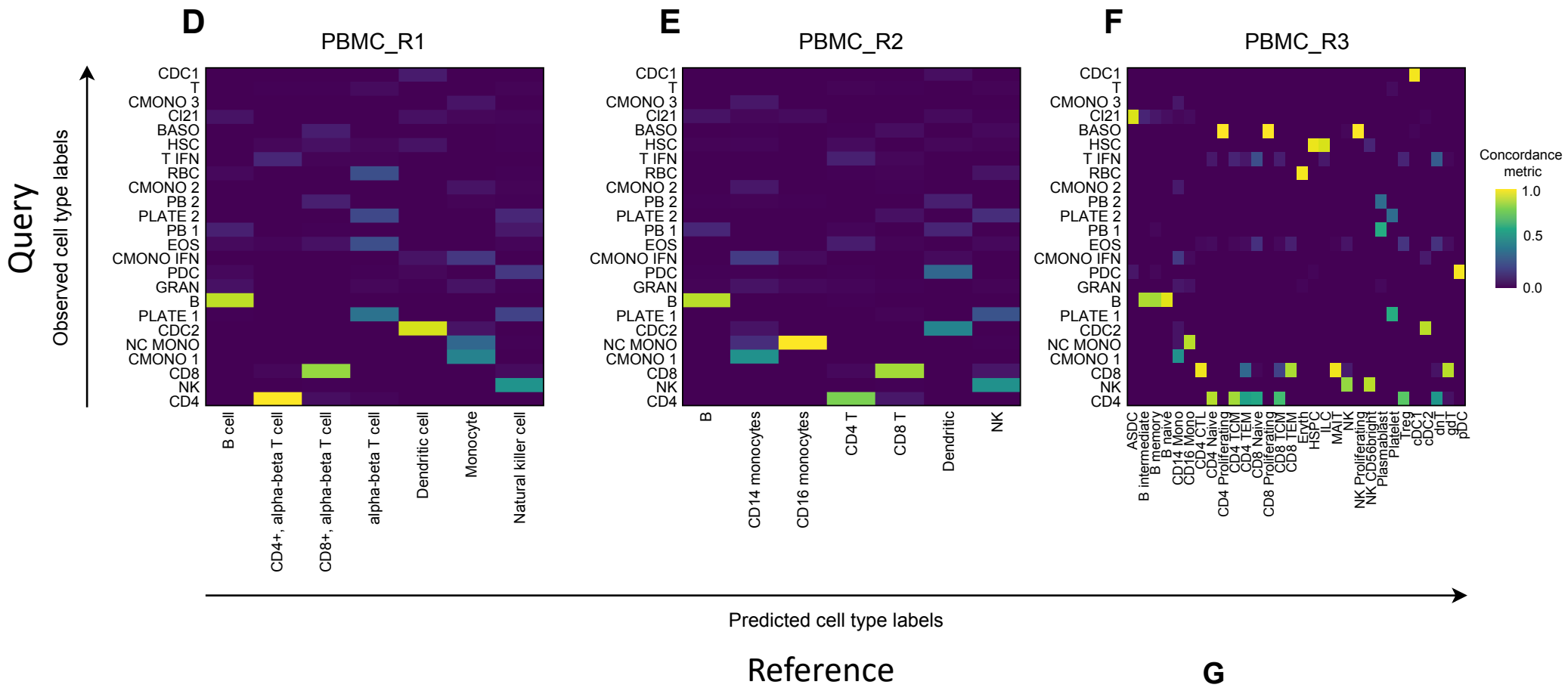




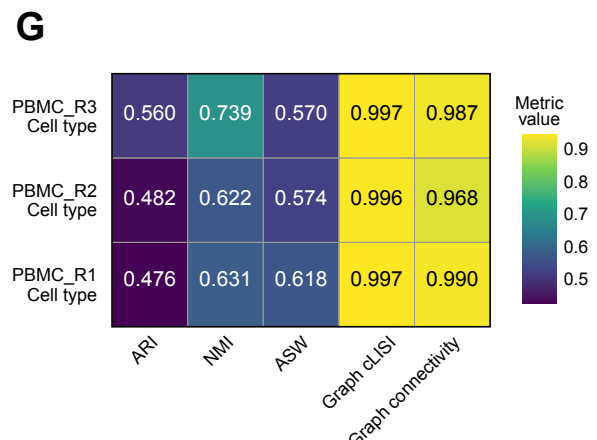
scArches: Lotfollahi et al. Nature Biotech 2022



