KSBi-BIML 2024

Bioinformatics & Machine Learning(BIML)
Workshop for Life and Medical Scientists

생명정보학 & 머신러닝 워크샵 (오프라인)

Single-cell Multiomics

최정민 _ 고려대학교

KSBI KOREAN SOCIETY FOR BIOINFORMATICS | 한국생명정보학회

KSBi-BIML 2024

Bioinformatics & Machine Learning(BIML)
Workshop for Life and Medical Scientists

안녕하십니까?

한국생명정보학회가 개최하는 동계 교육 워크샵인 BIML-2024에 여러분을 초대합니다. 생명정보학 분야의 연구자들에게 최신 동향의 데이터 분석기술을 이론과 실습을 겸비해 전달하고자 도입한 전문 교육 프로그램인 BIML 워크샵은 2015년에 시작하여 올해로 벌써 10년 차를 맞이하게 되었습니다. BIML 워크샵은 국내 생명정보학 분야의 최초이자 최고 수준의 교육프로그램으로 크게 인공지능과 생명정보분석 두 개의 분야로 구성되어 있습니다. 올해 인공지능 분야에서는 최근 생명정보 분석에서도 응용이 확대되고 있는 다양한 인공지능 기반 자료모델링 기법들에 대한 현장 강의가 진행될 예정이며, 관련하여 심층학습을 이용한 단백질구조예측, 유전체분석, 신약개발에 대한 이론과 실습 강의가 함께 제공될 예정입니다. 또한 단일세포오믹스, 공간오믹스, 메타오믹스, 그리고 롱리드염기서열 자료 분석에 대한 현장 강의는 많은 연구자의 연구 수월성 확보에 큰 도움을 줄 것으로 기대하고 있습니다.

올해 BIML의 가장 큰 변화는 최근 연구 수요가 급증하고 있는 의료정보자료 분석에 대한 현장 강의를 추가하였다는 것입니다. 특히 의료정보자료 분석을 많이 수행하시는 의과학자 및 의료정보 연구자들께서 본 강좌를 통해 많은 도움을 받으실 수 있기를 기대하고 있습니다. 또한 다양한 생명정보학 분야에 대한 온라인 강좌 프로그램도 점차 증가하고 있는 생명정보 분석기술의 다양화에 발맞추기 위해 작년과 비교해 5강좌 이상을 신규로 추가했습니다. 올해는 무료 강좌 5개를 포함하여 35개 이상의 온라인 강좌가 개설되어 제공되며, 연구 주제에 따른 연관된 강좌 추천 및 강연료 할인 프로그램도 제공되며, 온라인을 통한 Q&A 세션도 마련될 예정입니다. BIML-2024는 국내 주요 연구중심 대학의 전임 교원이자 각 분야 최고 전문가들의 강의로 구성되었기에 해당 분야의 기초부터 최신 연구 동향까지 포함하는 수준 높은 내용의 강의가 될 것이라 확신합니다.

BIML-2024을 준비하기까지 너무나 많은 수고를 해주신 운영위원회의 정성원, 우현구, 백대현, 김태민, 김준일, 김상우, 장혜식, 박종은 교수님과 KOBIC 이병욱 박사님께 커다란 감사를 드립니다. 마지막으로 부족한 시간에도 불구하고 강의 부탁을 흔쾌히 허락하시고 훌륭한 현장 강의와 온라인 강의를 준비하시는데 노고를 아끼지 않으신 모든 강사분들께 깊은 감사를 드립니다.

2024년 2월

한국생명정보학회장 이 인 석

강의 시간표

DAY1 : 2월 24일 (토)

시간	강 의 (자연과학대학 28동 101호)
12:30-12:50	등록
12:50-13:00	공지사항 전달
13:00-14:30	**의료빅데이터/인공지능 총론** 김헌성 교수(가톨릭대학교)
14:30-14:45	휴식
14:45-16:15	**의료영상 인공지능의 이해 및 의료영상 레이블링 실습** 백서연 교수(연석대학교)
16:15-16:30	휴식
16:30-18:00	**의료 정보처리 자동화 실습 / 독자적인 어플리케이션 만들기** 김선근 대표(원닥 주식회사), 서사도 조교

시간	강 의 (자연과학대학 28동 102호)
12:30-12:50	등록
12:50-13:00	공지사항 전달
13:00-14:20	**EMR 데이터를 활용한 머신러닝 기반 예후예측:** **Decision Tree-based Models + EMR 샘플 데이터 실습 (MIMIC sample dataset)** 고태훈 교수(가톨릭대학교)
14:20-14:40	휴식
14:40-16:00	**Chest X-ray 영상을 활용한 딥러닝 기반 폐질환 진단:** **Convolutional Neural Network + 의료영상 샘플 데이터 실습 (NIH Chest X-ray14)** 고태훈 교수(가톨릭대학교)
16:00-16:20	휴식
16:20-17:40	**심전도 데이터를 활용한 딥러닝 기반 부정맥 탐지: Recurrent Neural Network +** **Transformer + 심전도 샘플 데이터 실습 (MIT-BIH Arrhythmia Database)** 고태훈 교수(가톨릭대학교)

DAY1 : 2월 26일 (월)

시간	강 의 (자연과학대학 28동 101호)
09:00-09:20	등록
09:20-09:30	공지사항 전달
09:30-10:50	**DNN (이론)** 이상근 교수(고려대학교)
10:50-11:00	휴식
11:00-12:10	**CNN (이론)** 이상근 교수(고려대학교)
12:10-13:40	점심
13:40-15:10	**RNN, ChatGPT, XAI (이론)** 이상근 교수(고려대학교)
15:10-15:20	휴식
15:20-16:50	**CNN/RNN 모델 구조 정의, 학습 알고리즘 적용, 성능 평가, 시각화 방법 (Tensorflow 실습)** 이정현 조교, 한성민 조교

시간	강 의 (자연과학대학 28동 102호)
09:00-09:20	등록
09:20-09:30	공지사항 전달
09:30-11:00	**Best practice for single-cell data analysis** 박종은 교수(KAIST)
11:00-11:10	휴식
11:10-12:40	**Practice1: Scanpy basic workflow** 정성민 조교, 고용준 조교
12:40-14:10	점심
14:10-15:30	**Public database, data integration, reference mapping, multiomics** 박종은 교수(KAIST)
15:30-15:40	휴식
15:40-16:50	**Practice2: Advanced single-cell analysis (siVI universe)** 정성민 조교, 고용준 조교

DAY1 : 2월 27일 (화)

시간	강 의 (자연과학대학 28동 101호)
09:00-09:20	등록
09:20-09:30	공지사항 전달
09:30-10:50	**AI-based protein structure prediction** **- Intro to protein structure prediction** **- Early AI-based approaches** **- AlphaFold and RoseTTAFold** 백민경 교수(서울대학교)
10:50-11:00	휴식
11:00-12:10	**단백질 구조 예측 실습** **- ColabFold를 활용한 단백질 구조 및 상호작용 예측** **- Tips &Tricks for better structure modeling** 백민경 교수(서울대학교)
12:10-13:40	점심
13:40-15:10	**AI-based protein design** **- Intro to protein design** **- Protein backbone design using RFdiffusion** **- Protein sequence design using ProteinMPNN** 백민경 교수(서울대학교)
15:10-15:20	휴식
15:20-16:50	**단백질 디자인 실습** **- RFdiffusion 및 ProteinMPNN의 활용법 실습** 백민경 교수(서울대학교)

시간	강 의 (자연과학대학 28동 102호)
09:00-09:20	등록
09:20-09:30	공지사항 전달
09:30-11:00	**Introduction to Single-cell biology** 최정민 교수(고려대학교)
11:00-11:10	휴식
11:10-12:40	**i. Unsupervised Spatial transcriptome analysis** **ii. Tumor Boundary Determination in Spatial Transcriptomics** 유광민 조교, 이문영 조교
12:40-14:10	점심
14:10-15:30	**i. Deconvolution Analysis Using Single-cell RNA Sequencing and Spatial Transcriptomics** **ii. Cell-Cell Interaction Analysis in Spatial Transcriptomics** 김지현 조교, 최승지 조교
15:30-15:40	휴식
15:40-16:50	**i. Open Chromatin Region Analysis and Biological Interpretation of Using scATAC-seq Dataset** **ii. Construction of Gene Regulatory Networks Based on Integrated Analysis of scATAC-seq and scRNA-seq Datasets** 천하림 조교, 이호진 조교

DAY1 : 2월 28일 (수)

시간	강 의 (자연과학대학 28동 101호)
09:00-09:20	등록
09:20-09:30	공지사항 전달
09:30-11:00	**Introduction to Transformers (이론)** 전민지 교수 (고려대학교)
11:00-11:10	휴식
11:10-12:40	**Introduction to Transformers (실습)** 봉현수 조교, 임우택 조교
12:40-14:10	점심
14:10-15:40	**Deep learning in Bioinformatics** 노미나 교수(한양대학교)
15:40-15:50	휴식
15:50-17:20	**Deep learning model을 이용한 실습** 박예솔 조교

시간	강 의 (자연과학대학 28동 102호)
09:00-09:20	등록
09:20-09:30	공지사항 전달
09:30-10:50	**마이크로바이옴 기본 이론** 이선재 교수(GIST)
10:50-11:00	휴식
11:00-12:10	**16S rRNA amplicon seq. - DADA2** 조준우 조교, 백재우 조교
12:10-13:40	점심
13:40-14:40	**최신 메타지놈 분석 기법의 현황** 이선재 교수(GIST)
14:40-14:50	휴식
14:50-16:50	**Shotgun metagenome 분석 (Linux)** 조준우 조교, 백재우 조교

DAY1 : 2월 29일 (목)

시간	강 의 (자연과학대학 28동 101호)
09:00-09:20	등록
09:20-09:30	공지사항 전달
09:30-10:50	**화학정보학 기초(Cheminformatics) / 약물특성 및 약물다움(druglikeness)** **Molecular Notations &Descriptors / AI 신약개발을 위한 Databases** **AI 신약개발을 위한 Programming 기초** 김동섭 교수(KAIST)
10:50-11:00	휴식
11:00-12:10	**Google Colab에 RDKit 설치 / 화합물 정보 읽기 실습** **Bioactivity database 검색 및 정보 읽기 실습** **Molecular descriptor (fingerprint) 생성 및 similarity 계산 실습** 정수재 조교, 나민주 조교
12:10-13:40	점심
13:40-15:10	**AI 신약개발을 위한 기계학습법 기초 / QSAR 모델링 기초 / AI 신약개발을 위한 딥러닝 모델** **Virtual screening (ligand-based, structure-based) 및 de novo design** 김동섭 교수(KAIST)
15:10-15:20	휴식
15:20-16:50	**QSAR modeling 전체 과정 실습/ 화합물의 Bioactivity 예측 모델 개발** **Virtual screening 과정을 통한 신약후보물질 발굴 실습** 정수재 조교, 나민주 조교

시간	강 의 (자연과학대학 28동 102호)
09:00-09:20	등록
09:20-09:30	공지사항 전달
09:30-11:00	**Single cell multiomics 이론 / Gene regulatory network 이론** 김준일 교수(숭실대학교)
11:00-11:10	휴식
11:10-12:40	**Seurat/Signac, ArchR, TENET+ 실습** 김현규 조교, 정회빈 조교
12:40-14:10	점심
14:10-15:40	**롱리드 시퀀싱 소개 및 유전체 조립 실습** 김준 교수(충남대학교)
15:40-15:50	휴식
15:50-17:20	**변이 분석 및 시각화 실습** 김준 교수(충남대학교)

강의개요

Single-cell Multiomics

다양한 생명 현상을 개별 세포 차원에서 파악하고 이해하기 위해 Single cell genomics 기술이 발전하고 있으며, 이를 통해 단일 세포 수준의 전사체(transcriptomics), 유전체(genomics), 후성유전체(epigenomics), 단백체(proteomics) 및 공간 전사체(spatial transcriptomics) 데이터 연구가 활발히 진행 중이다. 이 강의에서는 R 프로그래밍을 기반으로 scRNA-seq, scATAC-seq 데이터와 10X visium 및 xenium을 포함하는 spatial transcriptomics 데이터 분석법을 다룬다. 각 데이터의 특성과 기본 분석 파이프라인을 소개하며, multi-omics 데이터의 통합적 분석을 통해 세포 간의 다양성을 확인하고 생물학적 기전을 심층적으로 이해하는 데 목표를 둔다.

강의 내용은 다음과 같다:

- Spatial Transcriptomics, Single cell ATAC-seq 소개
- 다양한 단일 세포 유전체 데이터의 전처리(preprocessing) 및 분석
- 단일 세포 유전체 데이터를 이용한 deconvolution 및 공간 전사체 데이터를 활용한 세포 간 상호작용 분석 연구

* 교육생준비물:
노트북 (메모리 8GB 이상, 디스크 여유공간 30GB 이상)
분석에 필요한 R library packages list를 제공할 예정이니 원활한 강의 진행을 위해 강의 전에 모두 설치해 오기 바랍니다.

* 강의 난이도: 초급-중급

* 강의: 최정민 (고려대학교 의과학과 의료정보학 교실)
실습: 천하림, 김지현, 유광민, 이호진, 이문영, 홍주현, 이다준, 최승지

Curriculum Vitae

Speaker Name: Jungmin Choi, Ph.D.

▶ Personal Info

Name	Jungmin Choi
Title	Associate Professor
Affiliation	Korea University

▶ Contact Information

Address	73, Goryeodae-ro, Seongbuk-gu, Seoul 02841, South Korea
Email	jungminchoi@korea.ac.kr
Phone Number	02-2286-1469

Research Interest

Genetics, genomics, computational biology

Educational Experience

2012	Ph.D. in Genetics, University of Maryland, USA
2004	B.S. in Chemistry, Yonsei university, Korea

Professional Experience

2018-2019	Research Associate, Rockefeller University, USA
2013-2018	Postdoctoral research fellow, Yale University, USA

Selected Publications (5 maximum)

1. Jeong J, Lee J, Talaia G, Kim W, Song J, Hong J, Yoo K, Gonzalez DG, Athonvarangkul D, Shin J, Dann P, Haberman AM, Kim LK, Ferguson SM, **Choi J**, Wysolmerski J. Intracellular Calcium links Milk Stasis to Lysosome Dependent Cell Death During Early Mammary Gland Involution. Cell. Mol. Life Sci. 2023 in press.
2. Hwang JY, Chai P, Nawaz S, **Choi J**, Lopez-Giraldez F, Hussain S, Bilguvar K, Mane S, Lifton RP, Ahmad W, Zhang K, Chung JJ. LRRC23 truncation impairs radial spoke 3 head assembly and sperm motility underlying male infertility. Elife. 2023 Dec 13;12:RP90095. doi: 10.7554/eLife.90095. PMID: 38091523; PMCID: PMC10721216.
3. Cho JM, Park HC, Lee JW, Ryu H, Kim YC, Ahn C, Lee KB, Kim YH, Han S, Kim Y, Bae EH, Kang HG, Park E, Jeong K, Kang S, **Choi J**, Oh KH, Oh YK. Baseline characteristics of the Korean genetic cohort of inherited cystic kidney disease. Kidney Res Clin Pract. 2023 Sep;42(5):617-627. doi: 10.23876/j.krcp.23.097. Epub 2023 Sep 27. PMID: 37813524; PMCID: PMC10565461.
4. Kim Y, Park HC, Ryu H, Kim YC, Ahn C, Lee KB, Kim YH, Han S, Bae EH, Jeong K, **Choi J**, Oh KH, Oh YK. Factors Associated With the Development and Severity of Polycystic Liver in Patients With Autosomal Dominant Polycystic Kidney Disease. J Korean Med Sci. 2023 Sep 25;38(38):e296. doi: 10.3346/jkms.2023.38.e296. PMID: 37750370; PMCID: PMC10519778.
5. Cho S, Chun Y, He L, Ramirez CB, Ganesh KS, Jeong K, Song J, Cheong JG, Li Z, **Choi J**, Kim J, Koundouros N, Ding F, Dephoure N, Jang C, Blenis J, Lee G. FAM120A couples SREBP-dependent transcription and splicing of lipogenesis enzymes downstream of mTORC1. Mol Cell. 2023 Aug 17;83(16):3010-3026.e8. doi: 10.1016/j.molcel.2023.07.017. PMID: 37595559; PMCID: PMC10494788.

KSBI | 한국생명정보학회

KSBi-BIML 2024

Table of Contents

1. What is R programming?

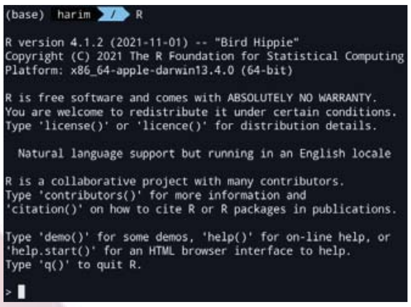
What is R and Why R?

- R is used widely in biological research and provides a solid platform for beginner scientific programmers.
- It's free and open-source.
- It runs on all major operating systems.
- R is the most common statistics platform in genomics and easy to use.

4

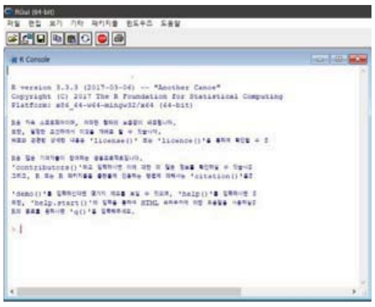
Several ways to use R

Command line



```
(base) harim / R

R version 4.1.2 (2021-11-01) -- "Bird Hippie"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.4.0 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> 
```

Graphical user interface



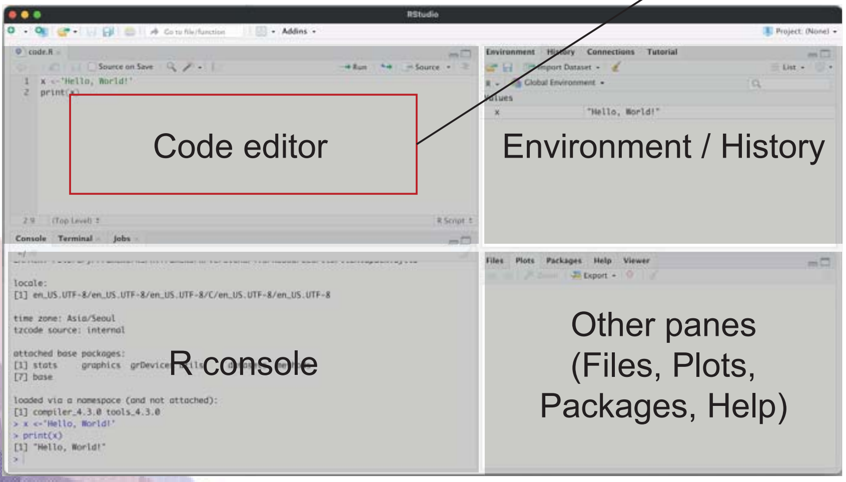
```
R version 3.3.3 (2017-03-06) -- "Another Canoe"
Copyright (C) 2017 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

[illegible]
[illegible]
[illegible] 'license()' [illegible] 'licence()'[illegible]

[illegible]
'contributors()'[illegible]
[illegible] 'citation()'[illegible]

'demo()'[illegible] 'help()'[illegible]
[illegible] 'help.start()'[illegible] HTML [illegible]
[illegible] 'q()'[illegible]

> 
```

Rstudio

Rstudio

- Rstudio – Integrated development environment (IDE) for R and python

Code editor에서 코드를 작성하거나 적혀져 있는 코드를 실행합니다.
Ctrl+Enter를 통해 각 한 줄씩의 코드를 실행할 수 있습니다.



```
x <-'Hello, World!'
print(x)
```

```
locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

time zone: Asia/Seoul
tzcode source: internal

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods
[7] base

loaded via a namespace (and not attached):
[1] compiler_4.3.0 tools_4.3.0
> x <-'Hello, World!'
> print(x)
[1] "Hello, World!"
> 
```

2. Space Ranger

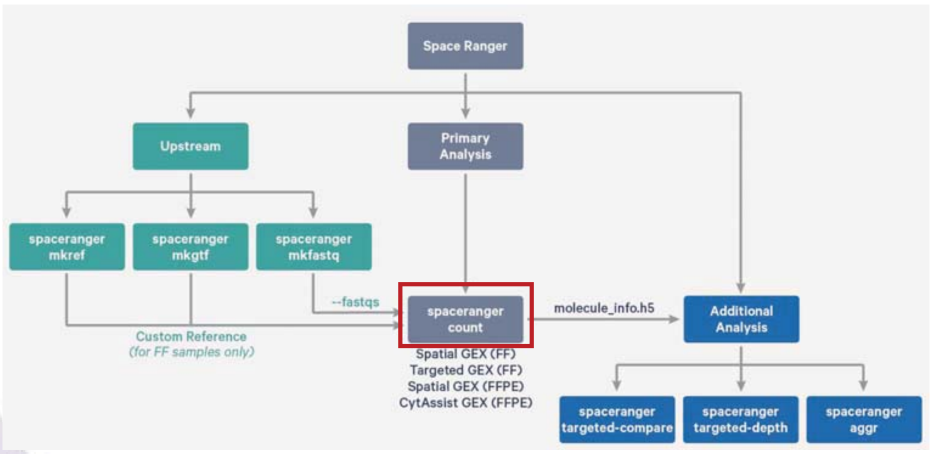
What is Space Ranger?

- Space Ranger is a set of analysis pipelines for processing 10X Genomics Visium sequence data (FAST Q files) with high resolution microscope images of tissue.
- It maps the transcriptomic reads to the microscope image of the tissue from which the reads were obtain ed
- We will introduce spaceranger count pipeline among the 5 pipelines

8

Space Ranger Pipelines

- spaceranger mkfastq
- **spaceranger count**
- spaceranger aggr
- spaceranger targeted-compare
- spaceranger targeted-depth



https://support.10xgenomics.com/spatial-gene-expression/software/pipelines/latest/what-is-space-ranger

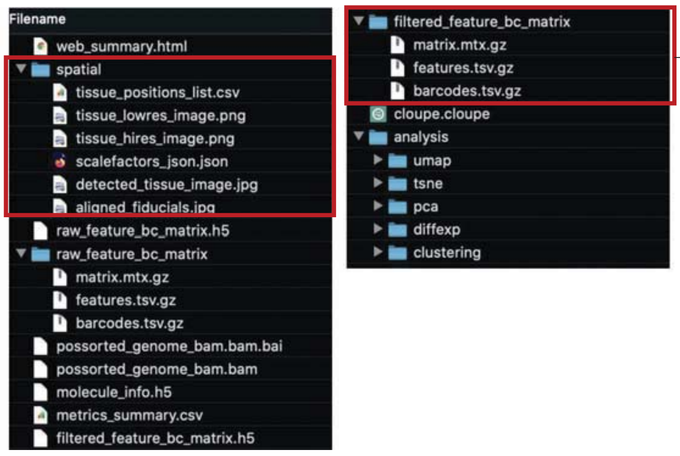
Run *sapceranger count* command

# Do not run below
```
$cd /home/jdoe/runs
$spaceranger count --id=sample345 \ #Output directory
                   --transcriptome=/home/jdoe/refdata/GRCh38-2020-A
\ #Path to Reference
                   --fastqs=/home/jdoe/runs/HAWT7ADXX/outs/fastq_path
\ #Path to FASTQs
                   --sample=mysample \ #Sample name from FASTQ filename
                   --image=/home/jdoe/runs/images/sample345.tiff \ #Path
to brightfield image
                   --slide=V19J01-123 \ #Slide ID
                   --area=A1 \ #Capture area
                   --localcores=8 \ #Allowed cores in localmode
                   --localmem=64 #Allowed memory (GB) in localmode
```

https://support.10xgenomics.com/spatial-gene-expression/software/pipelines/latest/using/count

- Input : the microscope image (.tiff), FASTQ files(Fastq)
- Perform : sequence alignment, tissue detection
- Output : gene-spot matrix

Output files of Space Ranger



```
Filename
  web_summary.html
▼ spatial
    tissue_positions_list.csv
    tissue_lowres_image.png
    tissue_hires_image.png
    scalefactors_json.json
    detected_tissue_image.jpg
    aligned_fiducials.jpg
  raw_feature_bc_matrix.h5
▼ raw_feature_bc_matrix
    matrix.mtx.gz
    features.tsv.gz
    barcodes.tsv.gz
  possorted_genome_bam.bam.bai
  possorted_genome_bam.bam
  molecule_info.h5
  metrics_summary.csv
  filtered_feature_bc_matrix.h5
```

```
▼ filtered_feature_bc_matrix
    matrix.mtx.gz
    features.tsv.gz
    barcodes.tsv.gz
  cloupe.cloupe
▼ analysis
  ▶ umap
  ▶ tsne
  ▶ pca
  ▶ diffexp
  ▶ clustering
```

→ read10XVisium Input files

11

Output files

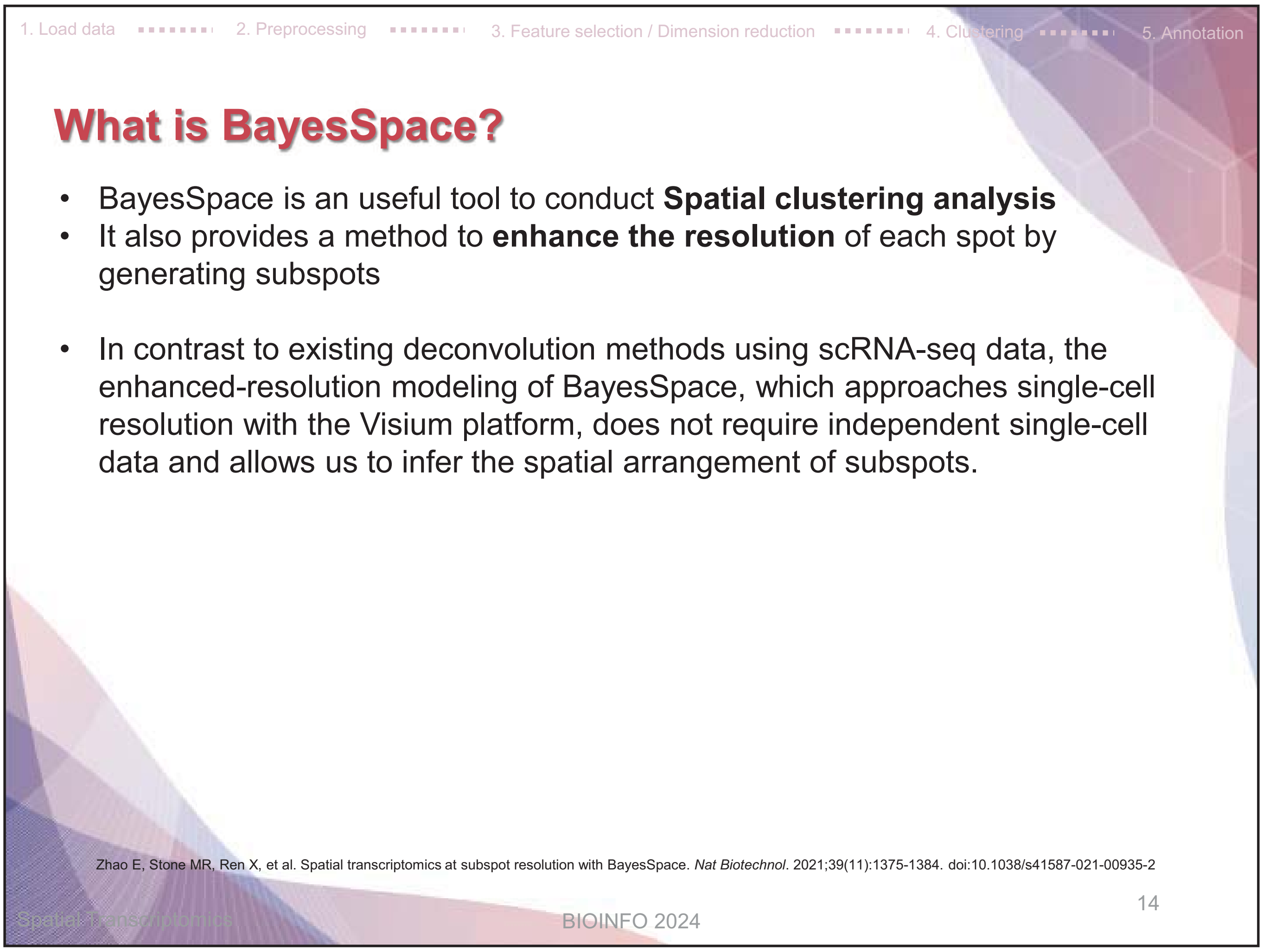
- **raw_feature_bc_matrix**

  Dataset having spots that theoretically don't overlap with tissue
- **filtered_feature_bc_matrix**

  Dataset filtered to the spots overlapping tissue, as determined by Loupe Browser (Visium) alignment file
- **tissue_positions_list.csv**
  The spot coordinates information is stored
- **scalefactors_json.json**
  Scaling factors that convert spot coordinates to pixel coordinates
- **metrics_summary.csv**
  Metrics displayed in the interactive website
- **web_summary.html**
  Interactive website

12

3. BayesSpace



What is BayesSpace?

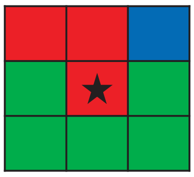
- BayesSpace is an useful tool to conduct **Spatial clustering analysis**
- It also provides a method to **enhance the resolution** of each spot by generating subspots

- In contrast to existing deconvolution methods using scRNA-seq data, the enhanced-resolution modeling of BayesSpace, which approaches single-cell resolution with the Visium platform, does not require independent single-cell data and allows us to infer the spatial arrangement of subspots.

Zhao E, Stone MR, Ren X, et al. Spatial transcriptomics at subspot resolution with BayesSpace. *Nat Biotechnol*. 2021;39(11):1375-1384. doi:10.1038/s41587-021-00935-2

BayesSpace algorithm

1. Initial labeling with non-spatial clustering method ex) mclust (Follows Mixture of Gaussian distribution)

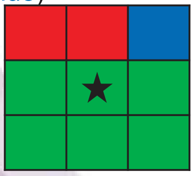


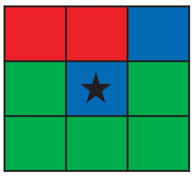
2. Calculate the P(red) for ★ spot

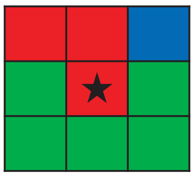
Metropolis–Hastings algorithm

3. Considering neighboring spots label and label of ★ at present, with certain function, calculate P(green), P(blue) for ★ spot

4. If P(green) > P(red) or P(blue) > P(red) update the label for ★ spot, if P(red) > P(green) & P(blue)





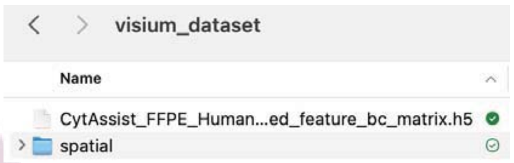


5. Repeat step 2~4 for set iteration

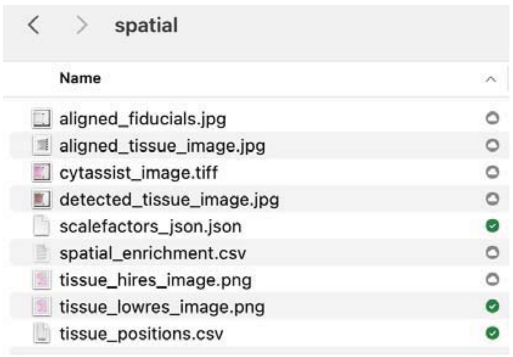
Zhao E, Stone MR, Ren X, et al. Spatial transcriptomics at subspot resolution with BayesSpace. *Nat Biotechnol*. 2021;39(11):1375-1384. doi:10.1038/s41587-021-00935-2

Load data

# Load 10X Genomics Visium dataset into Seurat
```
breast_seurat <- Seurat::Load10X_Spatial(
                    data.dir = 'visium_dataset',
                    filename =
'CytAssist_FFPE_Human_Breast_Cancer_filtered_feature_bc_matrix.h5',
                    assay = "Spatial", # specify name of the initial assay
                    slice = "slice1", # specify name of the stored image
                    filter.matrix = TRUE,
                    to.upper = FALSE
                    )
 # The directory contains the read count matrix H5 file and the image data in a
subdirectory called `spatial`.
```


visium_dataset

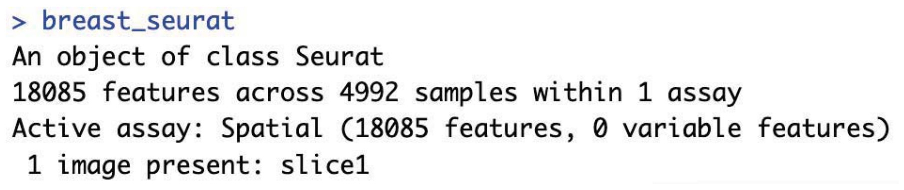
Name
- CytAssist_FFPE_Human...ed_feature_bc_matrix.h5
- spatial


spatial

Name
- aligned_fiducials.jpg
- aligned_tissue_image.jpg
- cytassist_image.tiff
- detected_tissue_image.jpg
- scalefactors_json.json
- spatial_enrichment.csv
- tissue_hires_image.png
- tissue_lowres_image.png
- tissue_positions.csv

Load data

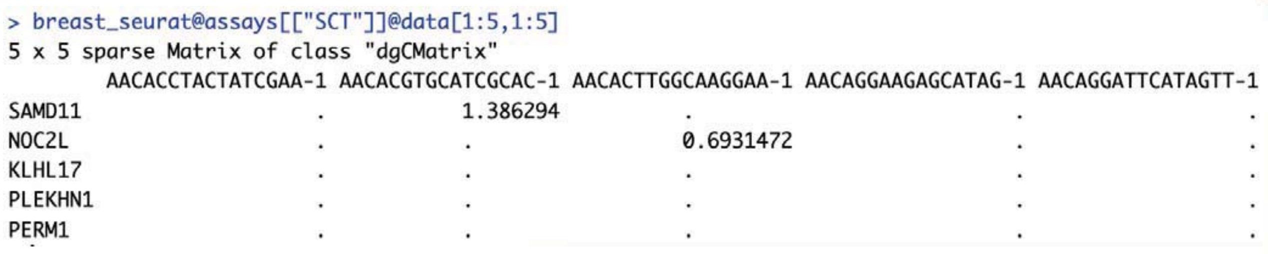
# Explore overall data
```
breast_seurat
```



```
> breast_seurat
An object of class Seurat
18085 features across 4992 samples within 1 assay
Active assay: Spatial (18085 features, 0 variable features)
 1 image present: slice1
```

Preprocessing

# Data normalization
```
breast_seurat <- SCTransform(breast_seurat, assay = "Spatial", verbose = FALSE)
```



```
> breast_seurat@assays[["SCT"]]@data[1:5,1:5]
5 x 5 sparse Matrix of class "dgCMatrix"
        AACACCTACTATCGAA-1 AACACGTGCATCGCAC-1 AACACTTGGCAAGGAA-1 AACAGGAAGAGCATAG-1 AACAGGATTCATAGTT-1
SAMD11                   .           1.386294                  .                  .                  .
NOC2L                    .                  .          0.6931472                  .                  .
KLHL17                   .                  .                  .                  .                  .
PLEKHN1                  .                  .                  .                  .                  .
PERM1                    .                  .                  .                  .                  .
```

The sctransform method models the UMI counts using a regularized negative binomial model to remove the variation due to sequencing depth (total nUMIs per cell), while adjusting the variance based on pooling information across genes with similar abundances

Feature selection and Dimension reduction

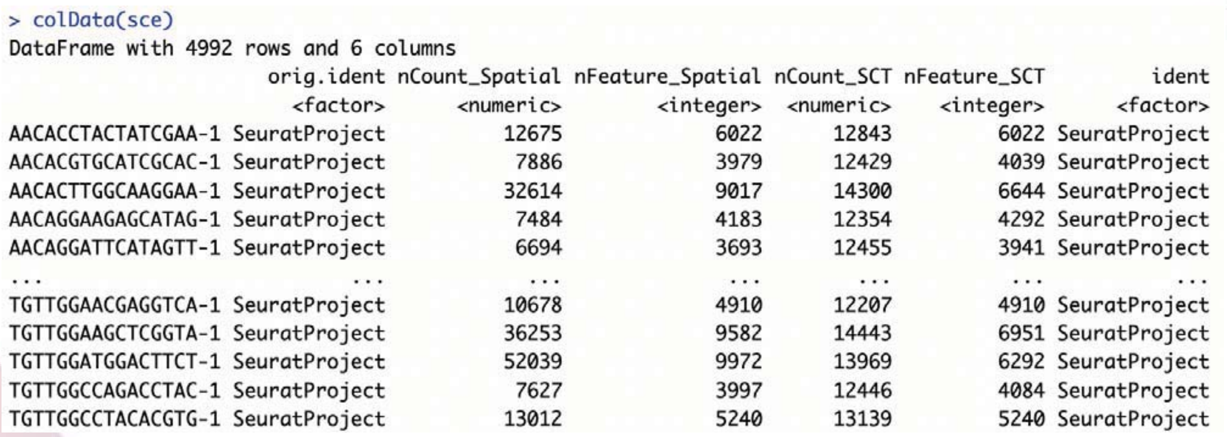
# PCA
```
breast_seurat <- RunPCA(breast_seurat, assay = "SCT", verbose = FALSE)
```

# Slim down Seurat obj prior to conversion
```
breast_seurat_diet = Seurat::DietSeurat(breast_seurat, graphs = "pca")
```

# Convert seurat to SCE
```
sce = as.SingleCellExperiment(breast_seurat_diet)
colData(sce) = cbind(colData(sce), breast_seurat@images$slice1@coordinates)
```

Feature selection and Dimension reduction data

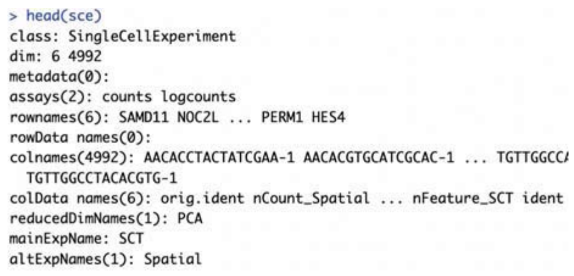
# Explore Col data
```
colData(sce)
```



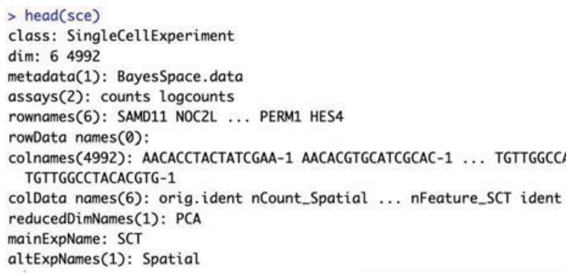
```
> colData(sce)
DataFrame with 4992 rows and 6 columns
                      orig.ident nCount_Spatial nFeature_Spatial nCount_SCT nFeature_SCT         ident
                        <factor>      <numeric>        <integer>  <numeric>    <integer>      <factor>
AACACCTACTATCGAA-1 SeuratProject          12675             6022      12843         6022 SeuratProject
AACACGTGCATCGCAC-1 SeuratProject           7886             3979      12429         4039 SeuratProject
AACACTTGGCAAGGAA-1 SeuratProject          32614             9017      14300         6644 SeuratProject
AACAGGAAGAGCATAG-1 SeuratProject           7484             4183      12354         4292 SeuratProject
AACAGGATTCATAGTT-1 SeuratProject           6694             3693      12455         3941 SeuratProject
...                          ...            ...              ...        ...          ...           ...
TGTTGGAACGAGGTCA-1 SeuratProject          10678             4910      12207         4910 SeuratProject
TGTTGGAAGCTCGGTA-1 SeuratProject          36253             9582      14443         6951 SeuratProject
TGTTGGATGGACTTCT-1 SeuratProject          52039             9972      13969         6292 SeuratProject
TGTTGGCCAGACCTAC-1 SeuratProject           7627             3997      12446         4084 SeuratProject
TGTTGGCCTACACGTG-1 SeuratProject          13012             5240      13139         5240 SeuratProject
```

Feature selection and Dimension reduction

```
library(BayesSpace)
sce = spatialPreprocess(sce, platform = "Visium", skip.PCA = T,
                                 log.normalize = F)
```



```
> head(sce)
class: SingleCellExperiment
dim: 6 4992
metadata(0):
assays(2): counts logcounts
rownames(6): SAMD11 NOC2L ... PERM1 HES4
rowData names(0):
colnames(4992): AACACCTACTATCGAA-1 AACACGTGCATCGCAC-1 ... TGTTGGCC
  TGTTGGCCTACACGTG-1
colData names(6): orig.ident nCount_Spatial ... nFeature_SCT ident
reducedDimNames(1): PCA
mainExpName: SCT
altExpNames(1): Spatial
```



```
> head(sce)
class: SingleCellExperiment
dim: 6 4992
metadata(1): BayesSpace.data
assays(2): counts logcounts
rownames(6): SAMD11 NOC2L ... PERM1 HES4
rowData names(0):
colnames(4992): AACACCTACTATCGAA-1 AACACGTGCATCGCAC-1 ... TGTTGGCC
  TGTTGGCCTACACGTG-1
colData names(6): orig.ident nCount_Spatial ... nFeature_SCT ident
reducedDimNames(1): PCA
mainExpName: SCT
altExpNames(1): Spatial
```

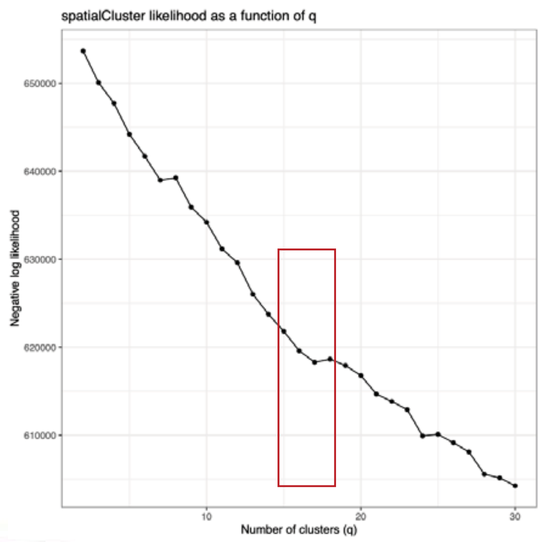
Clustering

```
breast_seurat <- RunPCA(breast_seurat, assay = "SCT", verbose = FALSE)
```

# Selecting the number of clusters

```
sce <- qTune(sce, qs=seq(2, 30), d=50)
qPlot(sce)
```

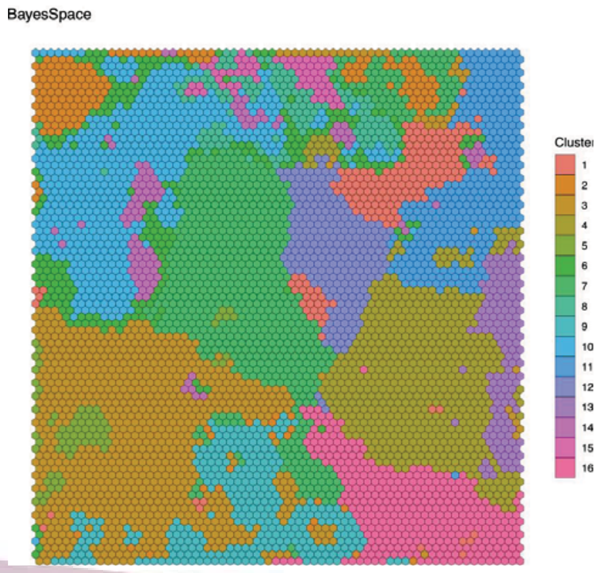
We suggest choosing a q around the elbow of this plot.



Visualization of our data

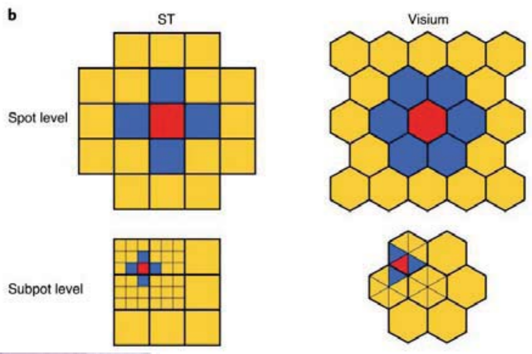
# Clustering with BayesSpace
```
sce = spatialCluster(sce, nrep = 10000, q = 16, d=50)
breast_sce <- sce
```

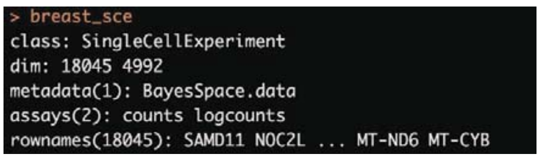
# Visualization of clustered data
```
clusterPlot(breast_sce, color="black", size=0.1)+labs(title="BayesSpace")
```

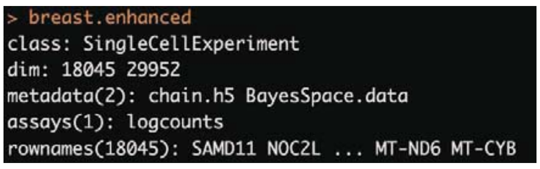


Enhancing the resolution (1/2)

# Enhance the resolution of the principal components
```
breast.enhanced = spatialEnhance(breast_sce, q=16, d=10,
                                   platform="Visium", gamma=2, nrep=1000,
                                   verbose=TRUE, save.chain=TRUE,
                              jitter_scale=3.5, jitter_prior=0.3,
                    burn.in=100)

breast.enhanced = readRDS('data/20240211_breast_enhanced_dl.rds')
breast_sce
breast.enhanced
```

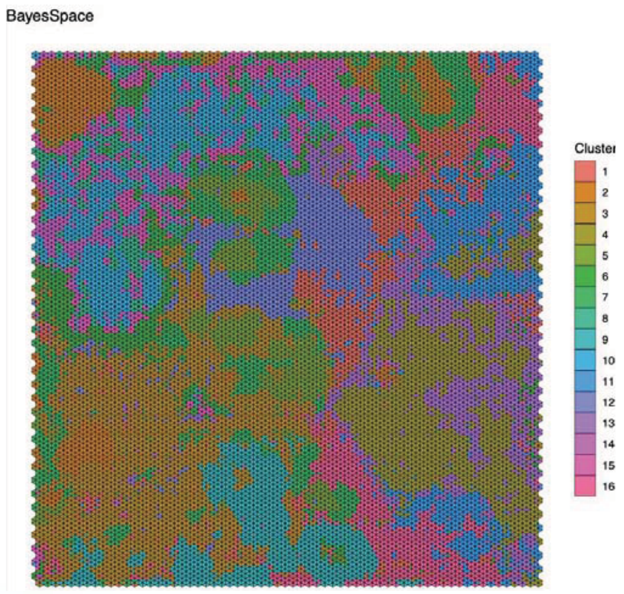



```
> breast_sce
class: SingleCellExperiment
dim: 18045 4992
metadata(1): BayesSpace.data
assays(2): counts logcounts
rownames(18045): SAMD11 NOC2L ... MT-ND6 MT-CYB
```


```
> breast.enhanced
class: SingleCellExperiment
dim: 18045 29952
metadata(2): chain.h5 BayesSpace.data
assays(1): logcounts
rownames(18045): SAMD11 NOC2L ... MT-ND6 MT-CYB
```

Enhancing the resolution (2/2)

```
# Visualization of enhanced data
clusterPlot(breast.enhanced, color="black", size=0.1) + labs(title="BayesSpace")
```



Define marker genes for plotting

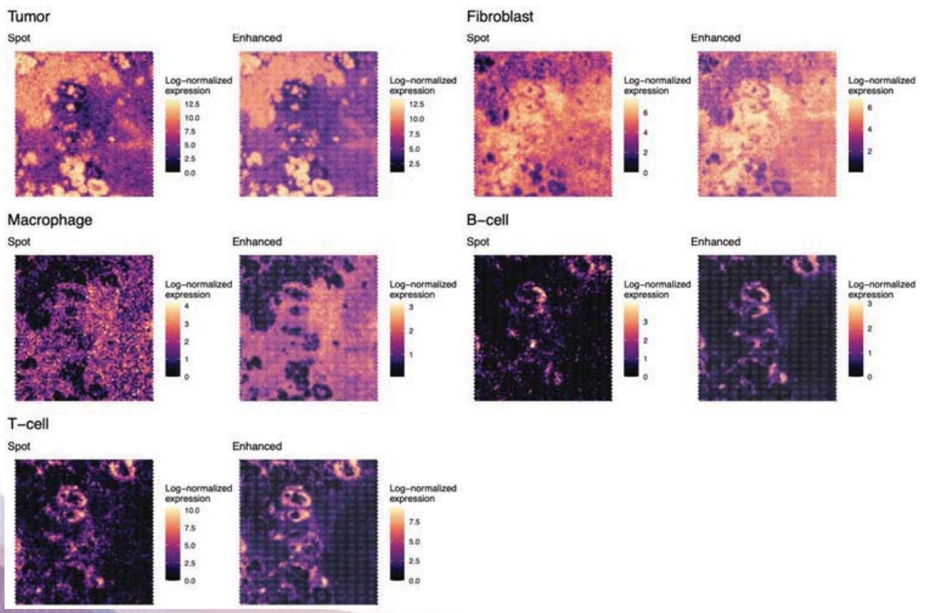
```
markers = list()
markers[["Tumor"]] = c("TACSTD2", "ERBB2", ESR1', 'PGR', 'FASN')
markers[["Fibroblast"]] = c("COL1A1")
markers[["Macrophage"]] = c("CD14", "FCGR1A", "FCGR1B")
markers[["B-cell"]] = c("CD19", "MS4A1")
markers[["T-cell"]] = c("CD2", "CD3D", "CD3E", "CD3G", "CD7")
marker_genes <- c("TACSTD2", "ERBB2", 'ESR1', 'PGR', 'FASN', 'COL1A1', 'CD14',
                  'FCGR1A', 'FCGR1B', 'CD19', 'MS4A1', "CD2", "CD3D", "CD3E",
                  "CD3G", "CD7")

breast.enhanced = enhanceFeatures(breast.enhanced, breast_sce,
                            model="xgboost",feature_names=marker_genes, nrounds=0)

sum_counts = function(sce, features) {
  if (length(features) > 1) {
    colSums(logcounts(sce)[features, ])
  } else {
    logcounts(sce)[features, ]
  }
}
```

Define marker genes for plotting

```
spot_expr = purrr::map(markers, function(xs) sum_counts(breast_sce, xs))
enhanced_expr = purrr::map(markers, function(xs)
                     sum_counts(breast.enhanced, xs))
plot_expression = function(sce, expr, title){
    featurePlot(sce, expr, color=NA) +
     viridis::scale_fill_viridis(option="A")+
    labs(title=title, fill="Log-normalized\nexpression")
}

plot_expression_comparison = function(cell_type){
  spot.plot = plot_expression(breast_sce, spot_expr[[cell_type]],"Spot")
  enhanced.plot = plot_expression(breast.enhanced,
                enhanced_expr[[cell_type]],"Enhanced")
  spot.plot + enhanced.plot +
  plot_annotation(title=cell_type,
                  theme=theme(plot.title=element_text(size=18)))
}
```

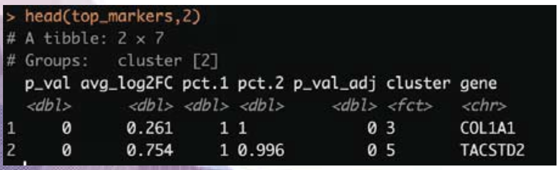
Rough cell type check

```
p1 = plot_expression_comparison("Tumor")
p2 = plot_expression_comparison("Fibroblast")
p3 = plot_expression_comparison("Macrophage")
p4 = plot_expression_comparison("B-cell")
p5 = plot_expression_comparison("T-cell")
cowplot::plot_grid(p1,p2,p3,p4,p5, ncol=2)
```



Find Marker genes for each other

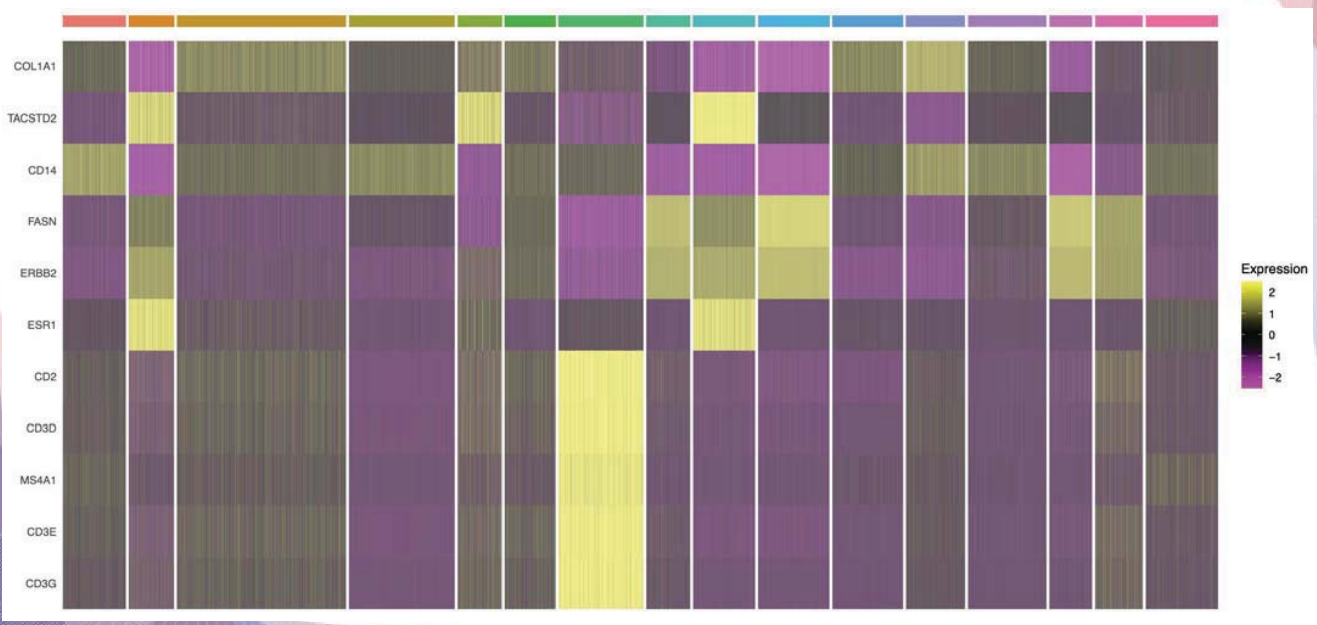
```
breast_seurat = Seurat::CreateSeuratObject(
     counts=logcounts(breast.enhanced),
     assay='Spatial',
     meta.data=as.data.frame(colData(breast.enhanced)))
breast_seurat = Seurat::SetIdent(breast_seurat, value="spatial.cluster")
breast_seurat@assays$Spatial@scale.data =
     breast_seurat@assays$Spatial@data %>%
     as.matrix %>%
     t %>%
     scale %>% t

top_markers = Seurat::FindAllMarkers(breast_seurat, assay='Spatial',
                                slot='data',
                                group.by='spatial.cluster',
                                only.pos=TRUE) %>%     group_by(cluster) %>%
     top_n(5, avg_log2FC)

head(top_markers,2)
```



```
> head(top_markers,2)
# A tibble: 2 × 7
# Groups:   cluster [2]
  p_val avg_log2FC pct.1 pct.2 p_val_adj cluster gene
  <dbl>      <dbl> <dbl> <dbl>     <dbl> <fct>   <chr>
1     0      0.261     1 1             0 3       COL1A1
2     0      0.754     1 0.996         0 5       TACSTD2
```

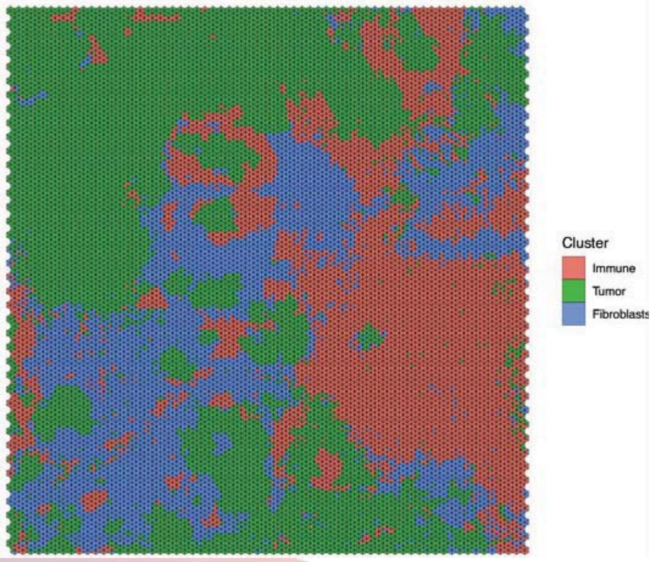
Visualize marker genes expression with heatmap

```
Seurat::DoHeatmap(breast_seurat,
                  features = top_markers$gene,    slot="scale.data",
                  group.by = "spatial.cluster", angle=0, label = FALSE,
                raster=FALSE) + guides(col = FALSE)
```



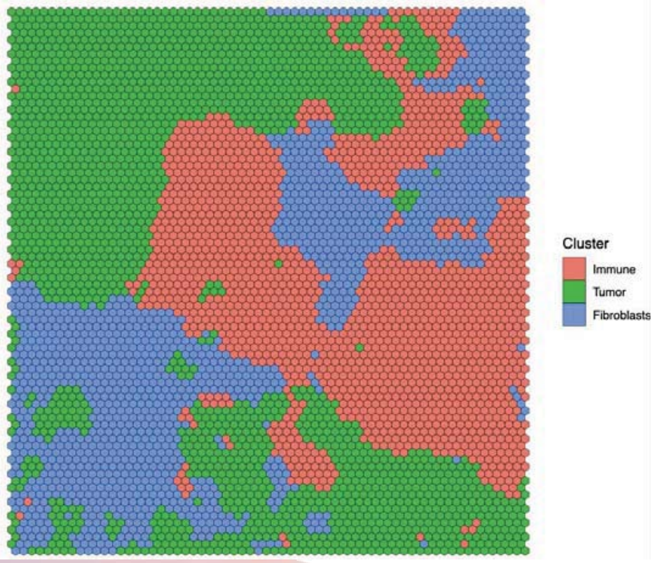
Annotate spots for enhanced object

```
coldata = colData(breast.enhanced)
coldata$spatial.cluster.annotation =
     ifelse(coldata$spatial.cluster %in% c(1, 4, 7, 13), "Immune",
     ifelse(coldata$spatial.cluster %in%
     c(2, 5, 6, 8, 9, 10, 14, 15, 16), "Tumor", "Fibroblasts"))
coldata$spatial.cluster.annotation =    factor(coldata$spatial.cluster.annotation,
     levels = c("Immune","Tumor","Fibroblasts"))
colData(breast.enhanced)= coldata
clusterPlot(breast.enhanced, color="black", size=0.1,
             label="spatial.cluster.annotation") + labs(title="Annotation")
```



Annotate spots for breast_sce object

```
coldata = colData(breast_sce)
coldata$spatial.cluster.annotation =
     ifelse(coldata$spatial.cluster %in% c(1, 4, 7, 13), "Immune",
     ifelse(coldata$spatial.cluster %in%
     c(2, 5, 6, 8, 9, 10, 14, 15, 16), "Tumor", "Fibroblasts"))
coldata$spatial.cluster.annotation =    factor(coldata$spatial.cluster.annotation,
     levels = c("Immune","Tumor","Fibroblasts"))
colData(breast_sce)= coldata
clusterPlot(breast_sce, color="black", size=0.1,
             label="spatial.cluster.annotation") + labs(title="Annotation")
```

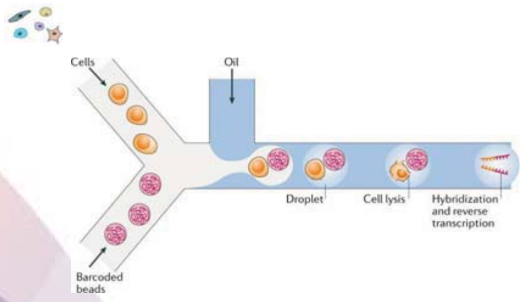


4. Cell Ranger

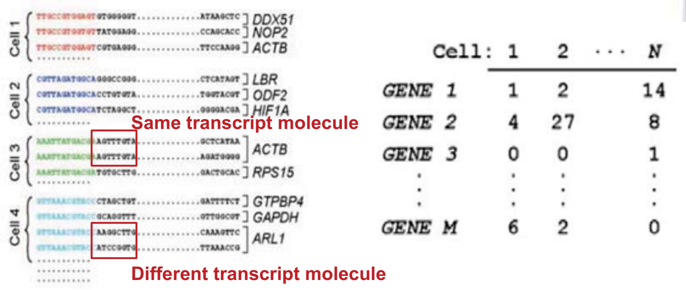
What is Cell Ranger?