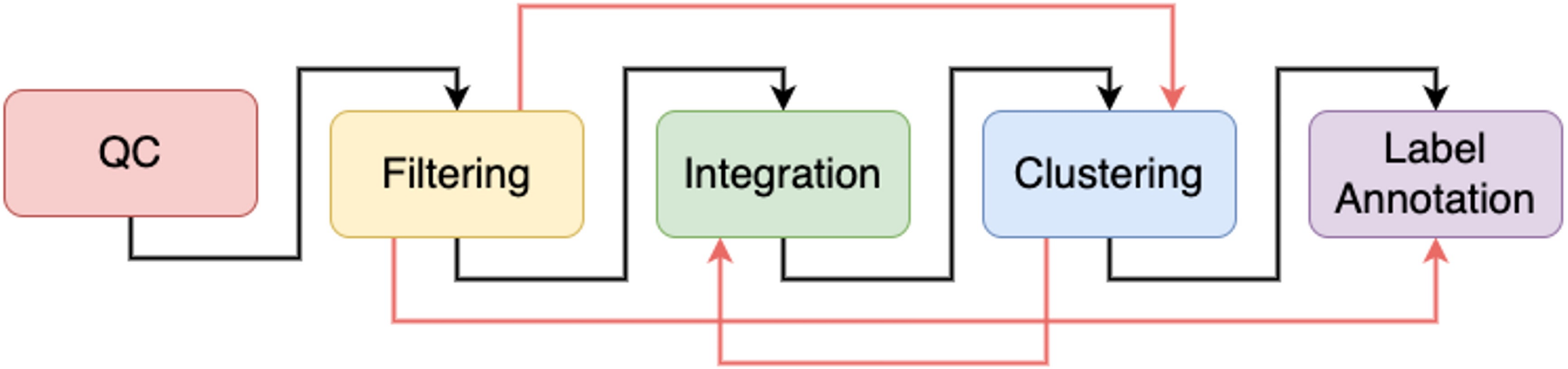
Comparing 1 query (rows)
to 3 references (columns)



Reference 3 best maps to our query



Panpipes is modular

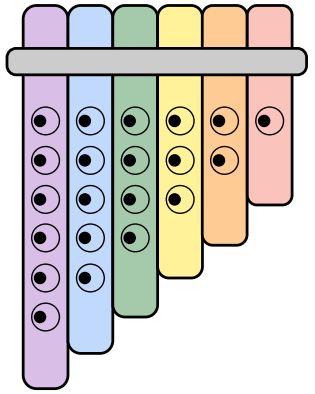


Acknowledgements

- **Fabiola Curion**

- Calli Dendrou
- Devika Agarwal
- Tom Thomas
- Melissa Grant-Peters
- The rest of the Dendrou Lab for their debugging efforts

Panpipes



Panpipes - tutorial

- The main tutorial is in the README.md
- https://github.com/DendrouLab/panpipes/blob/main/README.md

Panpipes installation

Today we are using python virtual env (but you can also use conda for this)

- Basic set up (only ever need to do once)
  1. Create .cgat.yml
  2. edit .bashrc to have the DRMAA path
  3. Set up git ssh keys
     https://github.com/DendrouLab/panpipes/blob/main/docs/set_up_ssh_keys_for_github.md

- Installation:
  1. Load Python modules (Forgetting to load modules first will cause errors)
  2. Create virtual env
  3. Activate virtual env¢
  4. Clone the repo
  5. Install the repo

- General installation instructions
  - https://github.com/DendrouLab/panpipes/blob/main/docs/install.md
- BMRC specific installation instructions:
  - https://github.com/DendrouLab/panpipes/blob/main/docs/installation_rescomp.md

QC MM (multimodal)

Running QC mm

Step 1: run configuration

```
panpipes qc_mm config
```

Step 2: check you have all the input files

For QCmm we need two files:

1. sample submission file
2. qc gene list file

Step 3: edit the pipeline.yml for your project details

Editing the pipeline.yml

Lets go through this together

- https://github.com/crichgriffin/panpipes_workshop/blob/main/qc_mm/pipeline.yml

Sample submission file

- This is the file that contains the paths to each of your samples (one sample per row)

- It must have the following columns:
  - Sample_id (a unique value on each row)
  - Gex_path (the path to the outs folder from cellranger)
  - Gex_filetype (see list of supported filetypes on github)

More details here:

https://github.com/DendrouLab/panpipes/blob/main/docs/setup_for_qc_mm.md

Gene list formats

In the qc_mm and vis pipelines, the user can provide custom genelists in order to compute gene list scores and to visualise

All [^1] gene lists provided to the pipeline should be in a 3 columns format, where the column headers are mod, feature and group. The gorup column is used to distinguish different gene groups.

mod	feature	group
rna	MT-ND1	mt
rna	MT-ND2	mt

The genes are not pre-determined within panpipes in order to maximise flexibility as all organisms will require separate lists.