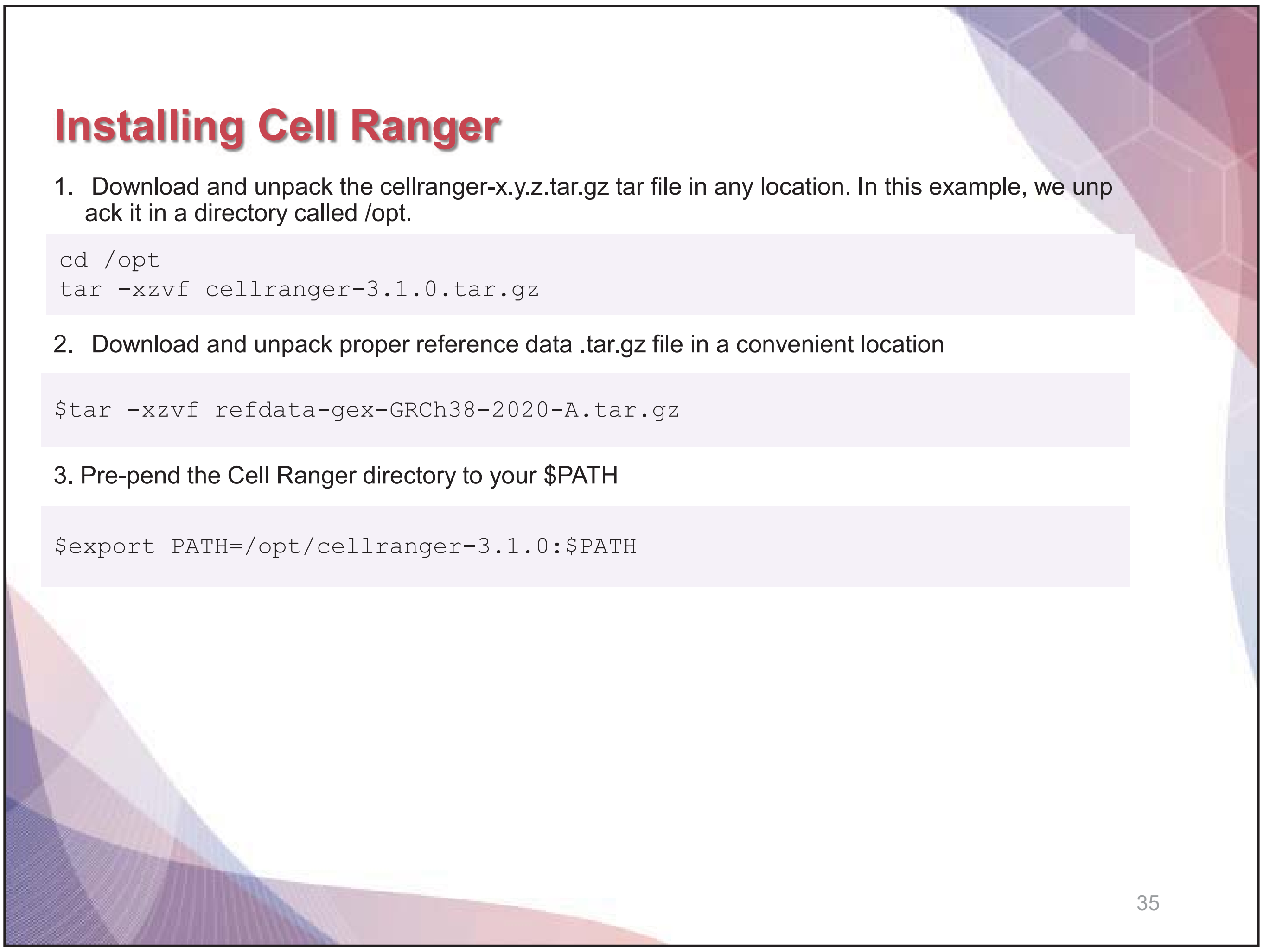
- Cell Ranger is a set of analysis pipelines that process single cell data to align reads, generate feature-ba rcode matrices, perform clustering and other secondary analysis.
- We will introduce **cellranger count** pipeline among the 5 pipelines

Cell isolation



cellranger count





Installing Cell Ranger

1. Download and unpack the cellranger-x.y.z.tar.gz tar file in any location. In this example, we unp ack it in a directory called /opt.

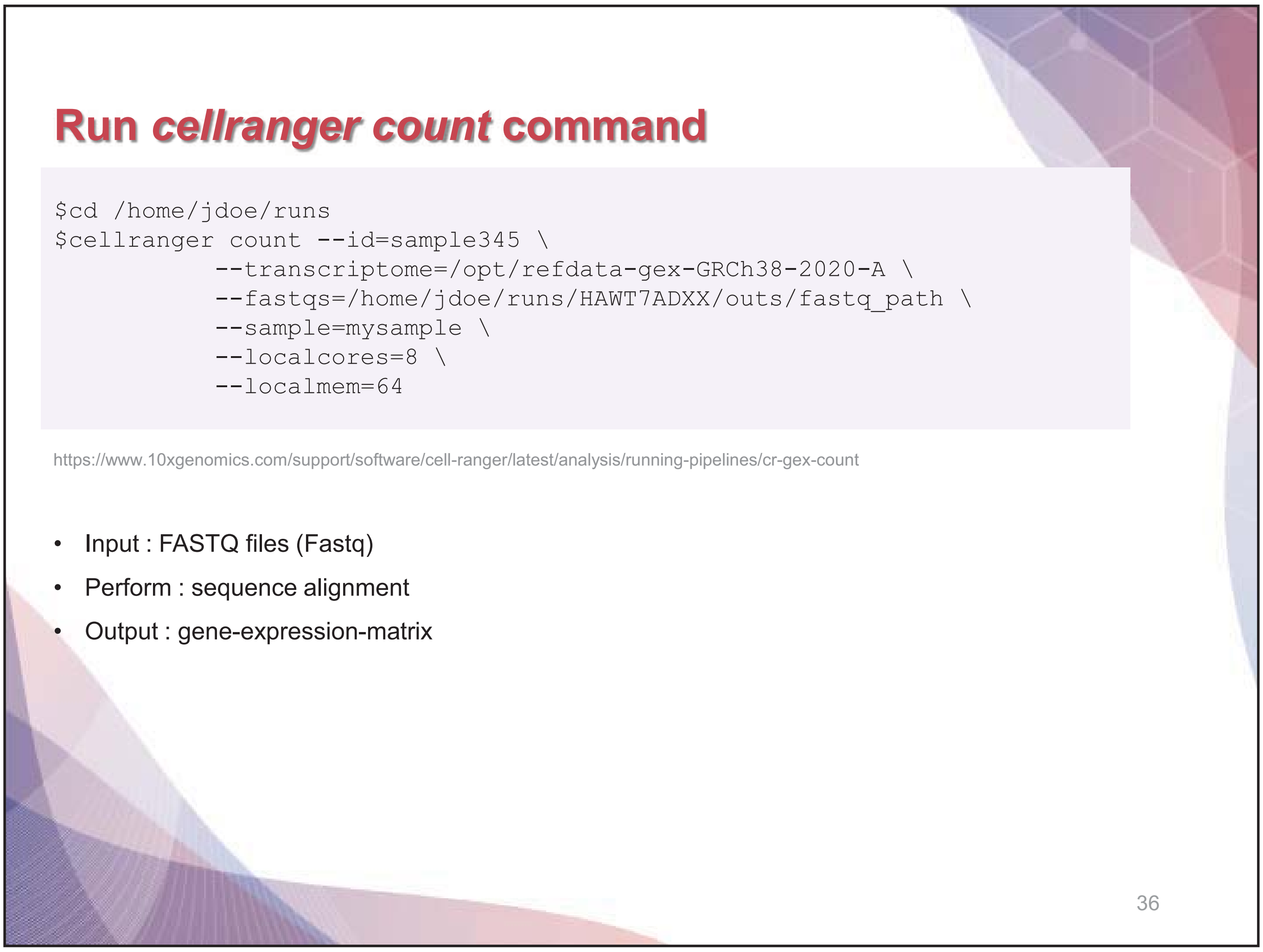
```
cd /opt
tar -xzvf cellranger-3.1.0.tar.gz
```

2. Download and unpack proper reference data .tar.gz file in a convenient location

```
$tar -xzvf refdata-gex-GRCh38-2020-A.tar.gz
```

3. Pre-pend the Cell Ranger directory to your $PATH

```
$export PATH=/opt/cellranger-3.1.0:$PATH
```



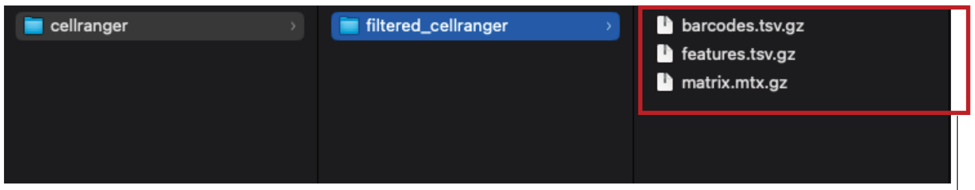
Run *cellranger count* command

```
$cd /home/jdoe/runs
$cellranger count --id=sample345 \
           --transcriptome=/opt/refdata-gex-GRCh38-2020-A \
           --fastqs=/home/jdoe/runs/HAWT7ADXX/outs/fastq_path \
           --sample=mysample \
           --localcores=8 \
           --localmem=64
```

https://www.10xgenomics.com/support/software/cell-ranger/latest/analysis/running-pipelines/cr-gex-count

- Input : FASTQ files (Fastq)
- Perform : sequence alignment
- Output : gene-expression-matrix

Output files of Cell Ranger



read10X Input files

37

Output files

- **raw_feature_bc_matrix**

  Contains all detected barcodes in MEX format. Each element of the matrix is the number of UMIs associated with a feature (row) and a barcode (column).

- **filtered_feature_bc_matrix**

  The filtered gene-barcode matrix excludes barcodes that correspond to background noise.

- **possorted_genome_bam.bam**

  Indexed BAM file containing position-sorted reads aligned to the genome and transcriptome, as well as unaligned reads, annotated with barcode information.

- **web_summary.html**

  Interactive website

38

5. scRNA-seq pre-processing

Load packages in R environment

```
library(Seurat)
library(scDblFinder)
library(dplyr)
library(ggplot2)
```

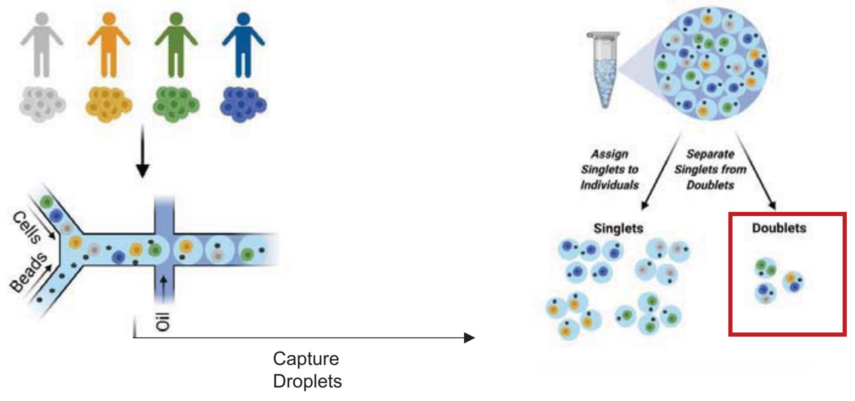
Load a 10X dataset in R

# Dropbox를 다운 받은 후에, 해당 폴더가 있는 경로로 설정합니다.

```
setwd("~/Downloads/BIML2024") # MAC
setwd("/Users/LG/Downloads/BIML2024") # Windows

dir <- c("SC3","SC5")
Breast_sc <- Read10X(data.dir = dir)
Breast_sc <- CreateSeuratObject(counts = Breast_sc, project =
"Breast_sc", min.cells = 3, min.features = 200)
```

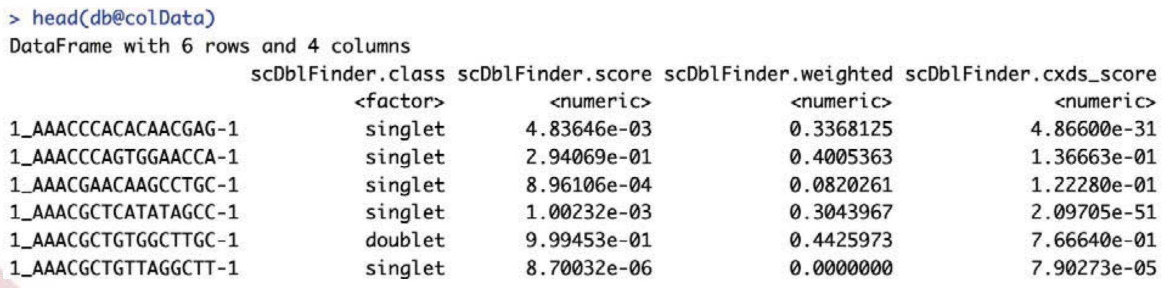
scDblFinder

```
set.seed(123)

doublet_ratio <- ncol(Breast_sc@assays$RNA$counts)/1000*0.008

db <- scDblFinder(Breast_sc@assays$RNA$counts, dbr = doublet_ratio)
```


Cells
Beads
Oil
Capture Droplets
Assign Singlets to Individuals
Separate Singlets from Doublets
Singlets
Doublets

scDblFinder

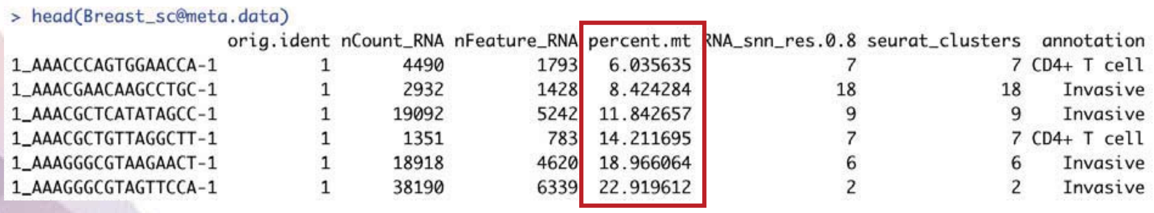
```
head(db@colData)
```



```
> head(db@colData)
DataFrame with 6 rows and 4 columns
                      scDblFinder.class scDblFinder.score scDblFinder.weighted scDblFinder.cxds_score
                               <factor>         <numeric>            <numeric>              <numeric>
1_AAACCCACACAACGAG-1            singlet       4.83646e-03            0.3368125            4.86600e-31
1_AAACCCAGTGGAACCA-1            singlet       2.94069e-01            0.4005363            1.36663e-01
1_AAACGAACAAGCCTGC-1            singlet       8.96106e-04            0.0820261            1.22280e-01
1_AAACGCTCATATAGCC-1            singlet       1.00232e-03            0.3043967            2.09705e-51
1_AAACGCTGTGGCTTGC-1            doublet       9.99453e-01            0.4425973            7.66640e-01
1_AAACGCTGTTAGGCTT-1            singlet       8.70032e-06            0.0000000            7.90273e-05
```

Calculate percent-mt of singlet

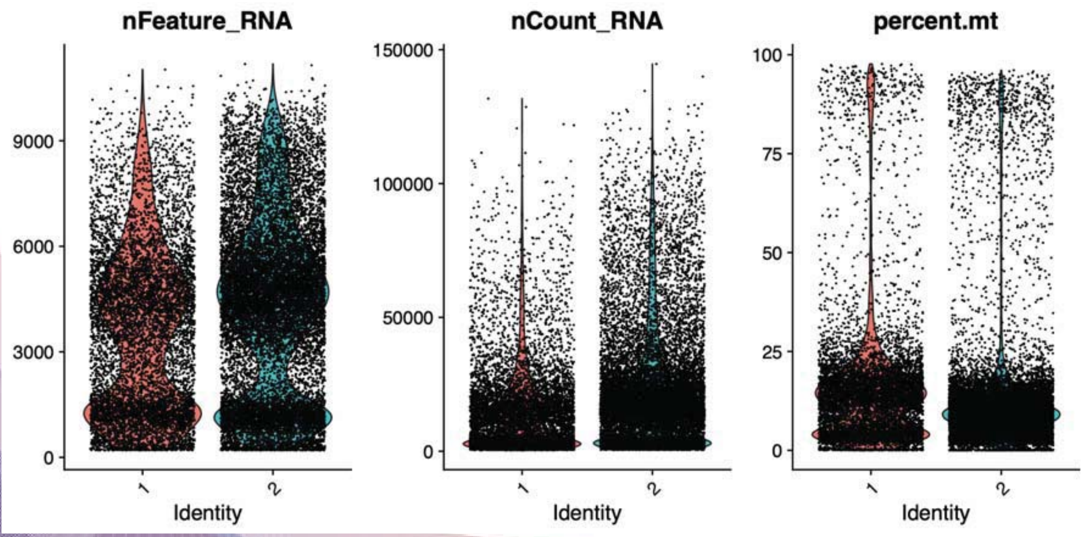
```
doublet <- row.names(db@colData)[which(db@colData$scDblFinder.class ==
'doublet')]

Breast_sc <- subset(Breast_sc, cells = doublet, invert = T)

Breast_sc[["percent.mt"]] <- PercentageFeatureSet(object =  Breast_sc,
pattern = "^MT-")

head(Breast_sc@meta.data)
```



```
> head(Breast_sc@meta.data)
                     orig.ident nCount_RNA nFeature_RNA percent.mt RNA_snn_res.0.8 seurat_clusters  annotation
1_AAACCCAGTGGAACCA-1          1       4490         1793   6.035635               7               7 CD4+ T cell
1_AAACGAACAAGCCTGC-1          1       2932         1428   8.424284              18              18    Invasive
1_AAACGCTCATATAGCC-1          1      19092         5242  11.842657               9               9    Invasive
1_AAACGCTGTTAGGCTT-1          1       1351          783  14.211695               7               7 CD4+ T cell
1_AAAGGGCGTAAGAACT-1          1      18918         4620  18.966064               6               6    Invasive
1_AAAGGGCGTAGTTCCA-1          1      38190         6339  22.919612               2               2    Invasive
```

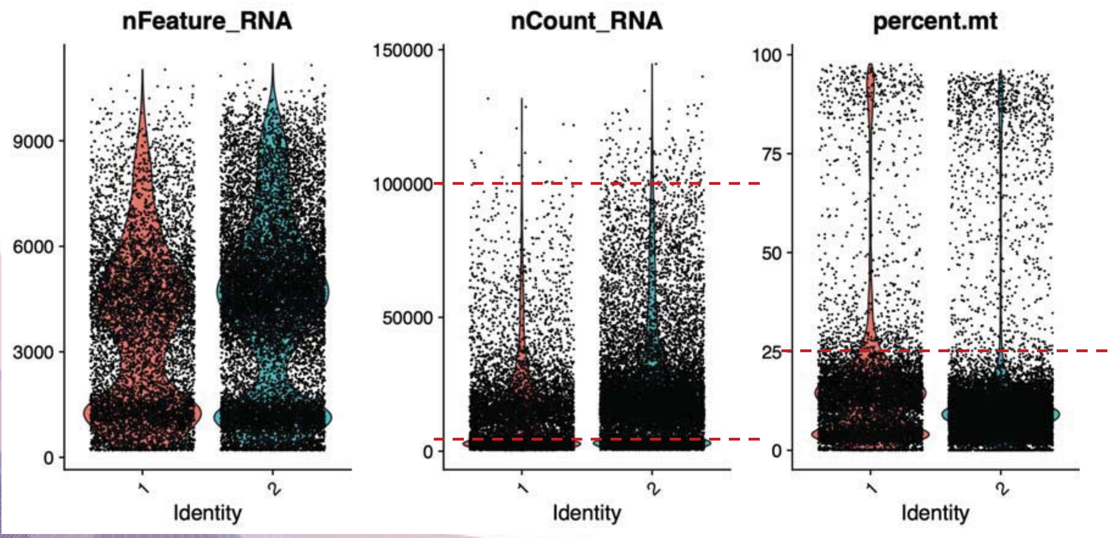
Doublets in plot

```
VlnPlot(Breast_sc,features =
c("nFeature_RNA","nCount_RNA","percent.mt"),ncol=3)
```



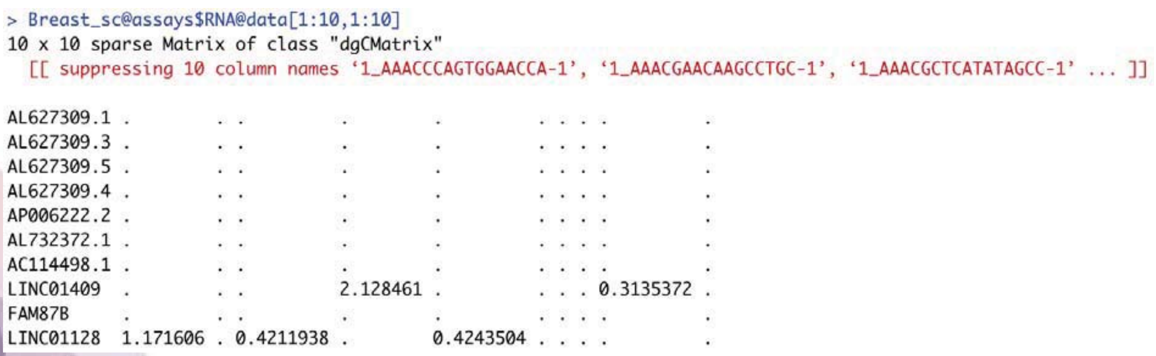
Quality control

```
Breast_sc <- subset(Breast_sc, subset = nCount_RNA > 500 & nCount_RNA
< 100000 & percent.mt < 25)
```



Log normalization

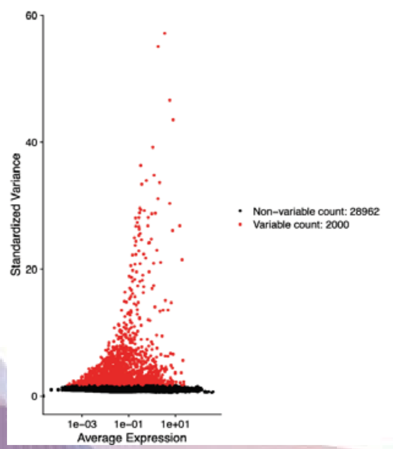
```
Breast_sc <- NormalizeData(Breast_sc, normalization.method =
"LogNormalize", scale.factor = 10000)

Breast_sc@assays$RNA@data[1:10,1:10]
```

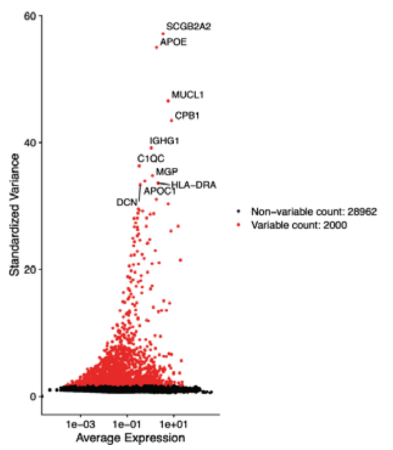


```
> Breast_sc@assays$RNA@data[1:10,1:10]
10 x 10 sparse Matrix of class "dgCMatrix"
  [[ suppressing 10 column names ‘1_AAACCCAGTGGAACCA-1’, ‘1_AAACGAACAAGCCTGC-1’, ‘1_AAACGCTCATATAGCC-1’ ... ]]

AL627309.1 .        . .         .        .         . . . .         .
AL627309.3 .        . .         .        .         . . . .         .
AL627309.5 .        . .         .        .         . . . .         .
AL627309.4 .        . .         .        .         . . . .         .
AP006222.2 .        . .         .        .         . . . .         .
AL732372.1 .        . .         .        .         . . . .         .
AC114498.1 .        . .         .        .         . . . .         .
LINC01409  .        . .         2.128461 .         . . . 0.3135372 .
FAM87B     .        . .         .        .         . . . .         .
LINC01128  1.171606 . 0.4211938 .        0.4243504 . . . .         .
```

Feature selection

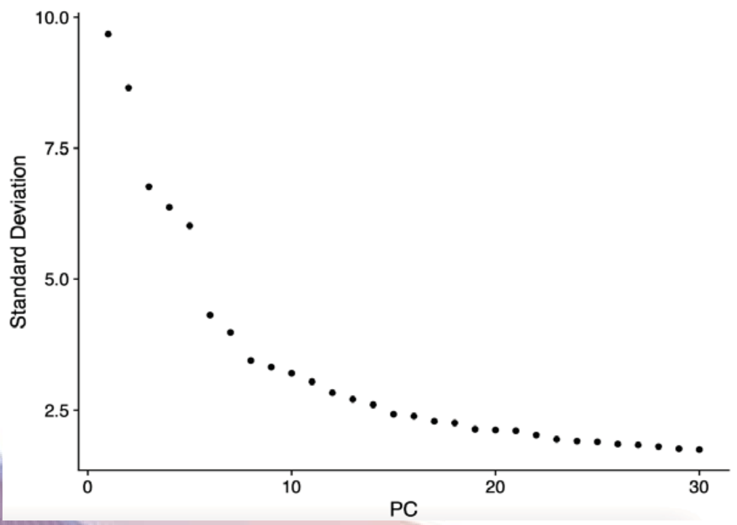
```
Breast_sc <- FindVariableFeatures(Breast_sc, selection.method = "vst",
nfeatures = 2000)

top10 <- head(VariableFeatures(Breast_sc), 10)

Plot1 <- VariableFeaturePlot(Breast_sc)
plot2 <- LabelPoints(plot = plot1, points = top10, repel = TRUE)
```





Dimension reduction

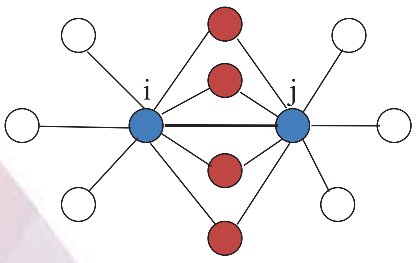
```
all.genes <- rownames(Breast_sc)

Breast_sc <- ScaleData(Breast_sc, features = all.genes)

Breast_sc <- RunPCA(Breast_sc, features = VariableFeatures(object =
Breast_sc))

ElbowPlot(Breast_sc, 30)
```

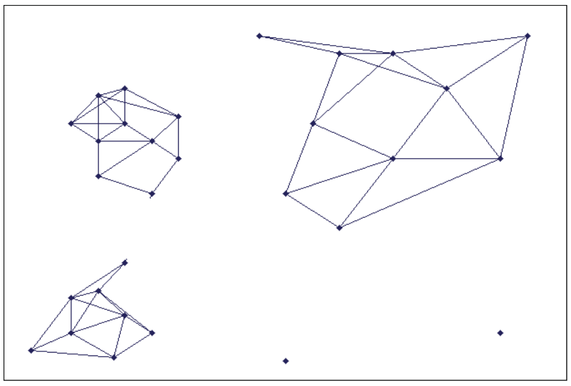


Clustering

```
Breast_sc <- FindNeighbors(Breast_sc, dims = 1:30)

Breast_sc <- FindClusters(Breast_sc, resolution = 0.8)
```

SNN graph: the weight of an edge is the number of shared neighbors between vertices given that the vertices are connected





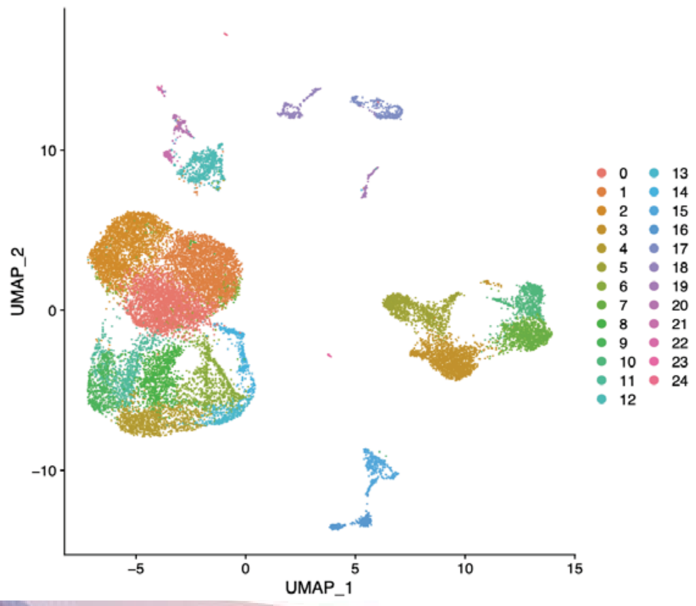
Shared Near Neighbor Graph

Link weights are number of Shared Nearest Neighbors

https://web.iitd.ac.in/~bspanda/graphclustering.pdf

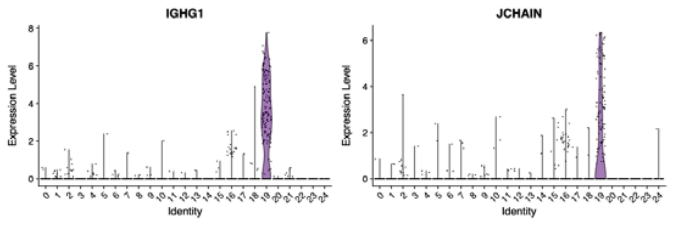
Visualization of clustering result

```
Breast_sc <- RunUMAP(Breast_sc, dims = 1:30)

DimPlot(Breast_sc)
```

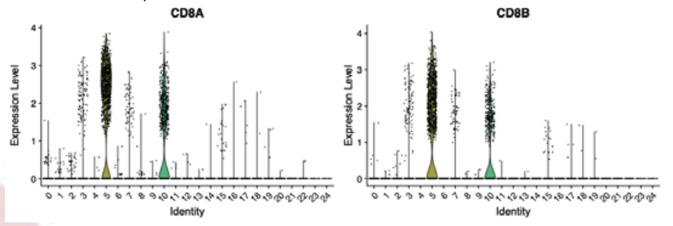


Finding differentially expressed genes

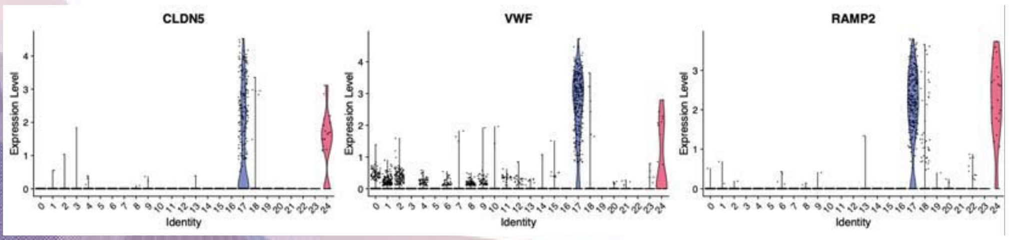
- "IGHG1", "JCHAIN" # Plasma - Cluster 19



- "CD8A","CD8B" # CD8+ T cell – Cluster 5 &10



- "CLDN5","VWF","RAMP2" # Endothelial– Cluster 17 &24



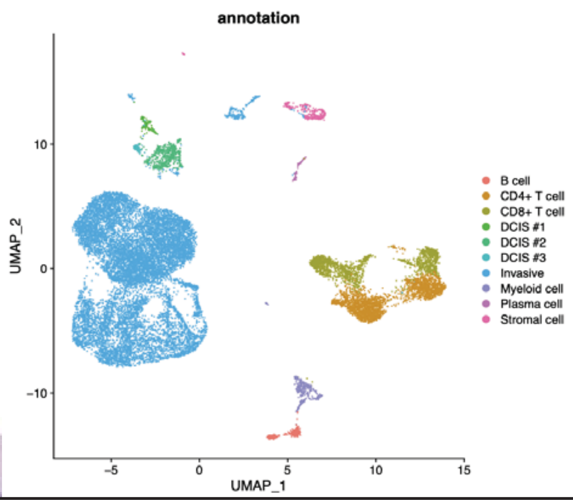
Annotation

```
Breast_sc_meta <- Breast_sc@meta.data
Breast_sc_meta <- Breast_sc_meta %>% mutate(
  annotation = ifelse(RNA_snn_res.0.8 %in% c(3, 7), c("CD4+ T cell"),
               ifelse(RNA_snn_res.0.8 %in% c(5, 10), c("CD8+ T cell"),
               ifelse(RNA_snn_res.0.8 == c(16), c("B cell"),
               ifelse(RNA_snn_res.0.8 %in% c(15, 23), c("Myeloid cell"),
               ifelse(RNA_snn_res.0.8 == c(19), c("Plasma cell"),
               ifelse(RNA_snn_res.0.8 %in% c(17, 24), c("Stromal cell"),
               ifelse(RNA_snn_res.0.8 %in% c(0, 1, 2, 4, 6, 8, 9, 11, 13,
14, 18, 22), c("Invasive"),
               ifelse(RNA_snn_res.0.8 == c(20), c("DCIS #1"),
               ifelse(RNA_snn_res.0.8 == c(12), c("DCIS #2"),
               ifelse(RNA_snn_res.0.8 == c(21), c("DCIS #3"),
               "Others"))))))))))
```

Annotation visualization

```
table(Breast_sc_meta$annotation)

Breast_sc <- AddMetaData(Breast_sc, Breast_sc_meta)

Breast_sc = SetIdent(Breast_sc, value = "annotation")

Breast_sc$annotation <- factor(Breast_sc$annotation)
```

```
> table(her2bc_meta$annotation)

      B cell  CD4+ T cell  CD8+ T cell      DCIS #1      DCIS #2      DCIS #3     Invasive Myeloid cell  Plasma cell Stromal cell
         341         2566         1760          166          677           97        14215          403          168          351
```



Load a Visium dataset in R

## renamed CytAssist_FFPE_Human_Breast_Cancer_filtered_feature_bc_matrix.h5 into filtered_feature_bc_matrix.h5

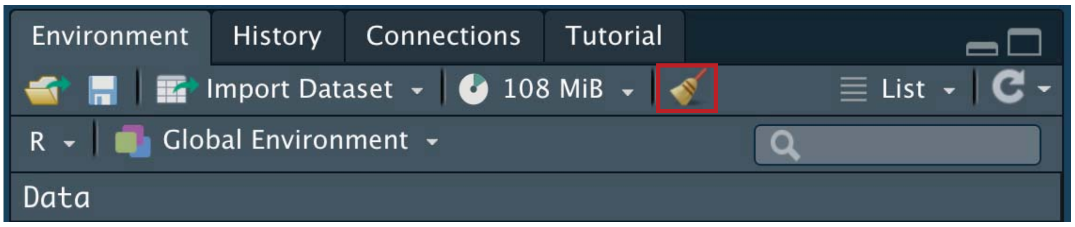
```
breast_visium = Load10X_Spatial("./Raw_file/visium/",
                 filename = "filtered_feature_bc_matrix.h5")
```

Preprocessing of Visium

```
breast_visium = SCTransform(breast_visium, assay = "Spatial",
             verbose = FALSE)
breast_visium = RunPCA(breast_visium, assay = "SCT", verbose = FALSE)
breast_visium = FindNeighbors(breast_visium, reduction = "pca",
             dims = 1:30)
breast_visium = FindClusters(breast_visium, verbose = FALSE)
breast_visium = RunUMAP(breast_visium, reduction = "pca",
             dims = 1:30)

# saveRDS(breast_visium, "./object/Biml2024_Breast_singlecell.rds")
```

Remove all objects before starting next chapter

- Clear objects from the workspace.



# clean up memory in R

```
gc()
```

6. Deconvolution Analysis - RCTD

Spatial Transcriptomics Analysis

Purpose and limitation	
Purpose	Discovery of cell-type-specific spatial patterns of localization and expression.
Limitation	Individual measurements may contain contributions from multiple cells in Visium.

Robust Cell Type Decomposition (RCTD) process

[1] Spatial mapping

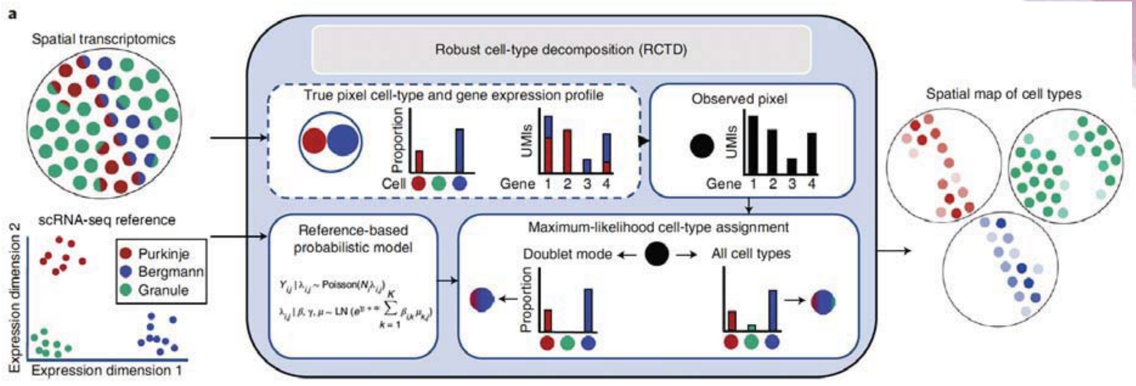
Spot1	CD8+ T Cell
Spot2	Invasive
Spot3	Myeloid cell
Spot4	Invasive
	…

[2] Spatial deconvolution

	Invasive	Myeloid cell	CD4+ T Cell	CD8+ T Cell	…
Spot1	0.4	0.1	0.04	0.3	…
Spot2	0.8	0.01	0.1	0.02	…
Spot3	0.01	0.88	0.01	0.08	…
Spot4	0.02	0.2	0.6	0.12	…
	0.3	0.4	0.25	0.01	…

Analysis types	
Spatial Mapping	One representative cell type is assigned to individual spot. ***-> For cell-cell interaction downstream analysis***
Spatial deconvolution	Proportions of multiple cell types is assigned to individual spot.

RCTD algorithm


a
Spatial transcriptomics
Robust cell-type decomposition (RCTD)
True pixel cell-type and gene expression profile
Proportion
Cell
UMIs
Gene 1 2 3 4
Observed pixel
Spatial map of cell types
scRNA-seq reference
Purkinje
Bergmann
Granule
Expression dimension 2
Expression dimension 1
Reference-based probabilistic model
Maximum-likelihood cell-type assignment
Doublet mode
All cell types

RCTD process	
1	Calculate **single-cell reference** cell type means
2	Creates a spatial map of cell types by fitting each **spatial transcriptomics pixel** as a linear combination of individ ual cell types
3	Fitting a **statistical model** where the observed gene counts Yi,j (pixel i and gene j) are assumed to be Poisson dis tributed
4	To account for **platform effects**, we assume that λi,j is a random variable
5	**Maximum-likelihood estimation (MLE)** to infer the cell type proportions, βi,k

RCTD model

$$Y_{i,j} | \lambda_{i,j} \sim \text{Poisson}\left(N_i \lambda_{i,j}\right)$$

$$\log\left(\lambda_{i,j}\right) = \alpha_i + \log\left(\sum_{k=1}^{K} \beta_{i,k}\mu_{k,j}\right) + \gamma_j + \varepsilon_{i,j}$$

Model counts with hierarchical model (Pixel I, Cell type K, Gene j)	
Yi,j	The observed gene expression **counts**
λi,j	Random variable to account for **platform effects**
μk,j	The **mean gene expression** profile for cell type k
γ_j	A gene-specific **platform random effect**
εi,j	A random effect to account for **gene-specific overdispersion.**
αi	A fixed **pixel-specific effect**
Goal	**Estimate the βi,k's, which represent the cell type or cell types present in each pixel i**

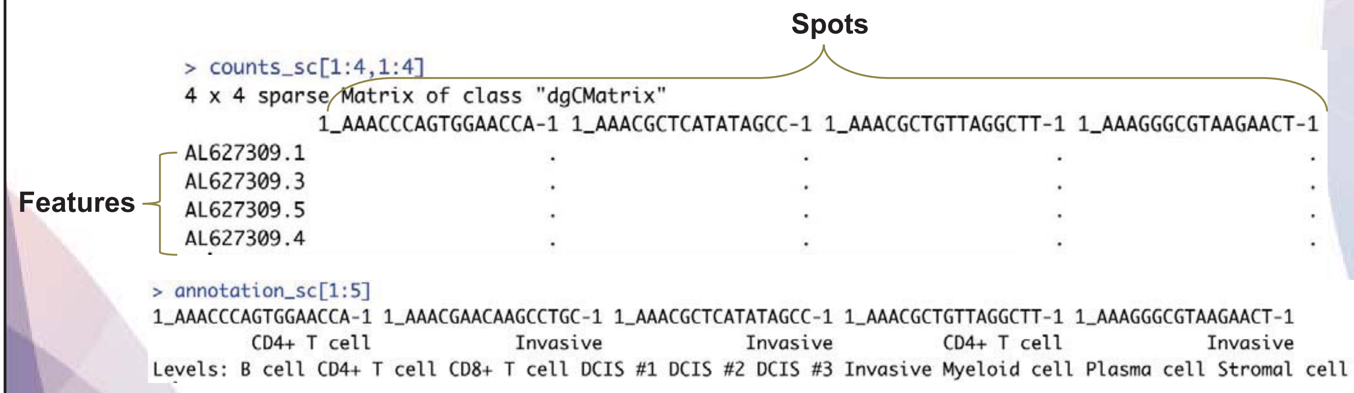
Load RCTD input reference dataset (single cell)

# Dropbox를 다운 받은 후에, 해당 폴더가 있는 경로로 설정합니다.

```
setwd("~/Downloads/BIML2024")  # MAC
setwd("/Users/LG/Downloads/BIML2024")  # Windows

Breast_sc <- readRDS("./object/Biml2024_Breast_singlecell.rds")
```

Prepare single cell dataset for RCTD input

```
counts_sc = Breast_sc$RNA@counts

annotation_sc  = Breast_sc$annotation
names(annotation_sc) = rownames(Breast_sc@meta.data)
annotation_sc = as.factor(annotation_sc)
```

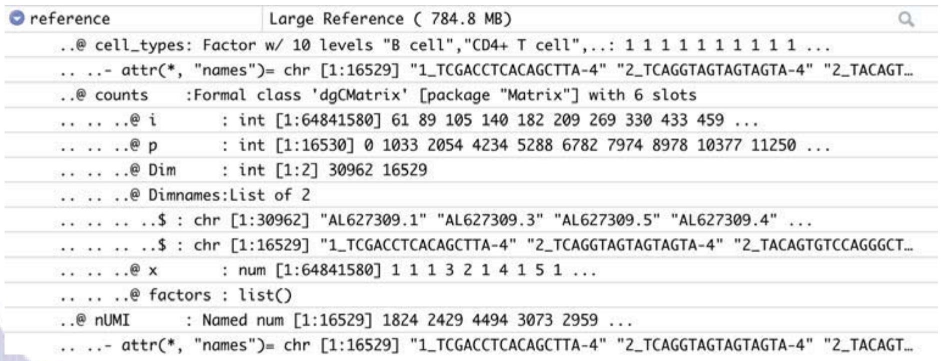


```
> counts_sc[1:4,1:4]
4 x 4 sparse Matrix of class "dgCMatrix"
           1_AAACCCAGTGGAACCA-1 1_AAACGCTCATATAGCC-1 1_AAACGCTGTTAGGCTT-1 1_AAAGGGCGTAAGAACT-1
AL627309.1                    .                    .                    .                    .
AL627309.3                    .                    .                    .                    .
AL627309.5                    .                    .                    .                    .
AL627309.4                    .                    .                    .                    .

> annotation_sc[1:5]
1_AAACCCAGTGGAACCA-1 1_AAACGAACAAGCCTGC-1 1_AAACGCTCATATAGCC-1 1_AAACGCTGTTAGGCTT-1 1_AAAGGGCGTAAGAACT-1
         CD4+ T cell             Invasive             Invasive          CD4+ T cell             Invasive
Levels: B cell CD4+ T cell CD8+ T cell DCIS #1 DCIS #2 DCIS #3 Invasive Myeloid cell Plasma cell Stromal cell
```

Prepare single cell dataset for RCTD input

```
nUMI_sc = Breast_sc@meta.data$nCount_RNA
names(nUMI_sc) = rownames(Breast_sc@meta.data)

reference = Reference(counts_sc, annotation_sc, nUMI_sc)
gc()
```

```
> nUMI_sc[1:5]
1_AAACCCAGTGGAACCA-1 1_AAACGAACAAGCCTGC-1 1_AAACGCTCATATAGCC-1 1_AAACGCTGTTAGGCTT-1 1_AAAGGGCGTAAGAACT-1
                4490                 2932                19092                 1351                18918
```



```
reference              Large Reference ( 784.8 MB)
  ..@ cell_types: Factor w/ 10 levels "B cell","CD4+ T cell",..: 1 1 1 1 1 1 1 1 1 1 ...
  .. ..- attr(*, "names")= chr [1:16529] "1_TCGACCTCACAGCTTA-4" "2_TCAGGTAGTAGTAGTA-4" "2_TACAGT…
  ..@ counts    :Formal class 'dgCMatrix' [package "Matrix"] with 6 slots
  .. .. ..@ i        : int [1:64841580] 61 89 105 140 182 209 269 330 433 459 ...
  .. .. ..@ p        : int [1:16530] 0 1033 2054 4234 5288 6782 7974 8978 10377 11250 ...
  .. .. ..@ Dim      : int [1:2] 30962 16529
  .. .. ..@ Dimnames:List of 2
  .. .. .. ..$ : chr [1:30962] "AL627309.1" "AL627309.3" "AL627309.5" "AL627309.4" ...
  .. .. .. ..$ : chr [1:16529] "1_TCGACCTCACAGCTTA-4" "2_TCAGGTAGTAGTAGTA-4" "2_TACAGTGTCCAGGGCT…
  .. .. ..@ x        : num [1:64841580] 1 1 1 3 2 1 4 1 5 1 ...
  .. .. ..@ factors : list()
  ..@ nUMI      : Named num [1:16529] 1824 2429 4494 3073 2959 ...
  .. ..- attr(*, "names")= chr [1:16529] "1_TCGACCTCACAGCTTA-4" "2_TCAGGTAGTAGTAGTA-4" "2_TACAGT…
```

Process spatial dataset for RCTD input

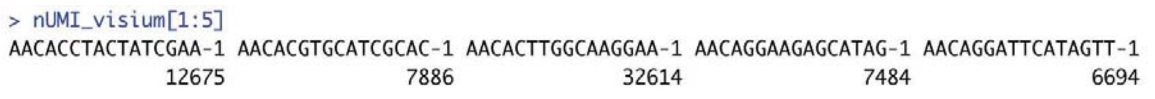
```
breast_visium = readRDS("./object/Biml2024_Breast_visium.rds")

coords_visium = breast_visium@images$slice1@coordinates[,c("col","row")]

counts_visium = breast_visium@assays$Spatial@counts
```

```
> head(breast_visium@images$slice1@coordinates)
                   tissue row col imagerow imagecol
AACACCTACTATCGAA-1      1   0 122     4636     4131
AACACGTGCATCGCAC-1      1  76  22    16640    13355
AACACTTGGCAAGGAA-1      1  47  71    12067     8845
AACAGGAAGAGCATAG-1      1  69   7    15518    14716
AACAGGATTCATAGTT-1      1  49  43    12365    11404
AACAGGCCAACGATTA-1      1  71 127    15920     3761
```

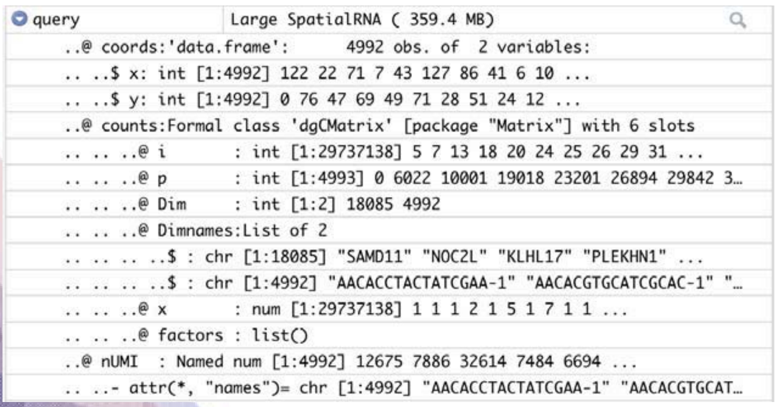
```
> counts_visium[1:4,1:4]
4 x 4 sparse Matrix of class "dgCMatrix"
        AACACCTACTATCGAA-1 AACACGTGCATCGCAC-1 AACACTTGGCAAGGAA-1 AACAGGAAGAGCATAG-1
SAMD11                   .                  2                  1                  .
NOC2L                    .                  .                  3                  .
KLHL17                   .                  .                  .                  .
PLEKHN1                  .                  .                  1                  .
```

Spatial Transcriptomics Analysis

```
nUMI_visium = colSums(counts_visium)

query = SpatialRNA(coords_visium, counts_visium, nUMI_visium)
gc()
```



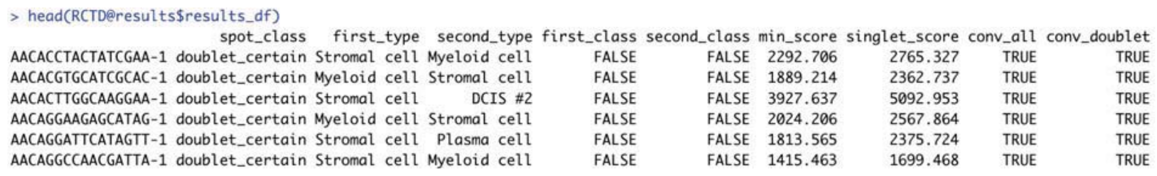
```
> nUMI_visium[1:5]
AACACCTACTATCGAA-1 AACACGTGCATCGCAC-1 AACACTTGGCAAGGAA-1 AACAGGAAGAGCATAG-1 AACAGGATTCATAGTT-1
             12675               7886              32614               7484               6694
```



```
query                    Large SpatialRNA ( 359.4 MB)
  ..@ coords:'data.frame':      4992 obs. of  2 variables:
  .. ..$ x: int [1:4992] 122 22 71 7 43 127 86 41 6 10 ...
  .. ..$ y: int [1:4992] 0 76 47 69 49 71 28 51 24 12 ...
  ..@ counts:Formal class 'dgCMatrix' [package "Matrix"] with 6 slots
  .. .. ..@ i        : int [1:29737138] 5 7 13 18 20 24 25 26 29 31 ...
  .. .. ..@ p        : int [1:4993] 0 6022 10001 19018 23201 26894 29842 3…
  .. .. ..@ Dim      : int [1:2] 18085 4992
  .. .. ..@ Dimnames:List of 2
  .. .. .. ..$ : chr [1:18085] "SAMD11" "NOC2L" "KLHL17" "PLEKHN1" ...
  .. .. .. ..$ : chr [1:4992] "AACACCTACTATCGAA-1" "AACACGTGCATCGCAC-1" "…
  .. .. ..@ x        : num [1:29737138] 1 1 1 2 1 5 1 7 1 1 ...
  .. .. ..@ factors : list()
  ..@ nUMI  : Named num [1:4992] 12675 7886 32614 7484 6694 ...
  .. ..- attr(*, "names")= chr [1:4992] "AACACCTACTATCGAA-1" "AACACGTGCAT…
```

Run RCTD in doublet mode

```
RCTD = create.RCTD(query, reference, max_cores = 8)
# RCTD = run.RCTD(RCTD, doublet_mode = 'doublet')
RCTD = readRDS("./Biml2024_Breast_RCTD.rds")
RCTD_results = RCTD@results$results_df
breast_visium = AddMetaData(breast_visium, metadata = RCTD_results)
```

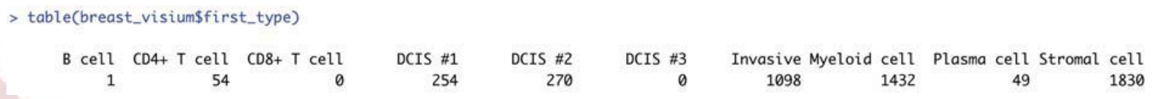


```
> head(RCTD@results$results_df)
                      spot_class   first_type  second_type first_class second_class min_score singlet_score conv_all conv_doublet
AACACCTACTATCGAA-1 doublet_certain Stromal cell Myeloid cell       FALSE        FALSE  2292.706      2765.327     TRUE         TRUE
AACACGTGCATCGCAC-1 doublet_certain Myeloid cell Stromal cell       FALSE        FALSE  1889.214      2362.737     TRUE         TRUE
AACACTTGGCAAGGAA-1 doublet_certain Stromal cell      DCIS #2       FALSE        FALSE  3927.637      5092.953     TRUE         TRUE
AACAGGAAGAGCATAG-1 doublet_certain Myeloid cell Stromal cell       FALSE        FALSE  2024.206      2567.864     TRUE         TRUE
AACAGGATTCATAGTT-1 doublet_certain Stromal cell  Plasma cell       FALSE        FALSE  1813.565      2375.724     TRUE         TRUE
AACAGGCCAACGATTA-1 doublet_certain Stromal cell Myeloid cell       FALSE        FALSE  1415.463      1699.468     TRUE         TRUE
```

Analysis mode	
Doublet	Fits at most two cell types per pixel
Full	No restrictions on number of cell types, recommended for low spatial resolution technologies such as Visium
Multi	Finitely many cell types per pixel, e.g. 3 or 4.

Process RCTD decomposed file

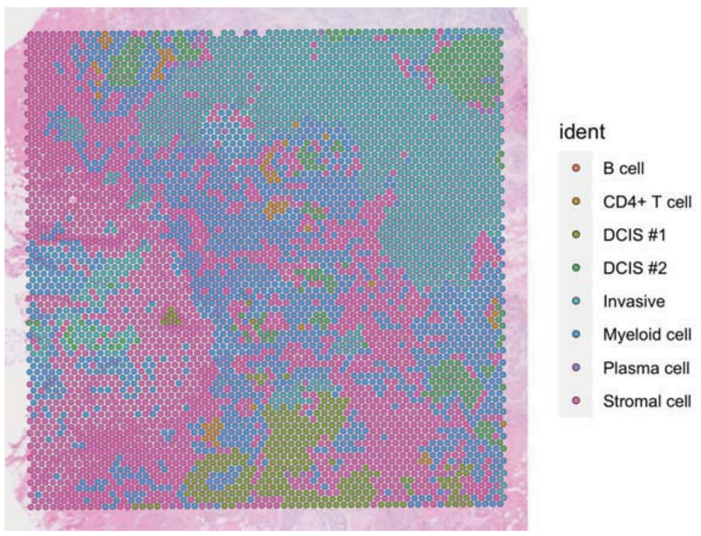
```
breast_visium = SetIdent(breast_visium, value="first_type")
table(breast_visium$first_type)
```



```
> table(breast_visium$first_type)

     B cell  CD4+ T cell  CD8+ T cell       DCIS #1       DCIS #2       DCIS #3      Invasive Myeloid cell  Plasma cell Stromal cell
          1           54            0           254           270             0          1098         1432           49         1830
```

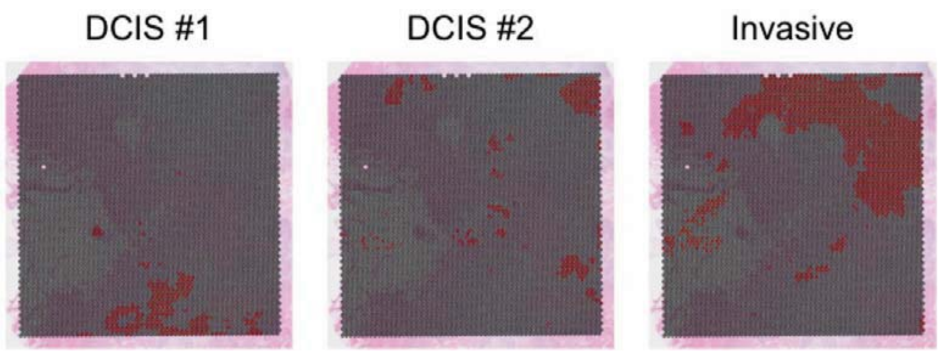
Breast spatial map of predicted cell type by RCTD

```
breast_visium <- subset(breast_visium, first_type %in%
                         names(table(breast_visium$first_type)))
breast_visium$first_type <- factor(breast_visium$first_type)
SpatialDimPlot(breast_visium)
```



Spatial map of predicted cell type by RCTD (1)

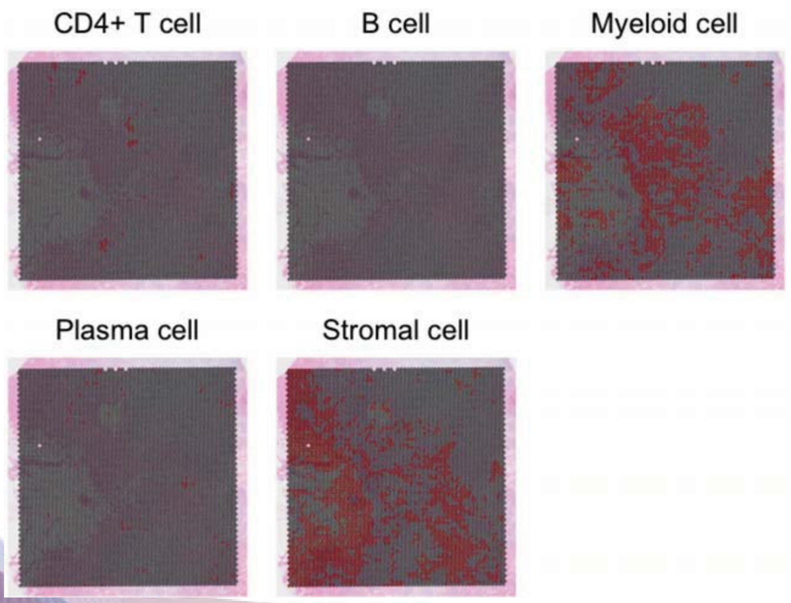
```
SpatialDimPlot(breast_visium,
     cells.highlight = CellsByIdentities(object = breast_visium,
     idents = c('DCIS #1','DCIS #2','Invasive')),
     facet.highlight = TRUE, ncol = 3)
```



Cancer types	
DCIS	low-grade and high-grade ductal carcinoma in situ
Invasive	invasive carcinoma

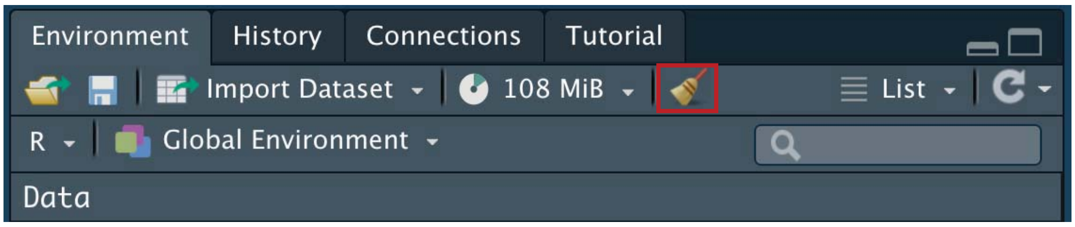
Spatial map of predicted cell type by RCTD (2)

```
SpatialDimPlot(breast_visium, cells.highlight = CellsByIdentities(object
= breast_visium,
idents = c('CD4+ T cell', 'B cell', 'Myeloid cell',  'Plasma cell',
'Stromal cell')),
facet.highlight = TRUE, ncol = 3)
```



Remove all objects before starting next chapter

- Clear objects from the workspace.