More details:

https://github.com/DendrouLab/panpipes/blob/main/docs/gene_list_format.md

A list of QC genes is included in the test data folder!

Run the pipeline!

Package name

Instruction for ruffus:
Make = run all the steps
Show = show what steps will be run

```
panpipes qc_mm make full
```

Workflow names:
- qc_mm
- preprocess
- integration
- refmap
- clustering

Stopping point for pipeline.
Full = the last function in the pipeline, so all steps will be run

You could substitute full for any task name from the pipeline

panpipes qc_mm show full

```
_____________________________________
Tasks which will be run:

Task = 'pipeline_qc_mm.load_mudatas'
Task = 'pipeline_qc_mm.concat_filtered_mudatas'
Task = "mkdir('scrublet')   before pipeline_qc_mm.run_scrublet "
Task = 'pipeline_qc_mm.run_scrublet'
Task = 'pipeline_qc_mm.run_rna_qc'
Task = 'pipeline_qc_mm.load_bg_mudatas'
Task = 'pipeline_qc_mm.downsample_bg_mudatas'
Task = 'pipeline_qc_mm.concat_bg_mudatas'
Task = 'pipeline_qc_mm.run_scanpy_prot_qc'
Task = 'pipeline_qc_mm.run_dsb_clr'
Task = 'pipeline_qc_mm.run_prot_qc'
Task = 'pipeline_qc_mm.run_repertoire_qc'
Task = 'pipeline_qc_mm.run_atac_qc'
Task = 'pipeline_qc_mm.run_qc'
Task = "mkdir('figures/background')   before pipeline_qc_mm.run_assess_background "
Task = 'pipeline_qc_mm.run_assess_background'
Task = 'pipeline_qc_mm.plot_qc'
Task = 'pipeline_qc_mm.all_rna_qc'
Task = 'pipeline_qc_mm.all_prot_qc'
Task = 'pipeline_qc_mm.full'
```

Outputs:

- .h5mu object containing all your samples.
- Cell_metadata.csv contains all the metadata per barcode, including the QC metrics (can be used for plotting without loading all the data up)
- Figures visualizing metrics folder.

Preprocess

Repeat for preprocess

- `panpipes preprocess config`

- Let's look at the yml:
- https://github.com/crichgriffin/panpipes_workshop/blob/main/pp/pipeline.yml

Dynamic filtering

- Open up your unfiltered h5mu

```python
import muon as mu
mdata = mu.read("10k_PBMC_TBNK_connect_unfilt.h5mu")
mdata.obs.head() # look at the top of the data frame

mdata.obs.columns.tolist() # look at the names of all the columns in obs

mdata['rna'].obs.columns.tolist() # look at the column names in RNA obs only
mdata['rna'].var.columns.tolist() # look at the column names in RNA obs only
```

Panpipes can filter on **ANY** column in your data object

(the column names might vary depending on your qc gene list)

Dynamic filtering

```
#------------------------------------------------------
rna:
#------------------------------------------------------
  ## obs filtering: cell level filtering here
  obs:
    min:
      n_genes_by_counts: 500
    max:
      total_counts: 50000
      n_genes_by_counts:
      # percent filtering:
      # this should be a value between 0 and 100%.
      # leave blank or set to 100 to avoid filtering for any of these param
      pct_counts_mt: 25
      pct_counts_rp:
      # either one score for all samples e.g. 0.25,
      # or a csv file with two columns sample_id, and cut off
      # less than
      doublet_scores: 0.25
      #  if you wanted to be more precise i.e. apply a different srublet threshold per sample
      # you could add a new column to the mudata['rna'].obs with True False values, and list
      # that column under bool:, you can do this for any modality
    bool:
      predicted_doublets: False
  ## var filtering: (feature) gene level filtering here
  var:
    min:
      n_cells_by_counts:
    max:
      total_counts:
      n_cells_by_counts:
```

```
modality:
  obs:
    min:
      column_name: value # will remove any barcodes below this value
    max:
      column_name: value # will remove any barcodes above this value
    bool:
      columns_name: False # will keep only barcodes that are False
```

Integration

Repeat for preprocess

- `panpipes integration config`

- Let's look at the yml:
- https://github.com/crichgriffin/panpipes_workshop/blob/main/int/pipeline.yml

- `panpipes integration make full`

    - all the batch corrected objects go into tmp
    - decide you favourites and set it at the bottom of the pipeline.yml then run:

- `panpipes integration make merge_batch_correction`

    - Then delete tmp to save on space