# clean up memory in R

```
gc()
```

7. Cell-cell interaction analysis - Cellchat

What is CellChat?

CellChat is a useful tool to **quantitatively infer and analyze intercellular communication networks** from single-cell RNA-sequencing data and spatial transcriptomics data.

Requires **gene expression** and **spatial location data** of spots/cells as the user input and models the proba bility of cell-cell communication by integrating gene expression with spatial distance as well as prior knowled ge of the interactions between signaling ligands, receptors and their cofactors.

JIN, Suoqin, et al. Inference and analysis of cell-cell communication using CellChat. Nature communications, 2021, 12.1: 1-20.
https://htmlpreview.github.io/?https://github.com/sqjin/CellChat/blob/master/tutorial/CellChat_analysis_of_spatial_imaging_data.html

Load data

# Load cell type annotated visium data and visualization

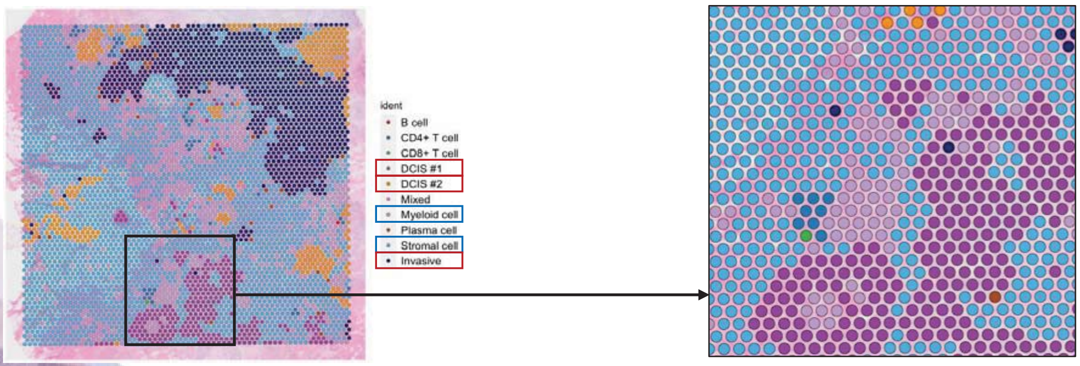
```
visium.breast = readRDS("./object/Biml2024_Breast_visium_final.rds")

visium.breast$first_type = factor(visium.breast$first_type,
          levels = c("B cell", "CD4+ T cell",  "DCIS #1", "DCIS #2",
                    "Myeloid cell", "Plasma cell", "Stromal cell", "Invasive"))
```

Load data

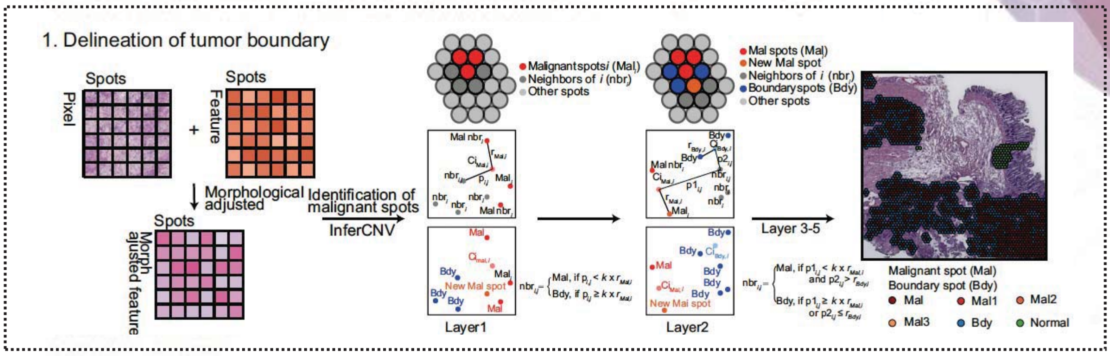
```
Idents(visium.breast) = visium.breast$first_type
colors = scPalette(nlevels(visium.breast))
names(colors) = c("B cell", "CD4+ T cell",  "DCIS #1", "DCIS #2",
                  "Myeloid cell", "Plasma cell", "Stromal cell", "Invasive")

SpatialDimPlot(visium.breast, label = F, cols = colors)
```

There are many myeloid cells and stromal cells around the tumor.

We focus on the cell-cell interaction that occurs in the **tumor layer**.



Cottrazm to delineate tumor boundary



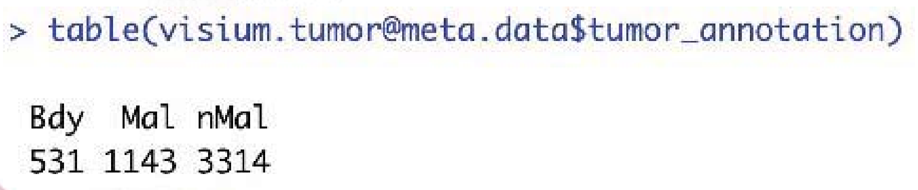
Reconstruction of the tumor spatial microenvironment along the malignant-boundary-nonmalignant axis

Cottrazm process	
1	Cottrazm initially normalizes ST gene expression data based on neighboring spot information and morphological distances of HE-staining images.
2	Immune-related signatures are used to score spots and select reference cluster.
3	InferCNV are used to assess CNV level for remained spots
4	Annotate malignant cluster
5	Find neighbor spots of tumor core
6	Decide malignant spots (Mal), boundary spot (Bdy), and non-malignant spots (nMal)

Load data

```
# Please see visium_cottrazm_script.R to run cottrazm
# Load tumor annotated visium data
visium.tumor = readRDS("./R_object/BIML2024_Breast_visium_TumorST.rds")
visium.tumor = subset(visium.tumor, nCount_Spatial > 100)
table(visium.tumor@meta.data$tumor_annotation)
```

We only need the boundary cells of the **tumor and their layers.**



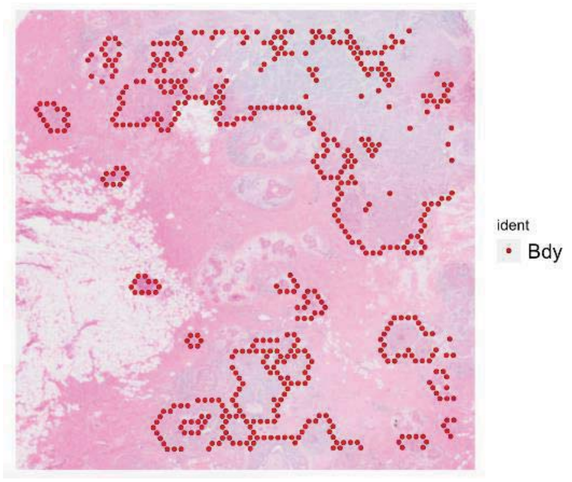
```
> table(visium.tumor@meta.data$tumor_annotation)

 Bdy  Mal nMal
 531 1143 3314
```

Load data

```
# Subset only tumor boundaries
visium.breast@meta.data$tumor_annotation =
visium.tumor@meta.data$tumor_annotation

Idents(visium.breast)= visium.breast@meta.data$tumor_annotation
visium.boundary = subset(visium.breast, idents = "Bdy")
```

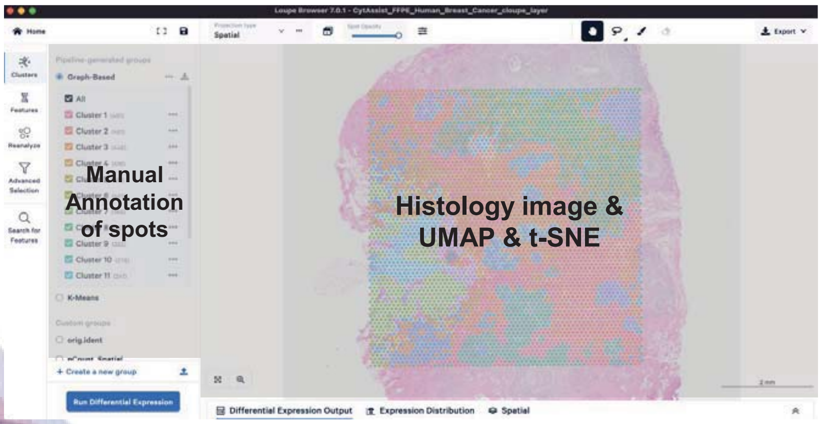
Load data

```
Idents(visium.boundary)= visium.boundary@meta.data$tumor_annotation
names(colors) = "Bdy"

SpatialDimPlot(visium.boundary, label = F, cols = colors)
```



Load data

- Loupe Browser is a desktop application from 10x Genomics that allows to visualize gene expression data without having to write code.
- Align gene expression spots to histological images, look for marker gene expression, annotate populations, and cluster.
- The .cloupe file is the one that need to import into the Loupe Browser.


Loupe Browser 7.0.1 - CytAssist_FFPE_Human_Breast_Cancer_cloupe_layer
Manual Annotation of spots
Histology image & UMAP & t-SNE
Graph-Based
All
Cluster 1
Cluster 2
Cluster 3
Cluster 9
Cluster 10
Cluster 11
K-Means
Create a new group
Run Differential Expression
Differential Expression Output
Expression Distribution
Spatial
Export

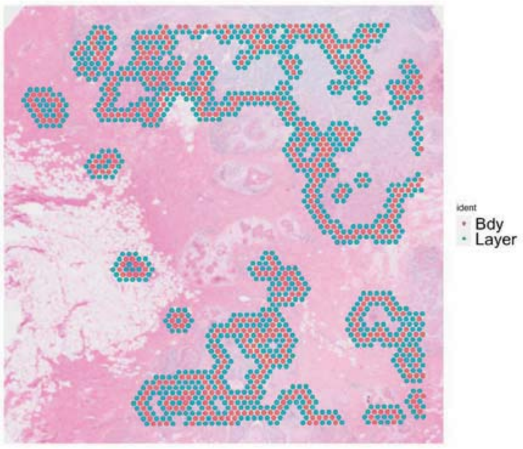
https://support.10xgenomics.com/spatial-gene-expression/software/visualization/latest/analysis

Load data

# Load layer annotated Loupe Browser data
```
loupe.data = read.csv("./Raw_file/LoupeBrowser/layer_annotation.csv",
                       header = T, row.names = 1)
table(loupe.data$layer_annotation)
```

Manually annotate the tumor layer using the Loupe browser.

```
> table(loupe.data$layer_annotation)

 Bdy Layer   Mal  nMal
 531   988   586  2883
```



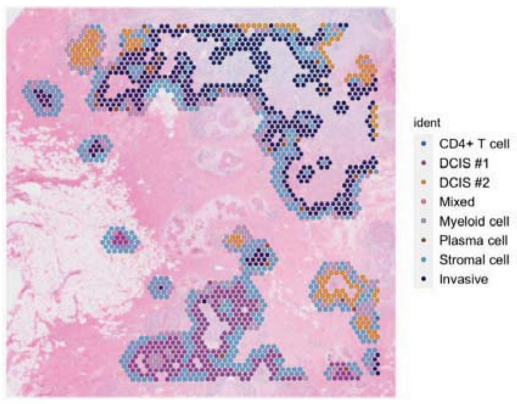
Load data

# Subset only tumor boundaries and layers
```
visium.breast@meta.data = merge(visium.breast@meta.data, loupe.data,
                                by = "row.names")

row.names(visium.breast@meta.data) = visium.breast@meta.data$Row.names

Idents(visium.breast) = visium.breast@meta.data$layer_annotation

tumor.bdy = subset(visium.breast, nCount_Spatial > 100,
                   idents = c("Bdy", "Layer"))
```

Visualization of our data

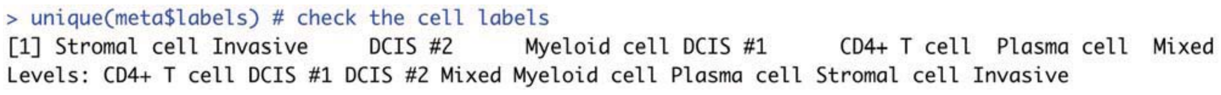
```
Idents(tumor.bdy) = tumor.bdy$first_type

names(colors) = c("B cell", "CD4+ T cell",  "DCIS #1", "DCIS #2",
                  "Myeloid cell", "Plasma cell", "Stromal cell", "Invasive")

SpatialDimPlot(tumor.bdy, label = F, cols = colors)
```

There are 8 cell types from our final tumor boundary data.



Prepare input data for CellChat analysis

```
data.input = GetAssayData(tumor.bdy, slot = "data", assay = "SCT")
meta = data.frame(labels = Idents(tumor.bdy),
              row.names = names(Idents(tumor.bdy)))

# check the cell labels
unique(meta$labels)
```



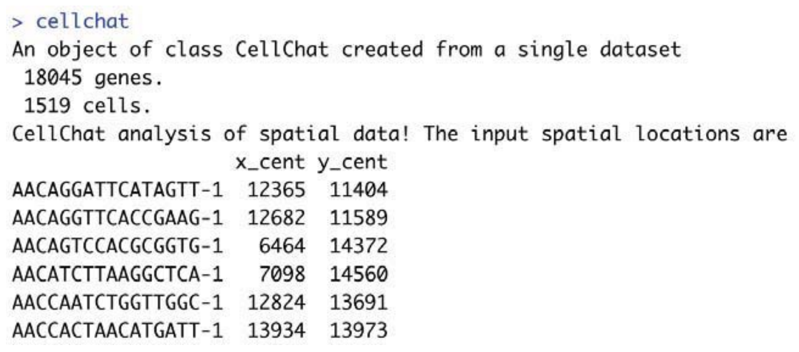
```
> unique(meta$labels) # check the cell labels
[1] Stromal cell Invasive     DCIS #2      Myeloid cell DCIS #1      CD4+ T cell  Plasma cell  Mixed
Levels: CD4+ T cell DCIS #1 DCIS #2 Mixed Myeloid cell Plasma cell Stromal cell Invasive
```

## Load spatial imaging information

# Load spatial imaging information to get the spot information
```
spatial.locs = GetTissueCoordinates(tumor.bdy, scale = NULL,
                                    cols = c("imagerow", "imagecol"))

scale.factors = jsonlite::fromJSON(txt =
    "./Raw_file/visium/spatial/scalefactors_json.json")

scale.factors = list(spot.diameter = 65,
                     spot = scale.factors$spot_diameter_fullres,
                     fiducial = scale.factors$fiducial_diameter_fullres,
                     hires = scale.factors$tissue_hires_scalef,
                     lowres = scale.factors$tissue_lowres_scalef)
```

## Create a CellChat object

# Create a CellChat object for the downstream analysis
```
cellchat = createCellChat(object = data.input,
                          meta = meta,
                          group.by = "labels",
                          datatype = "spatial",
                          coordinates = spatial.locs,
                          scale.factors = scale.factors)
cellchat
```


```
> cellchat
An object of class CellChat created from a single dataset
 18045 genes.
 1519 cells.
CellChat analysis of spatial data! The input spatial locations are
                   x_cent y_cent
AACAGGATTCATAGTT-1  12365  11404
AACAGGTTCACCGAAG-1  12682  11589
AACAGTCCACGCGGTG-1   6464  14372
AACATCTTAAGGCTCA-1   7098  14560
AACCAATCTGGTTGGC-1  12824  13691
AACCACTAACATGATT-1  13934  13973
```

Set the ligand-receptor interaction database

# Load CellChat DB
```
CellChatDB = CellChatDB.human
cellchat.gene = as.data.frame(CellChatDB.human$geneInfo$Symbol)
colnames(cellchat.gene) = "gene”
```

# Load Xenium gene panel
```
xenium.gene =
    read.csv("./Raw_file/xenium/Xenium_FFPE_Human_Breast_Cancer_Rep1_gene_group
s.csv")
```

CellChatDB : Manually curated database of literature-supported ligand-receptor interactions in both **human and mouse**.

Since our toy data is a human breast 10x visium data, we load **CellChatDB.human.**

Xenium In Situ Datasets : **313** genes chosen to explore a Xenium In Situ dataset from human breast cancer FFPE section.

Set the ligand-receptor interaction database

# Filter CellChat DB by Xenium gene
```
overlap.gene = merge(cellchat.gene, xenium.gene, by = "gene")

CellChatDB$interaction =
    CellChatDB$interaction[CellChatDB$interaction$ligand %in% overlap.gene$gene
        & CellChatDB$interaction$receptor %in% overlap.gene$gene,]

cellchat@DB = CellChatDB
```

Preprocessing of the expression data for cell-cell communication analysis

# Subset the expression data of signaling genes for saving computation cost
```
cellchat = subsetData(cellchat)
```

# Identify over-expressed ligands or receptors in one cell group
```
cellchat = identifyOverExpressedGenes(cellchat)
```

# Identify over-expressed ligand-receptor interactions if either ligand or receptor is over-expressed
```
cellchat = identifyOverExpressedInteractions(cellchat)
```

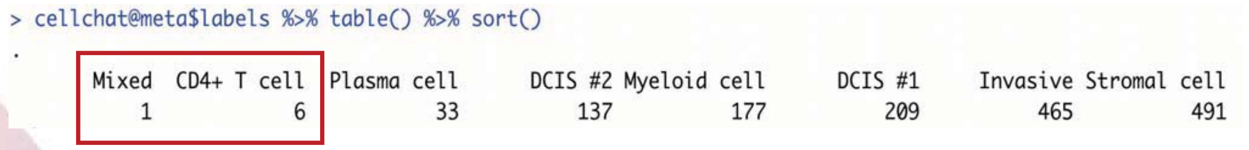
Compute the communication probability and infer cellular communication network

# Infers the biologically significant cell-cell communication with permutation test
```
cellchat = computeCommunProb(cellchat, type = "triMean", distance.use = TRUE,
                             interaction.length = 200, scale.distance = 0.1)
```

Compute the communication probability and infer cellular communication network

# Check the number of spots of a cell type
```
cellchat@meta$labels %>% table() %>% sort()
```

# Filter cell-cell communication if there are only few number of spots in certain cell types
```
cellchat = filterCommunication(cellchat, min.cells = 10)
                           interaction.length = 200, scale.distance = 0.1)
```


```
> cellchat@meta$labels %>% table() %>% sort()
.
     Mixed   CD4+ T cell  Plasma cell      DCIS #2 Myeloid cell        DCIS #1    Invasive Stromal cell
         1             6           33          137          177            209          465          491
```

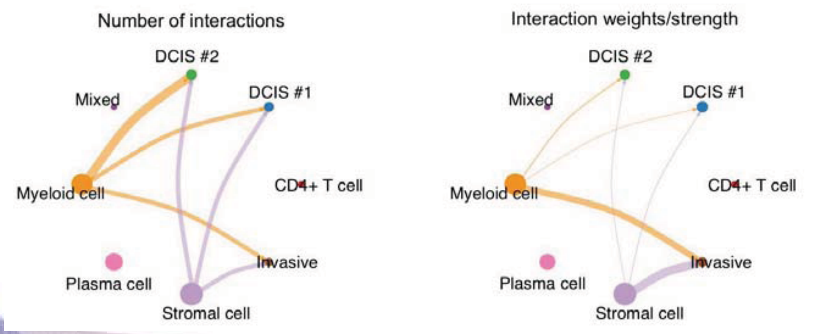
Compute the communication probability and infer cellular communication network

# Calculate the aggregated cell-cell communication network
```
cellchat = aggregateNet(cellchat)

#saveRDS(cellchat, file = './object/Bioinfo2023_Breast_CellChat.rds')
```

# read RDS file if computeCommunProb() takes too much time (optional)
```
cellchat = readRDS('./object/Bioinfo2023_Breast_CellChat.rds')
```

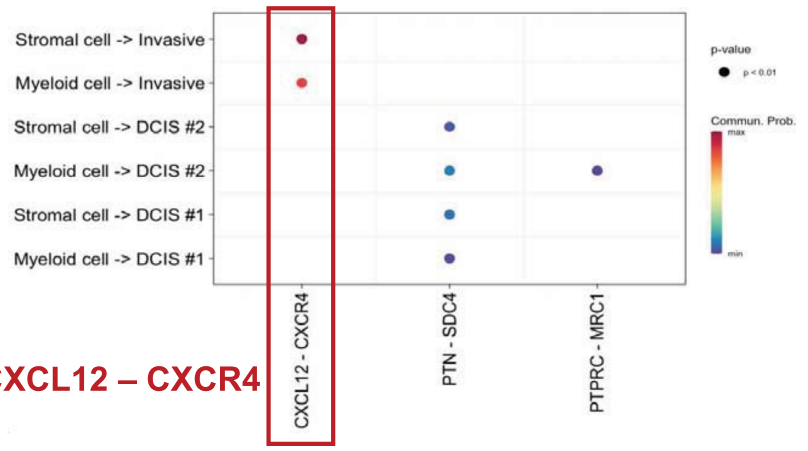
Visualization of the aggregated cell-cell communication network

```
netVisual_circle(cellchat@net$count, vertex.weight =
rowSums(cellchat@net$count),
                          sources.use =  c("Stromal cell", "Myeloid cell"),
                          targets.use = c("DCIS #1", "DCIS #2", "Invasive"),
                          title.name = "Number of interactions")

netVisual_circle(cellchat@net$weight, vertex.weight =
rowSums(cellchat@net$weight),
                          sources.use =  c("Stromal cell", "Myeloid cell"),
                          targets.use = c("DCIS #1", "DCIS #2", "Invasive"),
                          title.name = "Interaction weights/strength")
```



Identify ligand-receptor pairs between cell types

```
CellChat::netVisual_bubble(cellchat,
                 sources.use = c("Stromal cell", "Myeloid cell"),
                 targets.use = c("DCIS #1", "DCIS #2", "Invasive"),
                 remove.isolate = FALSE, angle.x = 90, thresh = 0.05) +
                 coord_flip()
```

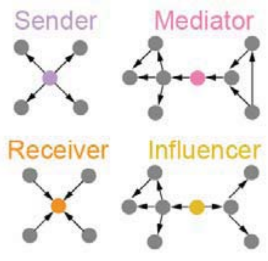


CXCL12 – CXCR4

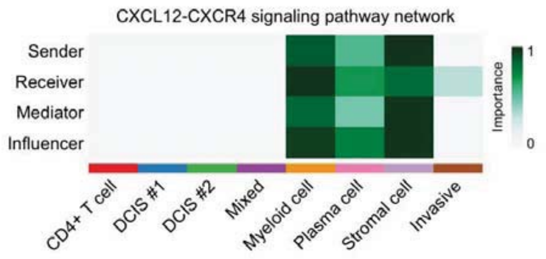
Compute the network centrality scores

```
# Compute the network centrality scores
cellchat = netAnalysis_computeCentrality(cellchat, net.name = "CXCL12-CXCR4")

# Visualize the centrality score
netAnalysis_signalingRole_network(cellchat, signaling = "CXCL12-CXCR4",
                                  width = 8, height = 2.5, font.size = 10)
```

Visualize the computed centrality scores using heatmap, allowing ready identification of major signaling roles of cell groups.

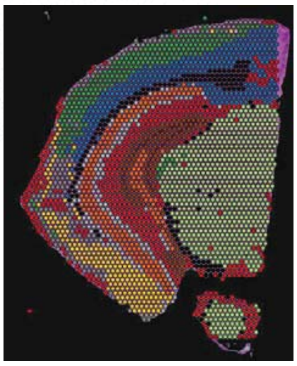


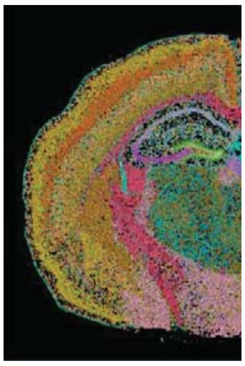
https://github.com/jinworks/CellChat



8. Xenium in situ

Differences between Visium and Xenium





https://support.10xgenomics.com/spatial-gene-expression/software/pipelines/latest/rkit
https://www.10xgenomics.com/platforms/xenium

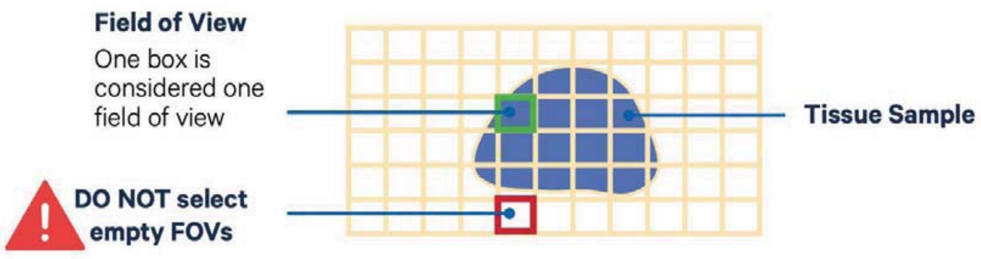
	10x Visium	10x Xenium
Number of genes	Tens of thousands	Hundreds to thousands
Spatial resolution	Lower resolution (~55μm)	Higher subcellular resolution (~200nm)
Cell-cell boundaries	Not defined	Defined with cellular segmentation

Load and preprocess the dataset

# Load the Xenium data
```
s1r1 =
LoadXenium('Raw_file/xenium/Xenium_FFPE_Human_Breast_Cancer_Rep1_outs',
fov = 'fov')
```

# Remove cells with 0 counts
```
S1r1 = subset(s1r1, subset = nCount_Xenium > 0)
```

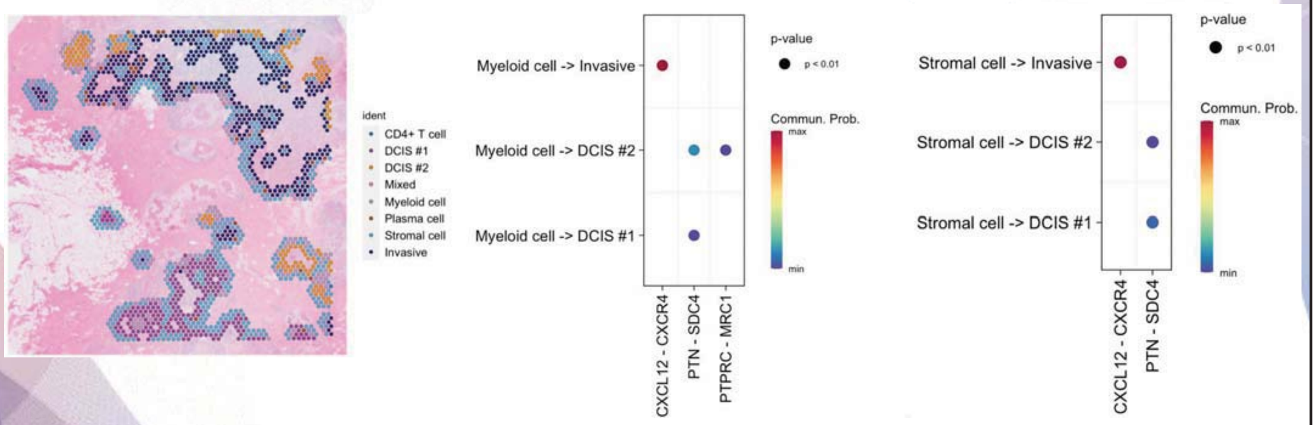
# Add metadata
```
s1r1@meta.data$cells = 'cells'
```



https://cdn.10xgenomics.com/image/upload/v1694469210/support-documents/CG000584_Xenium_Analyzer_UserGuide_RevC.pdf

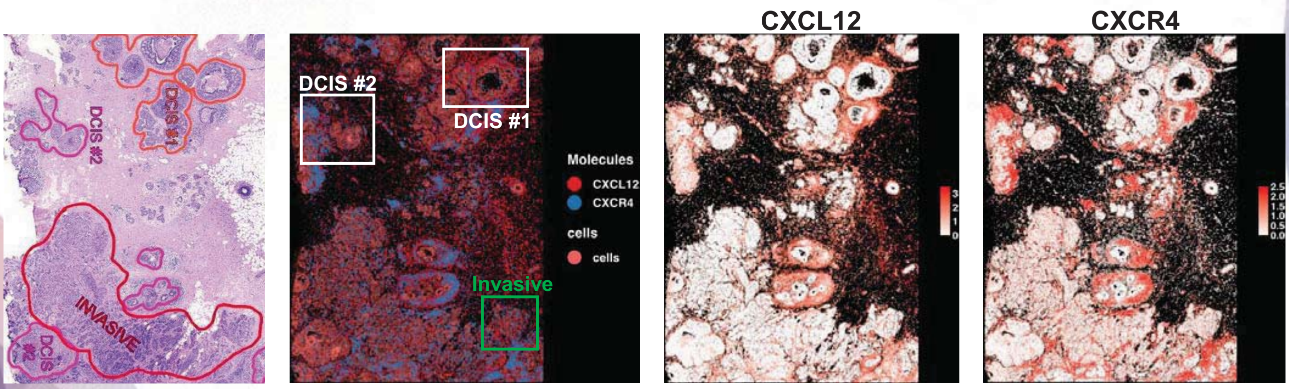
Intercellular communication inferred by cellchat

- Myeloid / Stromal cells - tumor cells (CXCL12 - CXCR4, PTN - SDC4)



Visualize the expression level of CXCL12 and CXCR4

```
# Plot the positions of CXCL12 and CXCR4
ImageDimPlot(s1r1, fov = "fov", molecules = c("CXCL12", "CXCR4"),
group.by = 'cells', nmols = 20000)

# Visualize the expression level of CXCL12 and CXCR4
ImageFeaturePlot(s1r1, features = c("CXCL12", "CXCR4"), max.cutoff =
c(15, 3), size = 0.5, cols = c("white", "red"))
```



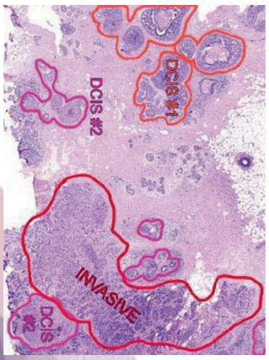
## Visualize the expression level of PTN and SDC4

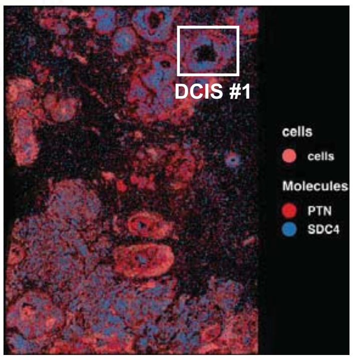
# Plot the positions of PTN and SDC4

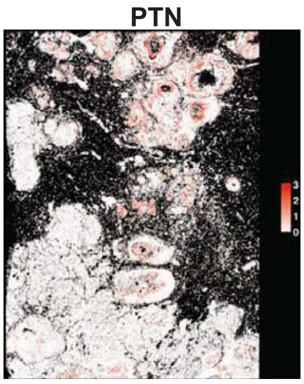
```
ImageDimPlot(s1r1, fov = "fov", molecules = c("PTN", "SDC4"), group.by =
'cells', nmols = 20000)
```

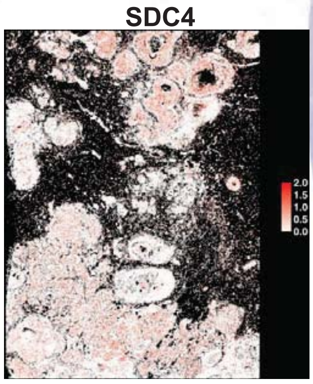
# Visualize the expression level of PTN and SDC4

```
ImageFeaturePlot(s1r1, features = c("PTN", "SDC4"), max.cutoff = c(8, 8),
size = 0.5, cols = c("white", "red"))
```









Zoom in on the PTN – SDC4 binding area

# Increase your RAM usage (8GB)

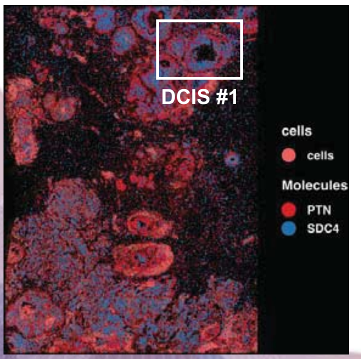
```
options(future.globals.maxSize = 8000 * 1024^2)
```

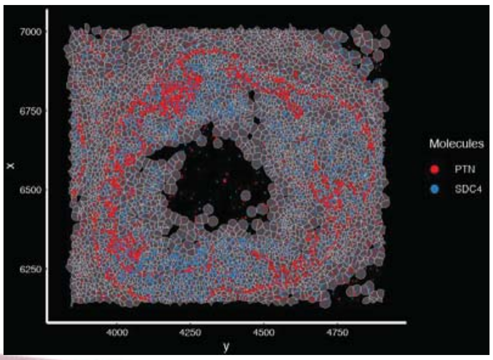
# Define cropped area

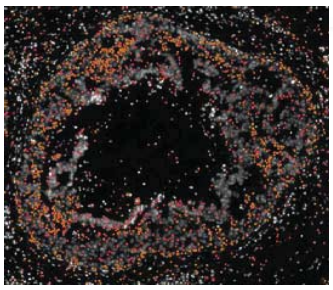
```
cropped.coords = Crop(s1r1[["fov"]], x = c(3850, 4900), y = c(6150,
7000), coords = "plot")
s1r1[["zoom"]] = cropped.cords
```

# Visualize cropped area with cell segmentations & selected molecules

```
DefaultBoundary(s1r1[["zoom"]]) = "segmentation"
ImageDimPlot(s1r1, fov = "zoom", axes = TRUE, border.color = "white",
border.size = 0.1, cols = "polychrome", coord.fixed = FALSE, molecules =
c("PTN", "SDC4"), nmols = 10000, group.by = 'cells')
```



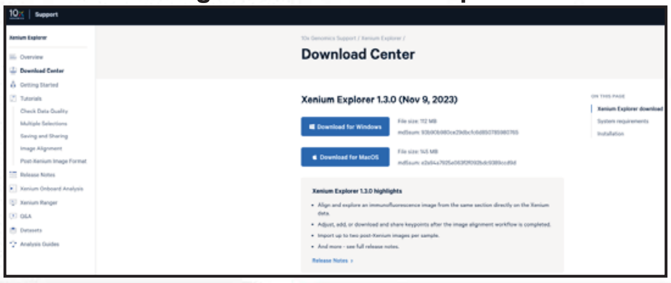




PTN SDC4

Visualize RNA transcript localization in tissue using Xenium Explorer

10x genomics Xenium Explorer



10x | Support

Xenium Explorer
- Overview
- Download Center
- Getting Started
- Tutorials
  - Check Data Quality
  - Multiple Selections
  - Saving and Sharing
  - Image Alignment
  - Post-Xenium Image Format
- Release Notes
- Xenium Onboard Analysis
- Xenium Ranger
- Q&A
- Datasets
- Analysis Guides

10x Genomics Support / Xenium Explorer /

Download Center

Xenium Explorer 1.3.0 (Nov 9, 2023)

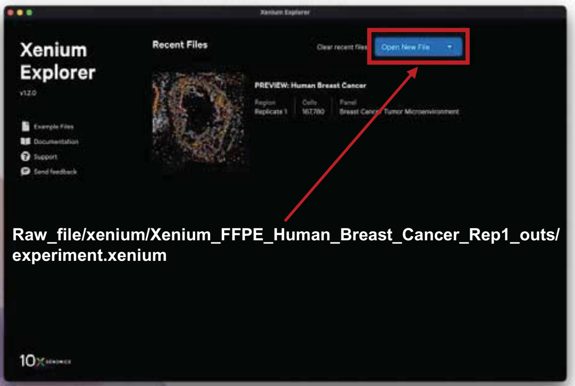
Download for Windows — File size: 112 MB, md5sum: 93b90b980ce29dbcfc6d8507185980765

Download for MacOS — File size: 145 MB, md5sum: e2a84a7925e063f2f0928dc9389ccd9d

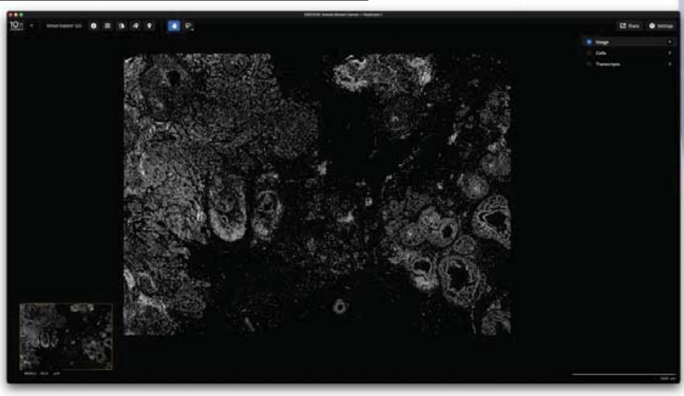
Xenium Explorer 1.3.0 highlights
- Align and explore an immunofluorescence image from the same section directly on the Xenium data.
- Adjust, add, or download and share keypoints after the image alignment workflow is completed.
- Import up to two post-Xenium images per sample.
- And more - see full release notes.

Release Notes ›

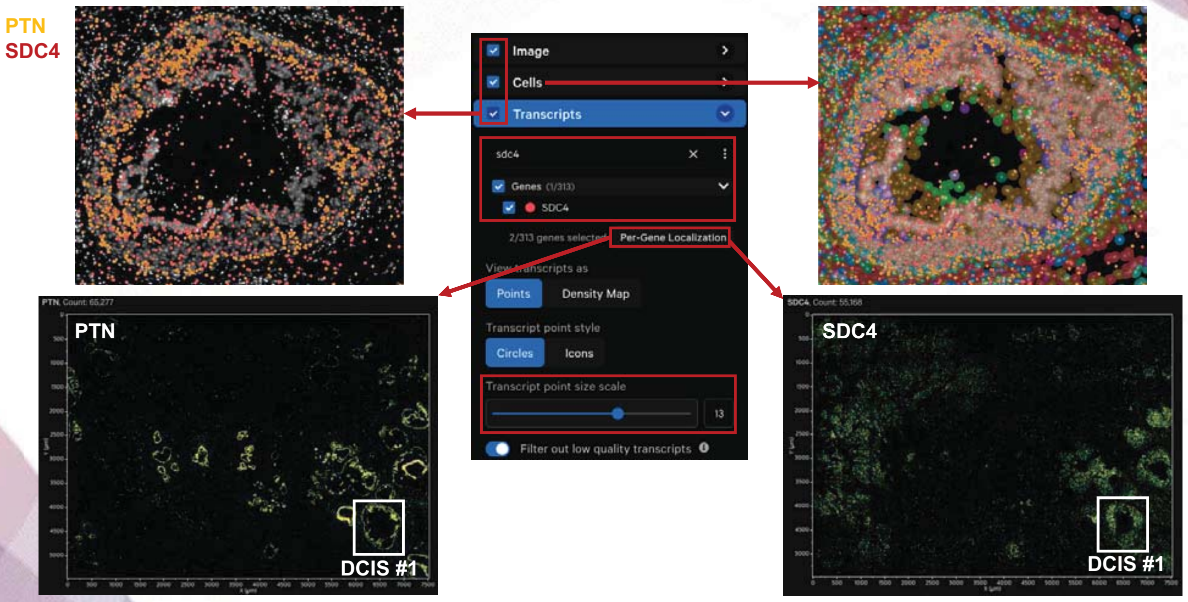
On this page: Xenium Explorer download · System requirements · Installation



Raw_file/xenium/Xenium_FFPE_Human_Breast_Cancer_Rep1_outs/experiment.xenium



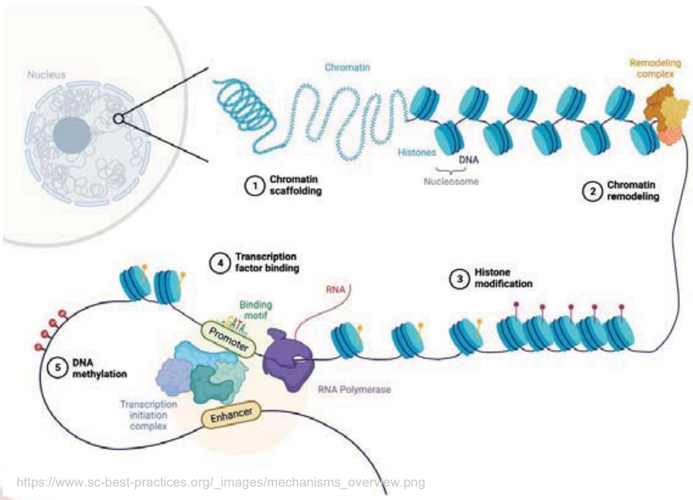
Visualize RNA transcript localization in tissue using Xenium Explorer



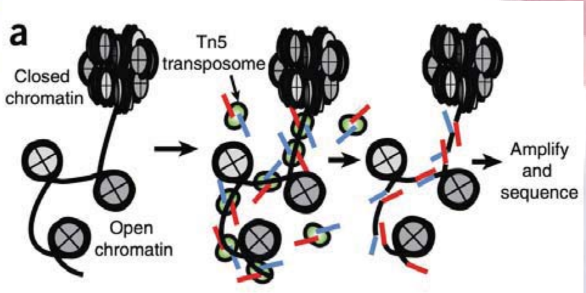
- ☑ Image
- ☑ Cells
- ☑ Transcripts
  - sdc4
  - ☑ Genes (1/313)
    - ☑ ● SDC4
  - 2/313 genes selected · Per-Gene Localization
  - View transcripts as: [✓ Points] Density Map
  - Transcript point style: [✓ Circles] Icons
  - Transcript point size scale: 13
  - Filter out low quality transcripts

9. ArchR

What is scATAC-seq and why is it used?



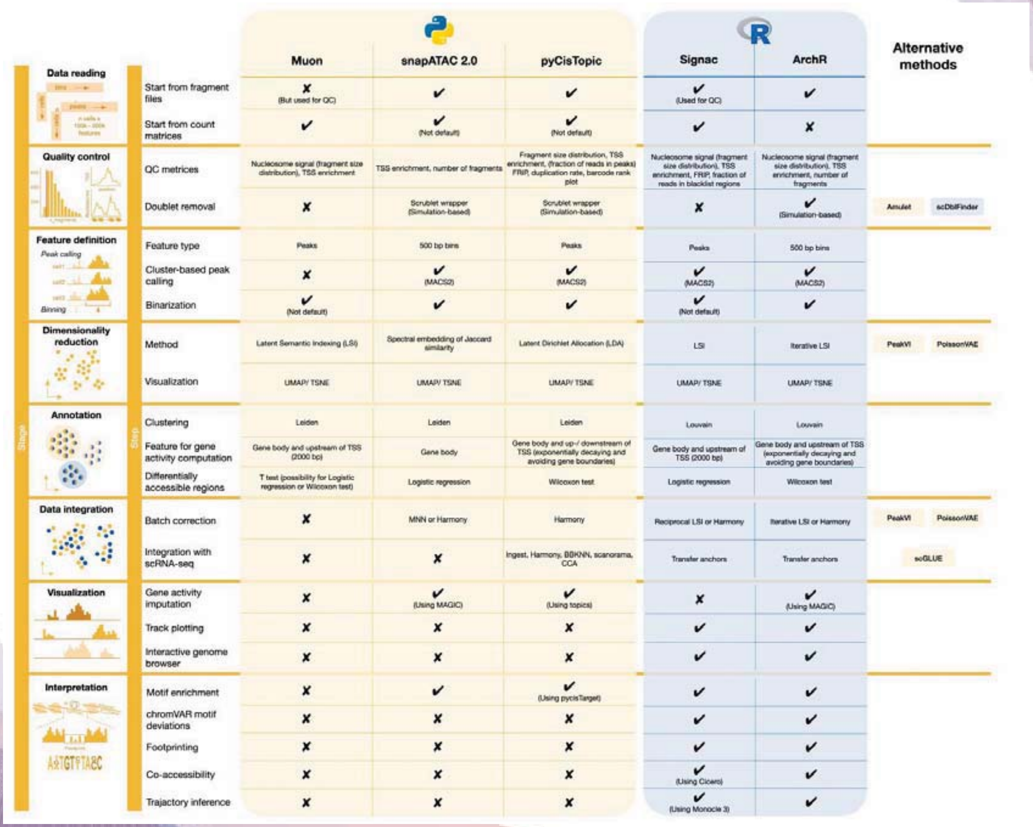
https://www.sc-best-practices.org/_images/mechanisms_overview.png



Buenrostro, Jason D., et al. *Nature methods* (2013)

- ✓ Gene expression control is orchestrated by a complex interplay of regulatory mechanisms, including DNA methylation, histone modifications, and transcription factor activity.
- ✓ Chromatin accessibility predominantly mirrors a cell's overall regulatory state, providing supplementary information to mRNA levels, which delineate cell identity.
- ✓ Moreover, delving into chromatin accessibility profiles offers further understanding of gene regulatory mechanisms and cellular differentiation processes that may not be fully elucidated by single-cell RNA sequencing data.

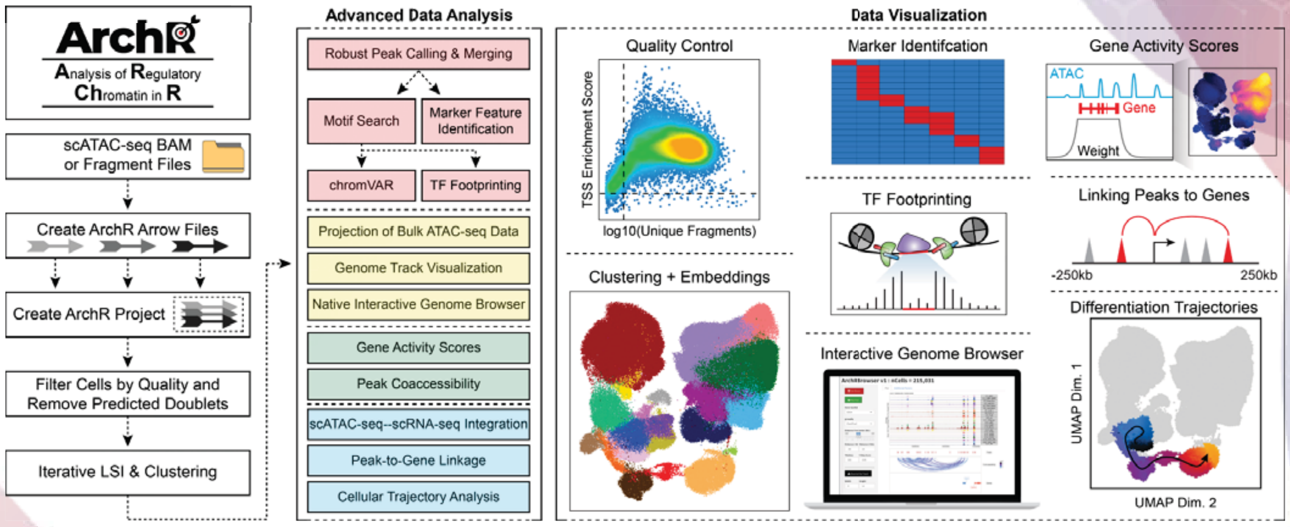
Several tools for the scATAC-seq data analysis



Stage	Step	Muon	snapATAC 2.0	pyCisTopic	Signac	ArchR	Alternative methods
Data reading	Start from fragment files	✗ (But used for QC)	✓	✓	✓ (Used for QC)	✓	
	Start from count matrices	✓	✓ (Not default)	✓ (Not default)	✓	✗	
Quality control	QC metrices	Nucleosome signal (fragment size distribution), TSS enrichment	TSS enrichment, number of fragments	Fragment size distribution, TSS enrichment, (fraction of reads in peaks) FRiP, duplication rate, barcode rank plot	Nucleosome signal (fragment size distribution), TSS enrichment, FRiP, fraction of reads in blacklist regions	Nucleosome signal (fragment size distribution), TSS enrichment, number of fragments	
	Doublet removal	✗	Scrublet wrapper (Simulation-based)	Scrublet wrapper (Simulation-based)	✗	✓ (Simulation-based)	Amulet, scDblFinder
Feature definition	Feature type	Peaks	500 bp bins	Peaks	Peaks	500 bp bins	
	Cluster-based peak calling	✗	✓ (MACS2)	✓ (MACS2)	✓ (MACS2)	✓ (MACS2)	
	Binarization	✓ (Not default)	✓	✓	✓ (Not default)	✓	
Dimensionality reduction	Method	Latent Semantic Indexing (LSI)	Spectral embedding of Jaccard similarity	Latent Dirichlet Allocation (LDA)	LSI	Iterative LSI	PeakVI, PoissonVAE
	Visualization	UMAP/ TSNE	UMAP/ TSNE	UMAP/ TSNE	UMAP/ TSNE	UMAP/ TSNE	
Annotation	Clustering	Leiden	Leiden	Leiden	Louvain	Louvain	
	Feature for gene activity computation	Gene body and upstream of TSS (2000 bp)	Gene body	Gene body and up-/ downstream of TSS (exponentially decaying and avoiding gene boundaries)	Gene body and upstream of TSS (2000 bp)	Gene body and upstream of TSS (exponentially decaying and avoiding gene boundaries)	
	Differentially accessible regions	T test (possibility for Logistic regression or Wilcoxon test)	Logistic regression	Wilcoxon test	Logistic regression	Wilcoxon test	
Data integration	Batch correction	✗	MNN or Harmony	Harmony	Reciprocal LSI or Harmony	Iterative LSI or Harmony	PeakVI, PoissonVAE
	Integration with scRNA-seq	✗	✗	Ingest, Harmony, BBKNN, scanorama, CCA	Transfer anchors	Transfer anchors	scGLUE
Visualization	Gene activity imputation	✗	✓ (Using MAGIC)	✓ (Using topics)	✗	✓ (Using MAGIC)	
	Track plotting	✗	✗	✗	✓	✓	
	Interactive genome browser	✗	✗	✗	✓	✓	
Interpretation	Motif enrichment	✗	✓	✓ (Using pycisTarget)	✓	✓	
	chromVAR motif deviations	✗	✗	✗	✓	✓	
	Footprinting	✗	✗	✗	✓	✓	
	Co-accessibility	✗	✗	✗	✓ (Using Cicero)	✓	
	Trajectory inference	✗	✗	✗	✓ (Using Monocle 3)	✓	

109

What is ArchR?



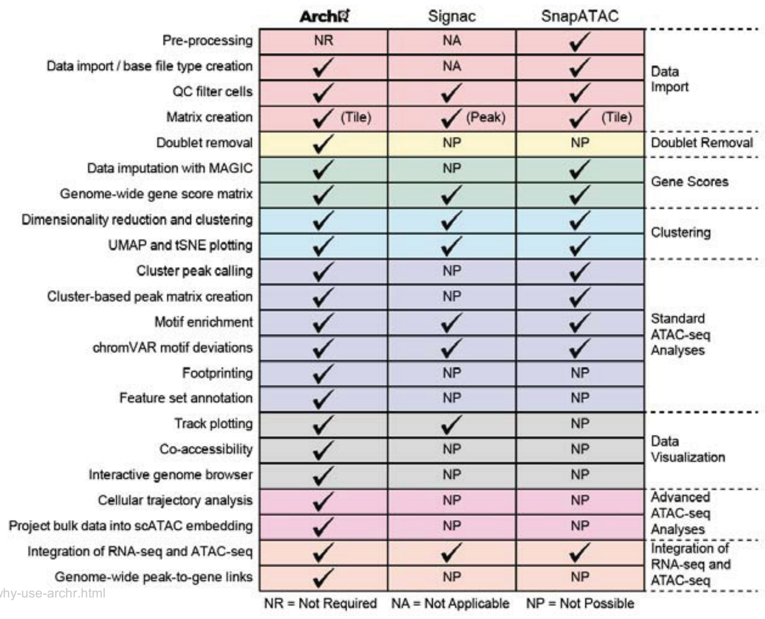
ArchR
Analysis of Regulatory Chromatin in R
scATAC-seq BAM or Fragment Files
Create ArchR Arrow Files
Create ArchR Project
Filter Cells by Quality and Remove Predicted Doublets
Iterative LSI & Clustering
Advanced Data Analysis
Robust Peak Calling & Merging
Motif Search
Marker Feature Identification
chromVAR
TF Footprinting
Projection of Bulk ATAC-seq Data
Genome Track Visualization
Native Interactive Genome Browser
Gene Activity Scores
Peak Coaccessibility
scATAC-seq--scRNA-seq Integration
Peak-to-Gene Linkage
Cellular Trajectory Analysis
Data Visualization
Quality Control
TSS Enrichment Score
log10(Unique Fragments)
Marker Identifcation
Gene Activity Scores
ATAC
Gene
Weight
TF Footprinting
Linking Peaks to Genes
-250kb
250kb
Clustering + Embeddings
Interactive Genome Browser
Differentiation Trajectories
UMAP Dim. 1
UMAP Dim. 2

https://github.com/GreenleafLab/ArchR?tab=readme-ov-file

- ✓ **ArchR** is a **comprehensive software suite for end-to-end analysis** of single-cell chromatin accessibility that will accelerate the understanding of gene regulation at the resolution of individual cells.
- ✓ Enabling the analysis of over **1.2 million single cells within 8 h on a standard Unix laptop**

110

Why ArchR?



	ArchR	Signac	SnapATAC	
Pre-processing	NR	NA	✓	Data Import
Data import / base file type creation	✓	NA	✓	
QC filter cells	✓	✓	✓	
Matrix creation	✓ (Tile)	✓ (Peak)	✓ (Tile)	
Doublet removal	✓	NP	NP	Doublet Removal
Data imputation with MAGIC	✓	NP	✓	Gene Scores
Genome-wide gene score matrix	✓	✓	✓	
Dimensionality reduction and clustering	✓	✓	✓	Clustering
UMAP and tSNE plotting	✓	✓	✓	
Cluster peak calling	✓	NP	✓	Standard ATAC-seq Analyses
Cluster-based peak matrix creation	✓	NP	✓	
Motif enrichment	✓	✓	✓	
chromVAR motif deviations	✓	✓	✓	
Footprinting	✓	NP	NP	
Feature set annotation	✓	NP	NP	
Track plotting	✓	✓	NP	Data Visualization
Co-accessibility	✓	NP	NP	
Interactive genome browser	✓	NP	NP	
Cellular trajectory analysis	✓	NP	NP	Advanced ATAC-seq Analyses
Project bulk data into scATAC embedding	✓	NP	NP	
Integration of RNA-seq and ATAC-seq	✓	✓	✓	Integration of RNA-seq and ATAC-seq
Genome-wide peak-to-gene links	✓	NP	NP	

NR = Not Required NA = Not Applicable NP = Not Possible

https://www.archrproject.com/bookdown/why-use-archr.html

- ✓ **ArchR** provides features and enables analyses that other tools do not:
- ✓ **ArchR** is **faster** and **uses less memory** than other available tools due to heavy optimization of the data structures and parallelization methods that form the basis of the ArchR software.

Installation of ArchR

- ✓ **Visit https://github.com/choilab-hr/KSBI_BIML_2024/tree/main/03_scATAC_seq/ArchR**
- ✓ **ArchR** is designed to be run on Unix-based operating systems such as macOS and linux. ArchR is **NOT supported on Windows or other operating systems.**
- ✓ **ArchR** installation currently requires devtools and BiocManager for installation of GitHub and Bioconductor packages.

First, install devtools (for installing GitHub packages) if it isn't already installed:

```
if (!requireNamespace("devtools", quietly = TRUE)) install.packages("devtools")
```

Then, install BiocManager (for installing bioconductor packages) if it isn't already installed:

```
if (!requireNamespace("BiocManager", quietly = TRUE)) install.packages("BiocManager")
```

Then, install ArchR:

```
devtools::install_github("GreenleafLab/ArchR", ref="master", repos =
BiocManager::repositories())
```

Install all of the ArchR dependencies that arent installed by default:

```
library(ArchR)
ArchR::installExtraPackages()
```

Set a working directory variable for the session:

```
biml_dir <- 'your/directory'
setwd(biml_dir)
```

Load ArchR and Download tutorial data

First, load **ArchR** and set a random seed.

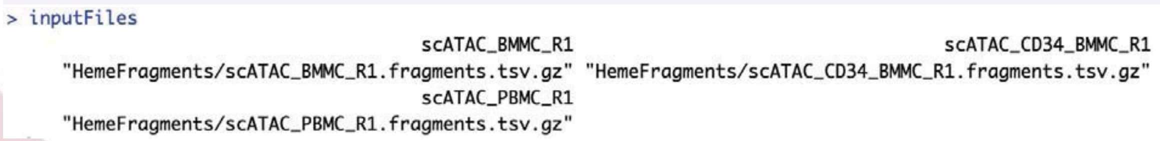
```
library(ArchR)
set.seed(2024)
```

Set the default number of threads for parallelized operations in **ArchR** functions.

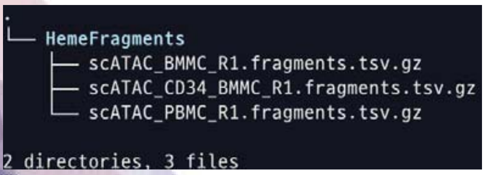
```
((ncore <- parallel::detectCores())) # 10
addArchRThreads(threads = ncore-2)
```

Get tutorial data for the session.

```
inputFiles <- getTutorialData("Hematopoiesis")
inputFiles
```



```
> inputFiles
                                          scATAC_BMMC_R1                                       scATAC_CD34_BMMC_R1
     "HemeFragments/scATAC_BMMC_R1.fragments.tsv.gz" "HemeFragments/scATAC_CD34_BMMC_R1.fragments.tsv.gz"
                                          scATAC_PBMC_R1
     "HemeFragments/scATAC_PBMC_R1.fragments.tsv.gz"
```



```
.
└── HemeFragments
    ├── scATAC_BMMC_R1.fragments.tsv.gz
    ├── scATAC_CD34_BMMC_R1.fragments.tsv.gz
    └── scATAC_PBMC_R1.fragments.tsv.gz

2 directories, 3 files
```

Lastly, add a reference genome annotation for **ArchR.**

```
addArchRGenome("hg19")
```

Creating Arrow files

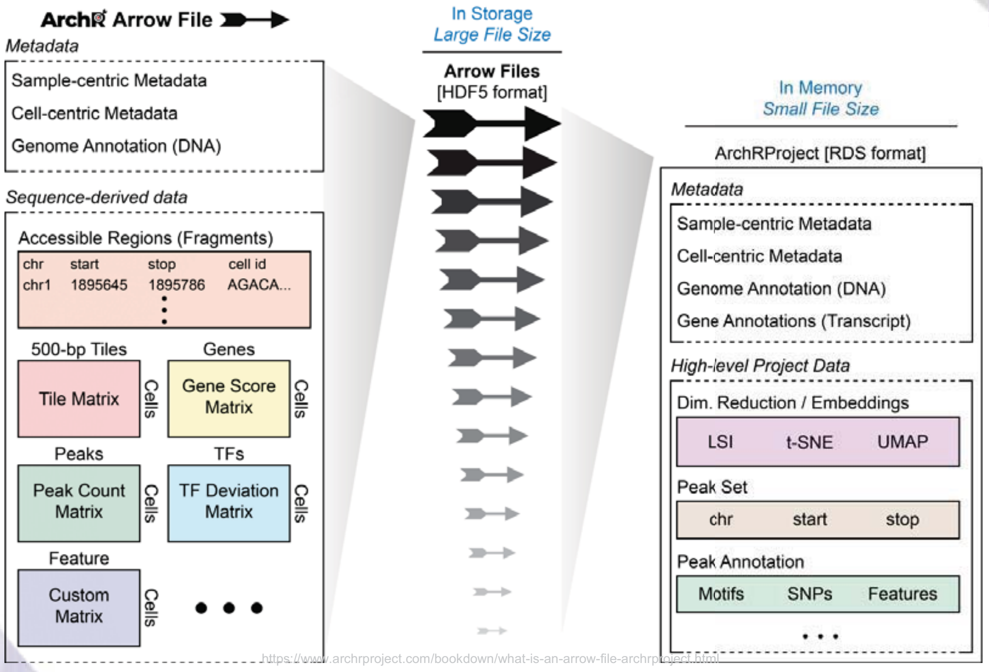
Now we will create our Arrow files which will take **10-15 minutes**. For each sample, this step will:

1. Read accessible fragments from the provided input files.
2. Calculate **quality control information for each cell** (i.e. TSS enrichment scores and nucleosome info).
3. **Filter cells** based on quality control parameters.
4. Create a genome-wide TileMatrix using 500-bp bins.
5. Create a GeneScoreMatrix using the custom geneAnnotation that was defined when we called addArchRGenome().

```
ArrowFiles <- createArrowFiles(
  inputFiles = inputFiles,
  sampleNames = names(inputFiles),
  filterTSS = 4, #Dont set this too high because you can always increase later
  filterFrags = 1000,
  addTileMat = TRUE,
  addGeneScoreMat = TRUE
)

ArrowFiles
```

What is an Arrow file?

✓ The base unit of an analytical project in ArchR is called an Arrow file.



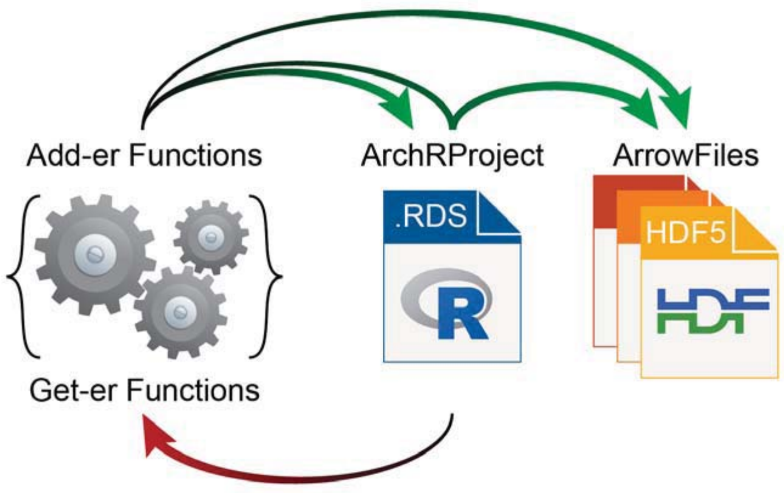
https://www.archrproject.com/bookdown/what-is-an-arrow-file-archrproject.html

✓ Each Arrow file stores all of the data associated with an individual sample (i.e. metadata, accessible fragments, and data matrices).

What is an ArchRProject?

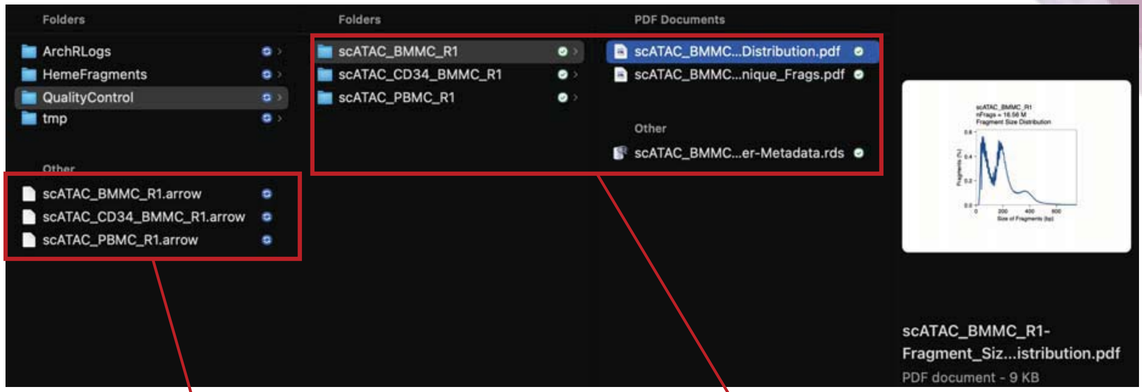
✓ More explicitly, an Arrow file is not an R-language object that is stored in memory but rather an HDF5-format file stored on disk.

✓ Because of this, we use an **ArchRProject** object to associate these Arrow files together into a single analytical framework that can be rapidly accessed in R. This ArchRProject object is small in size and is stored in memory.



https://www.archrproject.com/bookdown/what-is-an-arrow-file-archrproject.html

After the creation of Arrow files


Folders
ArchRLogs
HemeFragments
QualityControl
tmp
Other
scATAC_BMMC_R1.arrow
scATAC_CD34_BMMC_R1.arrow
scATAC_PBMC_R1.arrow

Folders
scATAC_BMMC_R1
scATAC_CD34_BMMC_R1
scATAC_PBMC_R1

PDF Documents
scATAC_BMMC...Distribution.pdf
scATAC_BMMC...nique_Frags.pdf
Other
scATAC_BMMC...er-Metadata.rds

scATAC_BMMC_R1-Fragment_Siz...istribution.pdf
PDF document - 9 KB

Per samples' Arrow file

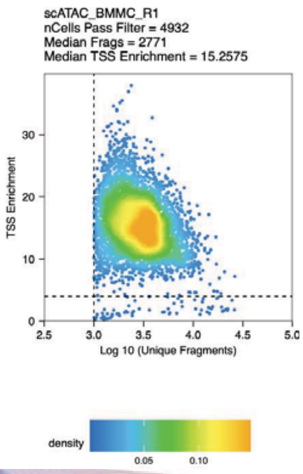
Per samples' Quality Control results

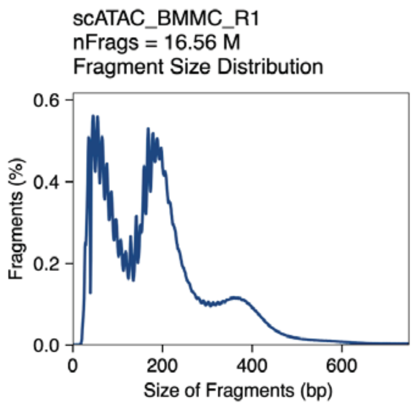
We are now ready to tidy up these Arrow files and then create an ArchRProject.

Quality Control of scATAC-seq data

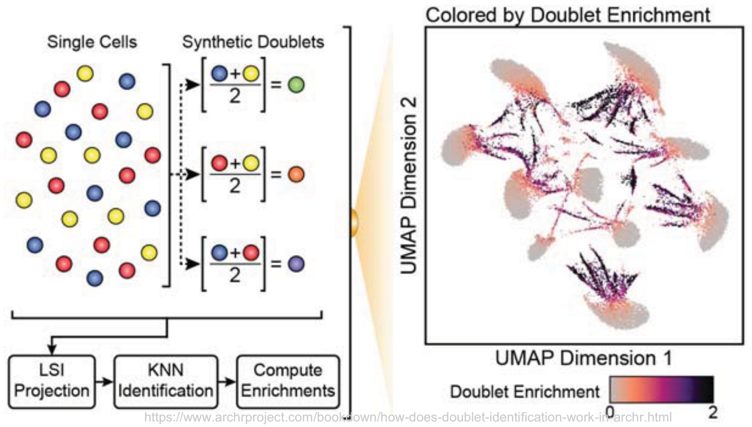
✓ In ArchR, three characteristics are considered for the quality control.

1. The number of unique nuclear fragments (i.e. not mapping to mitochondrial DNA).
2. The signal-to-background ratio. Low signal-to-background ratio is often attributed to dead or dying cells which have de-chromatinzed DNA which allows for random transposition genome-wide.
3. The fragment size distribution. Due to nucleosomal periodicity, we expect to see depletion of fragments that are the length of DNA wrapped around a nucleosome (approximately **147 bp**).





Doublet Inference with ArchR



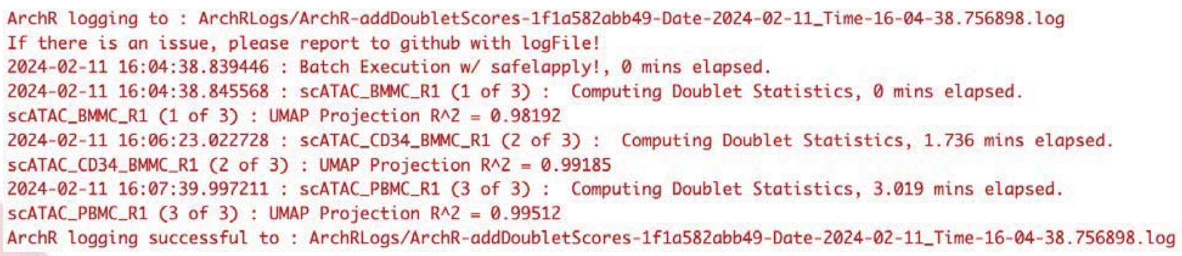
https://www.archrproject.com/bookdown/how-does-doublet-identification-work-in-archr.html

1. Synthesize in silico doublets from the data by mixing the reads from thousands of combinations of individual cells.
2. Project these synthetic doublets into the UMAP embedding and identify their nearest neighbor.
3. Identify "cells" in our data whose signal looks very similar to synthetic doublets.

Inferring scATAC-seq Doublets with ArchR

One simple function infers scATAC-seq Doublets.

```
doubScores <- addDoubletScores(
  input = ArrowFiles,
  k = 10, #Refers to how many cells near a "pseudo-doublet" to count.
  knnMethod = "UMAP", #Refers to the embedding to use for nearest neighbor search.
  LSIMethod = 1
)
```

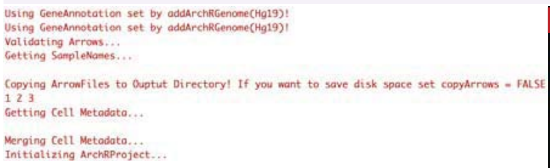


```
ArchR logging to : ArchRLogs/ArchR-addDoubletScores-1f1a582abb49-Date-2024-02-11_Time-16-04-38.756898.log
If there is an issue, please report to github with logFile!
2024-02-11 16:04:38.839446 : Batch Execution w/ safelapply!, 0 mins elapsed.
2024-02-11 16:04:38.845568 : scATAC_BMMC_R1 (1 of 3) :  Computing Doublet Statistics, 0 mins elapsed.
scATAC_BMMC_R1 (1 of 3) : UMAP Projection R^2 = 0.98192
2024-02-11 16:06:23.022728 : scATAC_CD34_BMMC_R1 (2 of 3) :  Computing Doublet Statistics, 1.736 mins elapsed.
scATAC_CD34_BMMC_R1 (2 of 3) : UMAP Projection R^2 = 0.99185
2024-02-11 16:07:39.997211 : scATAC_PBMC_R1 (3 of 3) :  Computing Doublet Statistics, 3.019 mins elapsed.
scATAC_PBMC_R1 (3 of 3) : UMAP Projection R^2 = 0.99512
ArchR logging successful to : ArchRLogs/ArchR-addDoubletScores-1f1a582abb49-Date-2024-02-11_Time-16-04-38.756898.log
```

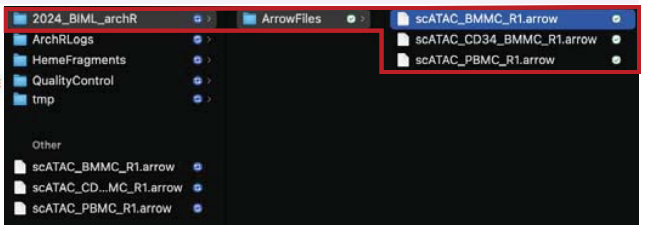
Creating an ArchRProject

✓ An ArchRProject allows us to group multiple Arrow files together into a single project.

```
proj <- ArchRProject(
  ArrowFiles = ArrowFiles,
  outputDirectory = "2024_BIML_archR",
  copyArrows = TRUE #This is recommened so that if you modify the Arrow files you
have an original copy for later usage.
)
```



```
Using GeneAnnotation set by addArchRGenome(Hg19)!
Using GeneAnnotation set by addArchRGenome(Hg19)!
Validating Arrows...
Getting SampleNames...

Copying ArrowFiles to Ouptut Directory! If you want to save disk space set copyArrows = FALSE
1 2 3
Getting Cell Metadata...

Merging Cell Metadata...
Initializing ArchRProject...
```

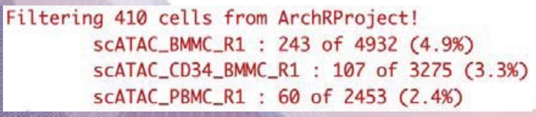


Which data matrices are available within the ArchRProject?

```
getAvailableMatrices(proj)
```

Filter putative doublets. This doesn't physically remove data from the Arrow files but rather tells the ArchRProject to ignore these cells for downstream analysis.

```
proj <- filterDoublets(ArchRProj = proj)
paste0("Memory Size = ", round(object.size(proj) / 10^6, 3), " MB")
```



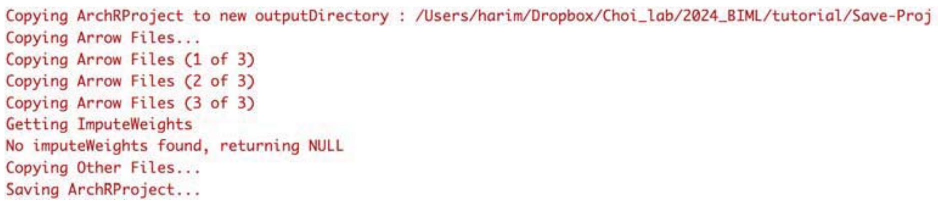
```
Filtering 410 cells from ArchRProject!
        scATAC_BMMC_R1 : 243 of 4932 (4.9%)
        scATAC_CD34_BMMC_R1 : 107 of 3275 (3.3%)
        scATAC_PBMC_R1 : 60 of 2453 (2.4%)
```

```
[1] "Memory Size = 37.389 MB"
```

Saving and Loading an ArchRProject

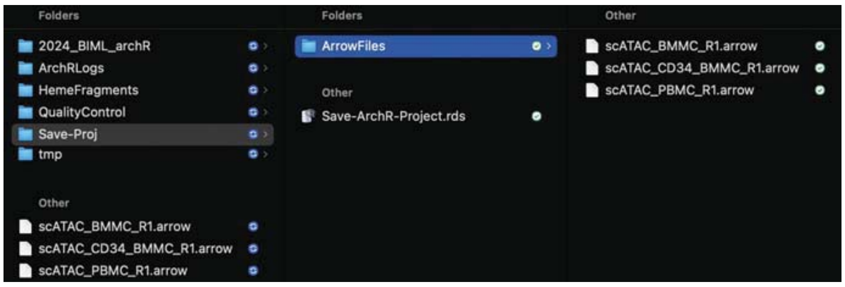
✓ It is crucial to save the ArchRProject after executing functions that require high computing power.

```
saveArchRProject(ArchRProj = proj, outputDirectory = "Save-Proj", load = FALSE)
```



```
Copying ArchRProject to new outputDirectory : /Users/harim/Dropbox/Choi_lab/2024_BIML/tutorial/Save-Proj
Copying Arrow Files...
Copying Arrow Files (1 of 3)
Copying Arrow Files (2 of 3)
Copying Arrow Files (3 of 3)
Getting ImputeWeights
No imputeWeights found, returning NULL
Copying Other Files...
Saving ArchRProject...
```

This will copy the **Arrow files** and save a **.RDS file** of the proj ArchRProject object in the specified outputDirectory.



To load an ArchRProject:

```
LoadedProj <- loadArchRProject(paste0(biml_dir, '/Save-Proj'))
rm(LoadedProj)
```

Load ArchRProject before the downstream analysis – "Optional"

If you have not followed the previous steps, don't worry. We provide ArchRProjcet file for the toturial.

First, check the `biml_dir`.

```
biml_dir
```

If there's no biml_dir, set them as following.

```
biml_dir <- 'your/directory'
```

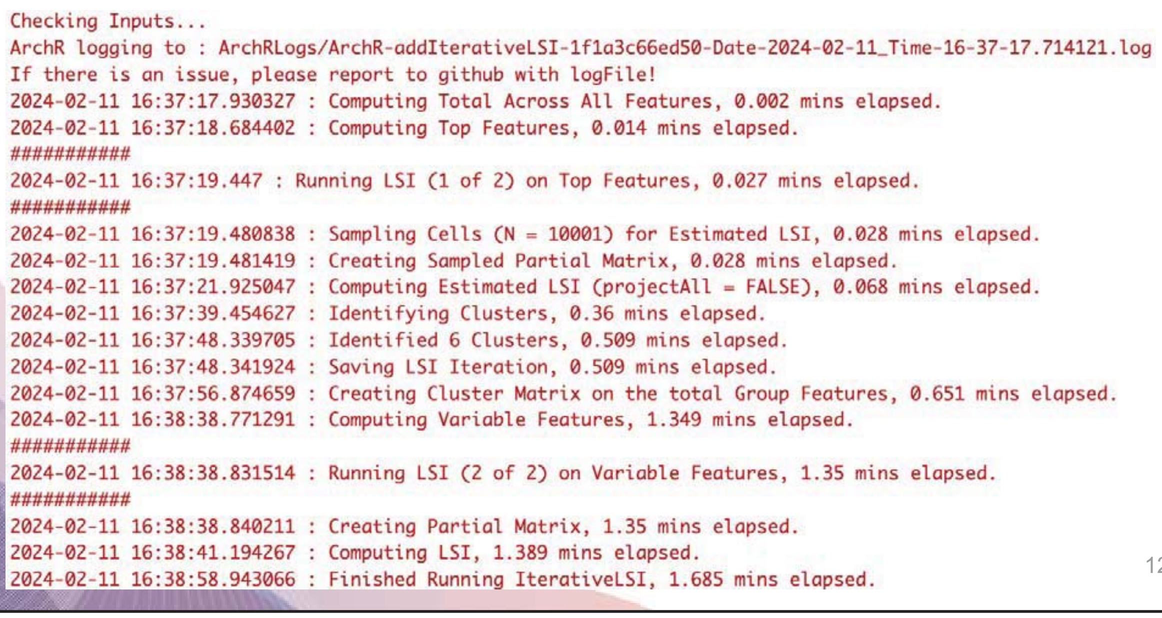
Load an ArchRProject

```
proj <- loadArchRProject(paste0(biml_dir, '/Save-Proj'))
```

Dimensionality reduction of scATAC-seq data

An iterative LSI dimensionality reduction via the addIterativeLSI() function.

```
proj <- addIterativeLSI(ArchRProj = proj, useMatrix = "TileMatrix", name = "IterativeLSI")

# saveArchRProject(ArchRProj = proj, outputDirectory = "01_Dimensionality_reduction", load
= FALSE)
# proj <- loadArchRProject(paste0(biml_dir, '/01_Dimensionality_reduction'))
```

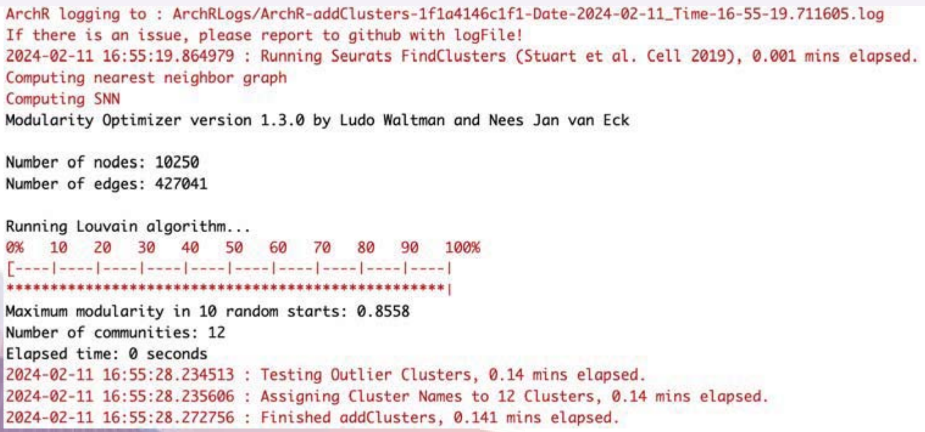


```
Checking Inputs...
ArchR logging to : ArchRLogs/ArchR-addIterativeLSI-1f1a3c66ed50-Date-2024-02-11_Time-16-37-17.714121.log
If there is an issue, please report to github with logFile!
2024-02-11 16:37:17.930327 : Computing Total Across All Features, 0.002 mins elapsed.
2024-02-11 16:37:18.684402 : Computing Top Features, 0.014 mins elapsed.
###########
2024-02-11 16:37:19.447 : Running LSI (1 of 2) on Top Features, 0.027 mins elapsed.
###########
2024-02-11 16:37:19.480838 : Sampling Cells (N = 10001) for Estimated LSI, 0.028 mins elapsed.
2024-02-11 16:37:19.481419 : Creating Sampled Partial Matrix, 0.028 mins elapsed.
2024-02-11 16:37:21.925047 : Computing Estimated LSI (projectAll = FALSE), 0.068 mins elapsed.
2024-02-11 16:37:39.454627 : Identifying Clusters, 0.36 mins elapsed.
2024-02-11 16:37:48.339705 : Identified 6 Clusters, 0.509 mins elapsed.
2024-02-11 16:37:48.341924 : Saving LSI Iteration, 0.509 mins elapsed.
2024-02-11 16:37:56.874659 : Creating Cluster Matrix on the total Group Features, 0.651 mins elapsed.
2024-02-11 16:38:38.771291 : Computing Variable Features, 1.349 mins elapsed.
###########
2024-02-11 16:38:38.831514 : Running LSI (2 of 2) on Variable Features, 1.35 mins elapsed.
###########
2024-02-11 16:38:38.840211 : Creating Partial Matrix, 1.35 mins elapsed.
2024-02-11 16:38:41.194267 : Computing LSI, 1.389 mins elapsed.
2024-02-11 16:38:58.943066 : Finished Running IterativeLSI, 1.685 mins elapsed.
```

Clustering with ArchR and Seurat

Using the graph clustering approach implemented by `Seurat::FindClusters()`, clustering is performed.

```
proj <- addClusters(
    input = proj,
    reducedDims = "IterativeLSI",
    method = "Seurat",
    name = "Clusters",
    resolution = 0.8
)

# saveArchRProject(ArchRProj = proj, outputDirectory = "02_Clustering", load =
FALSE)
# proj <- loadArchRProject(paste0(biml_dir, '/02_Clustering'))
```

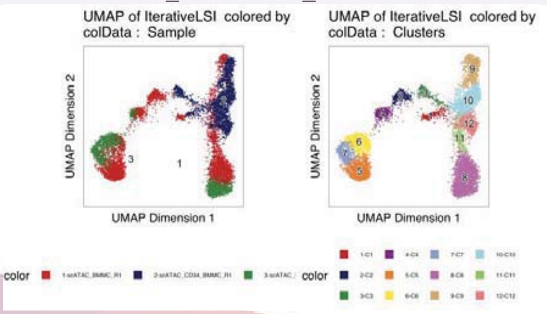


```
ArchR logging to : ArchRLogs/ArchR-addClusters-1f1a4146c1f1-Date-2024-02-11_Time-16-55-19.711605.log
If there is an issue, please report to github with logFile!
2024-02-11 16:55:19.864979 : Running Seurats FindClusters (Stuart et al. Cell 2019), 0.001 mins elapsed.
Computing nearest neighbor graph
Computing SNN
Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck

Number of nodes: 10250
Number of edges: 427041

Running Louvain algorithm...
0%   10   20   30   40   50   60   70   80   90   100%
[----|----|----|----|----|----|----|----|----|----|
**************************************************|
Maximum modularity in 10 random starts: 0.8558
Number of communities: 12
Elapsed time: 0 seconds
2024-02-11 16:55:28.234513 : Testing Outlier Clusters, 0.14 mins elapsed.
2024-02-11 16:55:28.235606 : Assigning Cluster Names to 12 Clusters, 0.14 mins elapsed.
2024-02-11 16:55:28.272756 : Finished addClusters, 0.141 mins elapsed.
```

Visualization in a Two-dimensional space

Visualization of single cell in a Two-dimensional **UMAP** space.

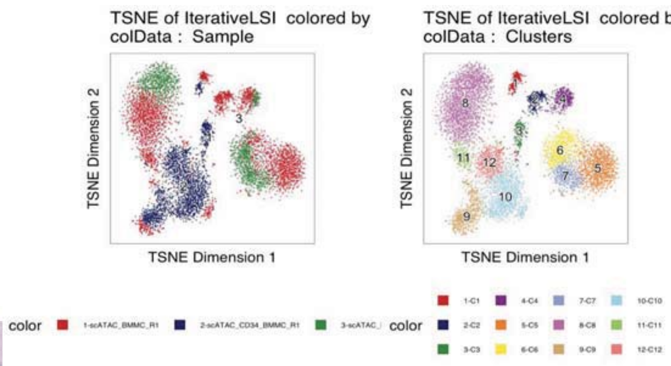
```
proj <- addUMAP(ArchRProj = proj, reducedDims = "IterativeLSI")

# UMAP colored by the Sample
p1 <- plotEmbedding(ArchRProj = proj, colorBy = "cellColData", name = "Sample",
embedding = "UMAP")

# UMAP colored by the Clusters
p2 <- plotEmbedding(ArchRProj = proj, colorBy = "cellColData", name = "Clusters",
embedding = "UMAP")
ggAlignPlots(p1, p2, type = "h")

# Save a plot
plotPDF(p1,p2, name = "Plot-UMAP-Sample-Clusters.pdf",
        ArchRProj = proj, addDOC = FALSE, width = 5, height = 5)

# saveArchRProject(ArchRProj = proj, outputDirectory = "03_UMAP", load = FALSE)
# proj <- loadArchRProject(paste0(biml_dir, '/03_UMAP'))
```



Visualization in a Two-dimensional space

Visualization of single cell in a Two-dimensional **tSNE** space. - **Optional**

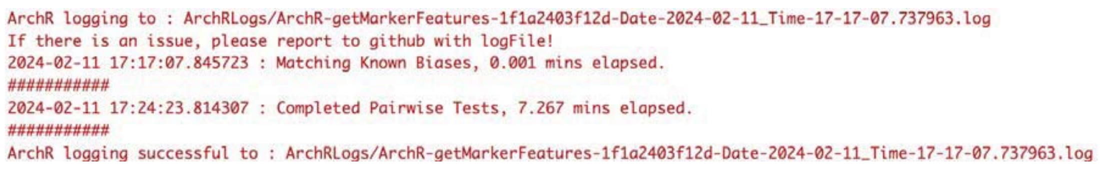
```
proj <- addTSNE(ArchRProj = proj, reducedDims = "IterativeLSI", name = "TSNE")

# UMAP colored by the Sample
p1 <- plotEmbedding(ArchRProj = proj, colorBy = "cellColData", name = "Sample",
embedding = "TSNE")

# UMAP colored by the Clusters
p2 <- plotEmbedding(ArchRProj = proj, colorBy = "cellColData", name = "Clusters",
embedding = "TSNE")
ggAlignPlots(p1, p2, type = "h")

# Save a plot
plotPDF(p1,p2, name = "Plot-TSNE-Sample-Clusters.pdf",
        ArchRProj = proj, addDOC = FALSE, width = 5, height = 5)
```



Identifying Marker genes for each cluster

This function takes several minutes depending on the computational resource.

```
# Do not run below
# markersGS <- getMarkerFeatures(
#   ArchRProj = proj,
#   useMatrix = "GeneScoreMatrix",
#   groupBy = "Clusters",
#   bias = c("TSSEnrichment", "log10(nFrags)"),
#   testMethod = "wilcoxon"
# )
# saveRDS(markersGS, file = paste0(biml_dir, '/2024_BIML_markerGS.rds')) # Optional
# Run below for the downstream tutorial
markersGS <- readRDS(file = paste0(biml_dir, '/2024_BIML_markerGS.rds'))
```



```
ArchR logging to : ArchRLogs/ArchR-getMarkerFeatures-1f1a2403f12d-Date-2024-02-11_Time-17-17-07.737963.log
If there is an issue, please report to github with logFile!
2024-02-11 17:17:07.845723 : Matching Known Biases, 0.001 mins elapsed.
###########
2024-02-11 17:24:23.814307 : Completed Pairwise Tests, 7.267 mins elapsed.
###########
ArchR logging successful to : ArchRLogs/ArchR-getMarkerFeatures-1f1a2403f12d-Date-2024-02-11_Time-17-17-07.737963.log
```

This function returns a `SummarizedExperiment` object containing relevant information on the marker features identified.

Get the marker genes for the Cluster 6

```
markerList <- getMarkers(markersGS, cutOff = "FDR <= 0.01 & Log2FC >= 1.25")
markerList$C6
markerList$C6 %>% as.data.frame() %>% dplyr::arrange(desc(Log2FC)) %>%
dplyr::pull(name) %>% head(10)
```

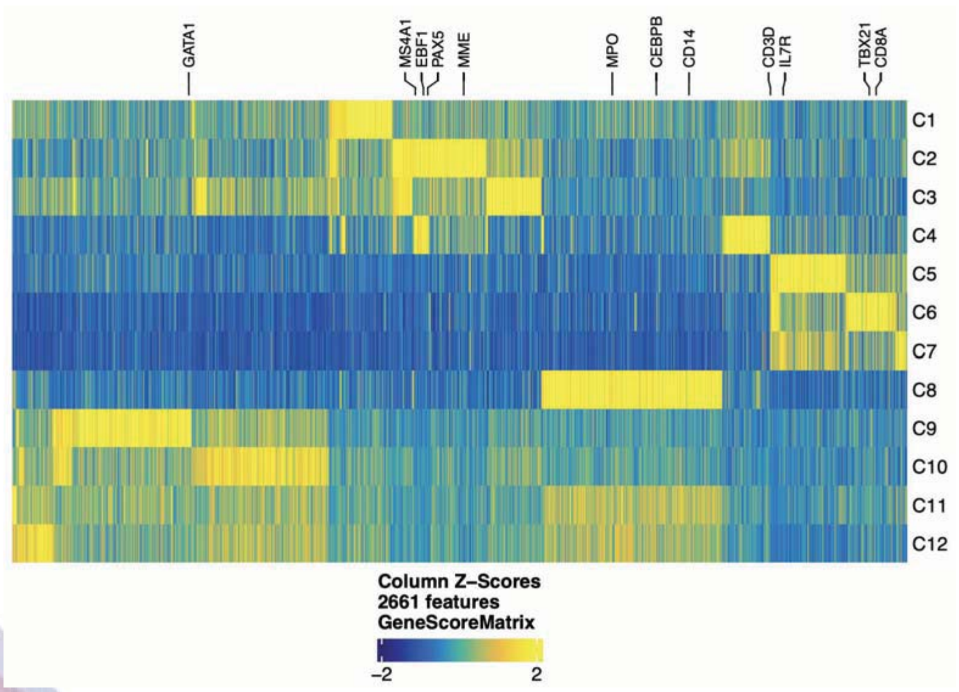
Visualize Gene scores - Heatmap

We can visualize gene scores by creating a **heatmap**

```
markerGenes  <- c(
  "CD34", #Early Progenitor
  "GATA1", #Erythroid
  "PAX5", "MS4A1", "EBF1", "MME", #B-Cell Trajectory
  "CD14", "CEBPB", "MPO", #Monocytes
  "IRF8",
  "CD3D", "CD8A", "TBX21", "IL7R" #TCells
)

heatmapGS <- markerHeatmap(seMarker = markersGS, cutOff = "FDR <= 0.01 & Log2FC >=
1.25", labelMarkers = markerGenes,transpose = TRUE)

ComplexHeatmap::draw(heatmapGS, heatmap_legend_side = "bot", annotation_legend_side
= "bot")
plotPDF(heatmapGS, name = "GeneScores-Marker-Heatmap", width = 8, height = 6,
ArchRProj = proj, addDOC = FALSE)
```

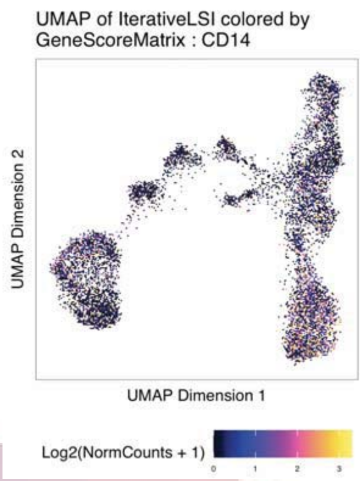
Visualize Gene scores - Heatmap

We can visualize gene scores by creating a **heatmap**



Visualize Gene scores – on an Embedding

```
markerGenes  <- c(
  "CD34",  #Early Progenitor
  "GATA1", #Erythroid
  "PAX5", "MS4A1", "MME", #B-Cell Trajectory
  "CD14", "MPO", #Monocytes
  "CD3D", "CD8A"#TCells
)

p <- plotEmbedding(ArchRProj = proj, colorBy = "GeneScoreMatrix", name =
markerGenes, embedding = "UMAP", imputeWeights = getImputeWeights(proj))

p$CD14
```



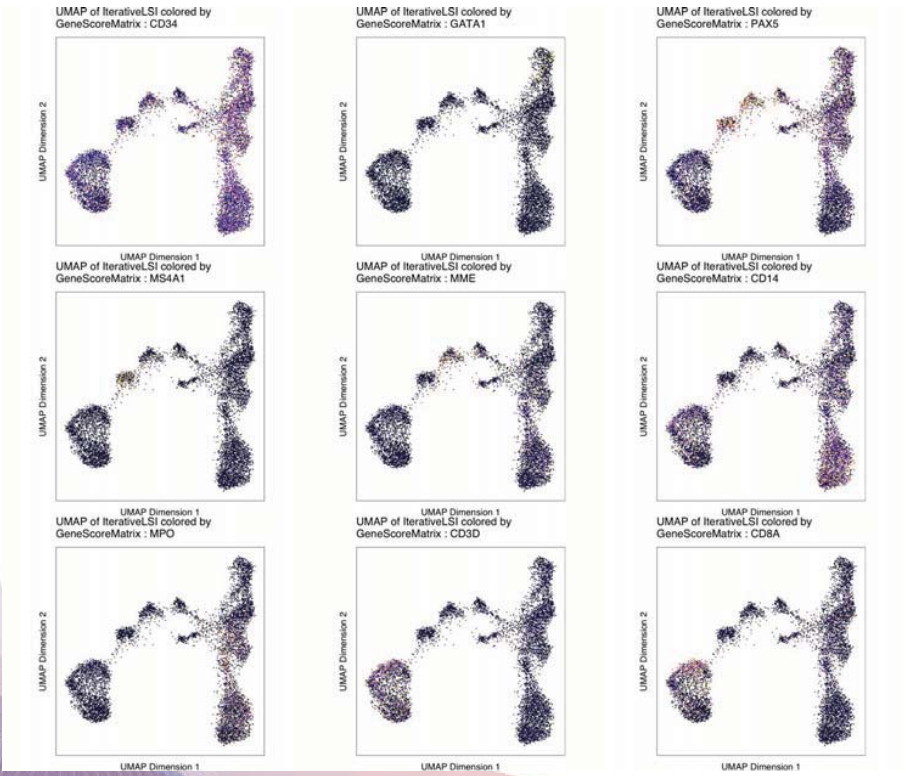
Visualize Gene scores – on an Embedding

To plot all genes we can use `cowplot` to arrange the various marker genes into a single plot.

```
p2 <- lapply(p, function(x){
    x + guides(color = FALSE, fill = FALSE) +
    theme_ArchR(baseSize = 6.5) +
    theme(plot.margin = unit(c(0, 0, 0, 0), "cm")) +
    theme(
        axis.text.x=element_blank(),
        axis.ticks.x=element_blank(),
        axis.text.y=element_blank(),
        axis.ticks.y=element_blank()
    )
})
do.call(cowplot::plot_grid, c(list(ncol = 3),p2))
```

Visualize Gene scores – on an Embedding

To plot all genes we can use `cowplot` to arrange the various marker genes into a single plot.

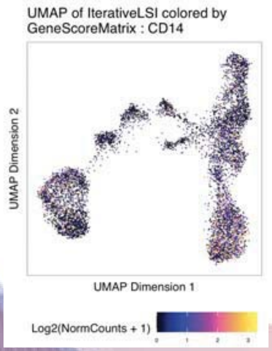


Marker Genes imputation with MAGIC

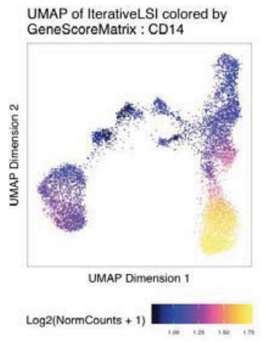
We use **MAGIC** to impute gene scores by smoothing signal across nearby cells.

```
proj <- addImputeWeights(proj)

p <- plotEmbedding(
  ArchRProj = proj,
  colorBy = "GeneScoreMatrix",
  name = 'CD14',
  embedding = "UMAP",
  imputeWeights = getImputeWeights(proj)
)

p
```

No imputation



After the imputation

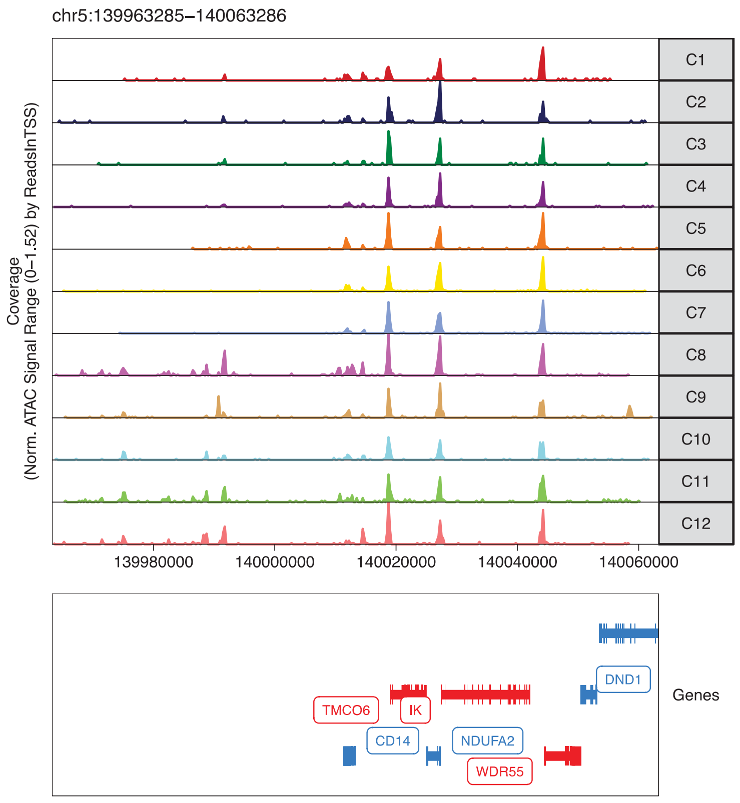


Visualizing Genome Browser Tracks

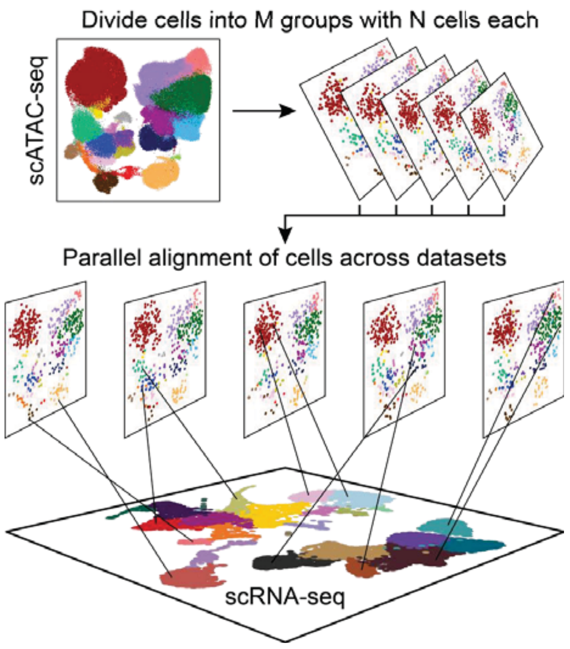
In addition to plotting gene scores per cell as a UMAP overlay, we can browse the local chromatin accessibility at these marker genes on a per cluster basis with genome browser tracks.

```
p <- plotBrowserTrack(
  ArchRProj = proj,
  groupBy = "Clusters",
  geneSymbol = markerGenes,
  upstream = 50000,
  downstream = 50000
)

grid::grid.newpage()
grid::grid.draw(p$CD14)

plotPDF(plotList = p,
        name = "Plot-Tracks-Marker-Genes.pdf",
        ArchRProj = proj,
        addDOC = FALSE, width = 5, height = 5)
```

Visualizing Genome Browser Tracks



Defining Cluster Identity with scRNA-seq data


Divide cells into M groups with N cells each
scATAC-seq
Parallel alignment of cells across datasets
scRNA-seq

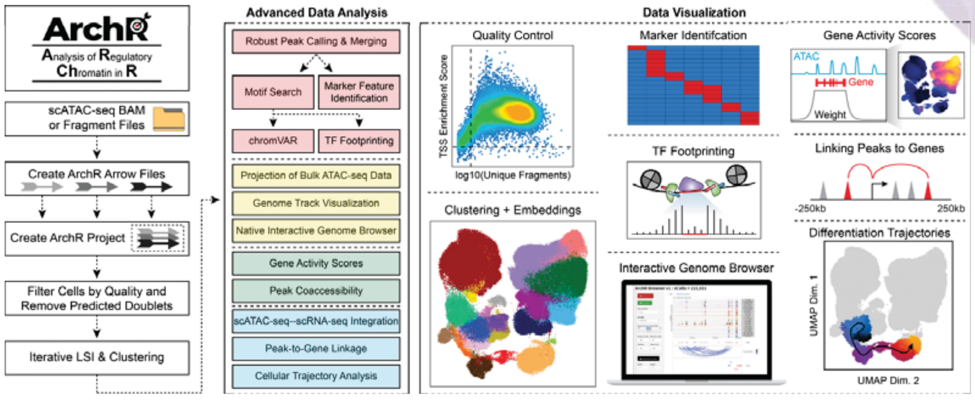
- ✓ In addition to allowing cluster identity assignment with gene scores, **ArchR** also enables integration with scRNA-seq.
- ✓ `FindTransferAnchors()` function from the **Seurat** package allows you to align data across two datasets.

Cross-platform linkage of scATAC-seq cells with scRNA-seq cells - "DO NOT RUN"

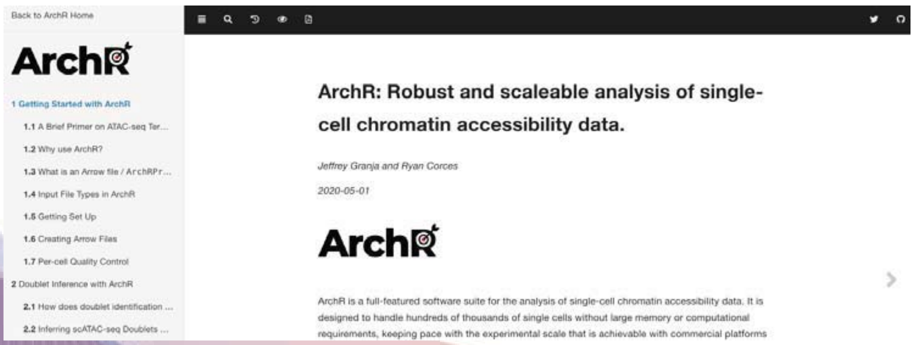
Download scRNA-seq data and Integration of scRNA-cells and scATAC-seq cells

```
if(!file.exists("scRNA-Hematopoiesis-Granja-2019.rds")){
  download.file(
    url = "https://jeffgranja.s3.amazonaws.com/ArchR/TestData/scRNA-Hematopoiesis-
Granja-2019.rds",
    destfile = "scRNA-Hematopoiesis-Granja-2019.rds"
  )
}

seRNA <- readRDS("scRNA-Hematopoiesis-Granja-2019.rds")
seRNA

colnames(colData(seRNA))

table(colData(seRNA)$BioClassification)

proj <- addGeneIntegrationMatrix(
  ArchRProj = proj,
  useMatrix = "GeneScoreMatrix",
  matrixName = "GeneIntegrationMatrix",
  reducedDims = "IterativeLSI",
  seRNA = seRNA,
  addToArrow = FALSE,
  groupRNA = "BioClassification",
  nameCell = "predictedCell_Un",
  nameGroup = "predictedGroup_Un",
  nameScore = "predictedScore_Un"
)
```

There are more advanced data analysis


ArchR
Analysis of Regulatory Chromatin in R
scATAC-seq BAM or Fragment Files
Create ArchR Arrow Files
Create ArchR Project
Filter Cells by Quality and Remove Predicted Doublets
Iterative LSI & Clustering
Advanced Data Analysis
Robust Peak Calling & Merging
Motif Search
Marker Feature Identification
chromVAR
TF Footprinting
Projection of Bulk ATAC-seq Data
Genome Track Visualization
Native Interactive Genome Browser
Gene Activity Scores
Peak Coaccessibility
scATAC-seq–scRNA-seq Integration
Peak-to-Gene Linkage
Cellular Trajectory Analysis
Data Visualization
Quality Control
TSS Enrichment Score
log10(Unique Fragments)
Marker Identifcation
Gene Activity Scores
ATAC
Gene
Weight
TF Footprinting
Linking Peaks to Genes
-250kb
250kb
Clustering + Embeddings
Interactive Genome Browser
Differentiation Trajectories
UMAP Dim. 1
UMAP Dim. 2

Visit following link of full manual for the advanced data analysis:
https://www.archrproject.com/bookdown/index.html



Back to ArchR Home

ArchR

1 Getting Started with ArchR
1.1 A Brief Primer on ATAC-seq Ter...
1.2 Why use ArchR?
1.3 What is an Arrow file / ArchRPr...
1.4 Input File Types in ArchR
1.5 Getting Set Up
1.6 Creating Arrow Files
1.7 Per-cell Quality Control
2 Doublet Inference with ArchR
2.1 How does doublet identification ...
2.2 Inferring scATAC-seq Doublets ...

ArchR: Robust and scaleable analysis of single-cell chromatin accessibility data.

Jeffrey Granja and Ryan Corces

2020-05-01

ArchR

ArchR is a full-featured software suite for the analysis of single-cell chromatin accessibility data. It is designed to handle hundreds of thousands of single cells without large memory or computational requirements, keeping pace with the experimental scale that is achievable with commercial platforms

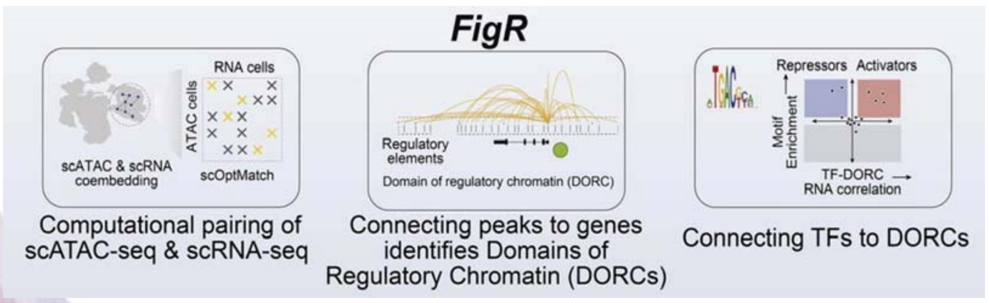
10. FigR

What is FigR?

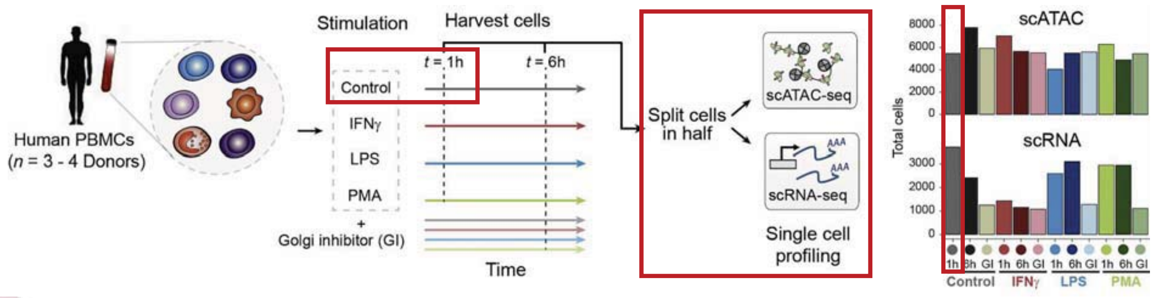
- Functional inference of gene regulation (FigR) is a framework to

  (1) computationally pair scATAC-seq with scRNA-seq cells by scOptMatch,
  (2) infers cis-regulatory interactions,
  (3) defines a TF-gene GRN

- Utilizing these paired multi-omics data, FigR define domains of regulatory chromatin (DORCs) of immune stimulation and find that cells alter chromatin accessibility and gene expression at timescales of minutes.


FigR
RNA cells
ATAC cells
scATAC & scRNA coembedding
scOptMatch
Computational pairing of scATAC-seq & scRNA-seq
Regulatory elements
Domain of regulatory chromatin (DORC)
Connecting peaks to genes identifies Domains of Regulatory Chromatin (DORCs)
Repressors
Activators
Motif Enrichment
TF-DORC RNA correlation
Connecting TFs to DORCs

Functional inference of gene regulation using scRNA-seq and scATAC-seq from resting immune cells



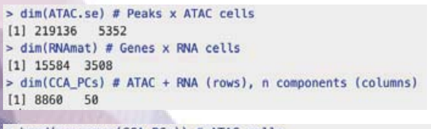
Installation of FigR

```
devtools::install_github("buenrostrolab/FigR")

BiocManager::install("BSgenome.Hsapiens.UCSC.hg19")
```

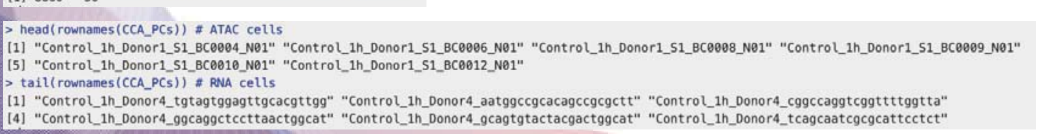
Loading ATAC-seq and RNA-seq data

Download data from Dropbox link, and unzip the data.
https://www.dropbox.com/scl/fi/q63f4wr4jlwtva7z72i9g/FigR_stim.zip?rlkey=gibefa8gdtj4z to78rnvmuym1&dl=0

```
library(doParallel)
library(BuenColors)
library(FigR)
library(BSgenome.Hsapiens.UCSC.hg19)

setwd("directory")
ATAC.se <- readRDS(”./FigR_stim/control1h_PBMC_atac_SE.rds")
RNAmat <- readRDS(”./FigR_stim/control1h_PBMC_RNAnorm.rds")
CCA_PCs <- readRDS(”./FigR_stim/control1h_PBMC_atac_rna_CCA_l2.rds”)

dim(ATAC.se) # Peaks x ATAC cells
dim(RNAmat) # Genes x RNA cells
dim(CCA_PCs) # ATAC + RNA (rows), n components (columns)
head(rownames(CCA_PCs)) # ATAC cells
tail(rownames(CCA_PCs)) # RNA cells
```



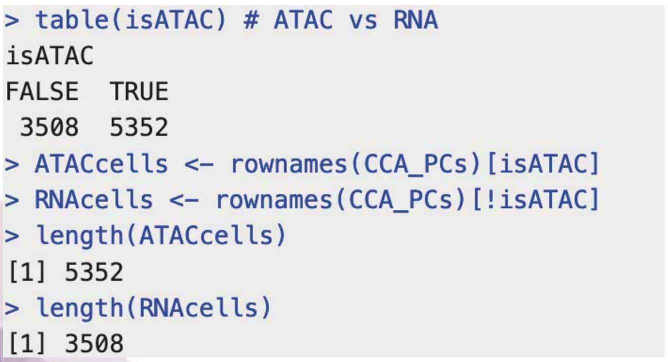
```
> dim(ATAC.se) # Peaks x ATAC cells
[1] 219136   5352
> dim(RNAmat) # Genes x RNA cells
[1] 15584  3508
> dim(CCA_PCs) # ATAC + RNA (rows), n components (columns)
[1] 8860   50
```



```
> head(rownames(CCA_PCs)) # ATAC cells
[1] "Control_1h_Donor1_S1_BC0004_N01" "Control_1h_Donor1_S1_BC0006_N01" "Control_1h_Donor1_S1_BC0008_N01" "Control_1h_Donor1_S1_BC0009_N01"
[5] "Control_1h_Donor1_S1_BC0010_N01" "Control_1h_Donor1_S1_BC0012_N01"
> tail(rownames(CCA_PCs)) # RNA cells
[1] "Control_1h_Donor4_tgtagtggagttgcacgttgg" "Control_1h_Donor4_aatggccgcacagccgcgctt" "Control_1h_Donor4_cggccaggtcggttttggtta"
[4] "Control_1h_Donor4_ggcaggctccttaactggcat" "Control_1h_Donor4_gcagtgtactacgactggcat" "Control_1h_Donor4_tcagcaatcgcgcattcctct"
```

Loading ATAC-seq and RNA-seq data

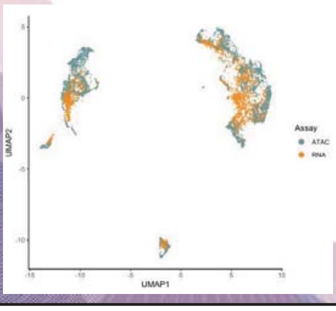
```
isATAC <- grepl("BC",rownames(CCA_PCs))
table(isATAC) # ATAC vs RNA

ATACcells <- rownames(CCA_PCs)[isATAC]
RNAcells <- rownames(CCA_PCs)[!isATAC]

length(ATACcells)
length(RNAcells)
```



```
> table(isATAC) # ATAC vs RNA
isATAC
FALSE  TRUE
 3508  5352
> ATACcells <- rownames(CCA_PCs)[isATAC]
> RNAcells <- rownames(CCA_PCs)[!isATAC]
> length(ATACcells)
[1] 5352
> length(RNAcells)
[1] 3508
```

Visualizing UMAP of the ATAC-RNA cell

```
nPCs <- 20 # Num CCA PCs to use when running UMAP / pairing

set.seed(123)
umap.out <- uwot::umap(CCA_PCs[,1:nPCs],
                       metric="cosine",
                       n_neighbors=30)

umap.d <- as.data.frame(umap.out)
colnames(umap.d) <- c("UMAP1","UMAP2")
rownames(umap.d) <- rownames(CCA_PCs)

umap.d$Assay <- ifelse(isATAC,"ATAC","RNA")

BuenColors::shuf(umap.d) %>%
  ggplot(aes(UMAP1,UMAP2,color=Assay)) +
  geom_point(size=0.1) +
  theme_classic() + theme_classic() +
  scale_color_manual(values = c("cadetblue","darkorange"))+
  guides(colour = guide_legend(override.aes = list(size=3))))
```



Pairing cells using scOptMatch

```
# Get PCs for each data
ATAC_PCs <- CCA_PCs[isATAC,]
RNA_PCs <- CCA_PCs[!isATAC,]
dim(ATAC_PCs)
dim(RNA_PCs)
# Pair cells using scOptMatch
pairing <- pairCells(ATAC = ATAC_PCs,
                     RNA = RNA_PCs,
                     keepUnique = TRUE)
Dim(pairing)
```

```
> dim(ATAC_PCs)
[1] 5352   50
> dim(RNA_PCs)
[1] 3508   50
```

5,352 cells, 50 PCs

3,508 cells, 50 PCs

```
> dim(pairing)
[1] 4912    3
```

Distance between two cells

Cells from ATAC-seq

Cells from RNA-seq

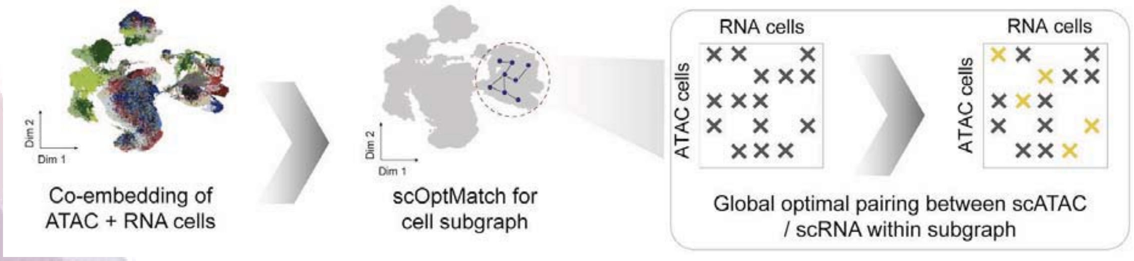
```
> head(pairing)
# A tibble: 6 × 3
# Groups:   ATAC [6]
  ATAC                            RNA                                     dist
  <chr>                           <chr>                                  <dbl>
1 Control_1h_Donor3_S1_BC1625_N01 Control_1h_Donor2_tggccttcagagagtccagtt 0.981
2 Control_1h_Donor3_S1_BC0358_N01 Control_1h_Donor4_agccgcctaagaggagtttct 1.15
3 Control_1h_Donor4_S1_BC0982_N01 Control_1h_Donor4_tgcgagcacgtattccaagct 0.824
4 Control_1h_Donor3_S1_BC0284_N01 Control_1h_Donor3_ctaactcagtttctggtcgta 0.926
5 Control_1h_Donor1_S1_BC0087_N01 Control_1h_Donor2_ttaagcggagaggtagccgcc 0.959
6 Control_1h_Donor2_S1_BC0879_N01 Control_1h_Donor3_gtgcattgtgtccaaactctt 1.10
```

Cell pairing by scOptMatch

1. Creating a shared co-embedding of scATAC-seq and scRNA-seq cells using canonical correlation analysis (CCA)

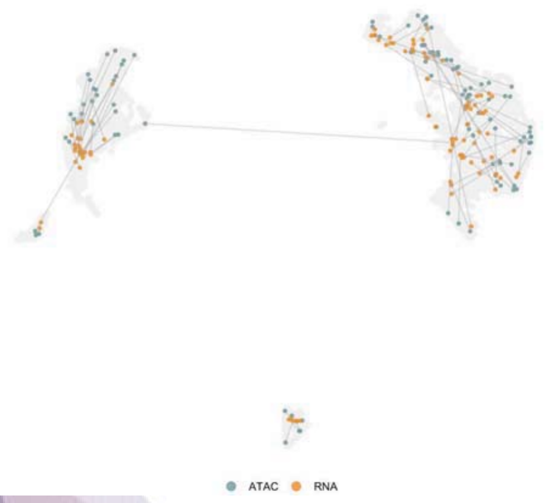
2. Sub-clustering the entire cell space and constructing a cell kNN graph between ATAC and RNA cells in the co-embedded space

3. Global optimal pairing between scATAC and scRNA within subgraph


Dim 2
Dim 1
Co-embedding of ATAC + RNA cells
Dim 2
Dim 1
scOptMatch for cell subgraph
RNA cells
ATAC cells
RNA cells
ATAC cells
Global optimal pairing between scATAC / scRNA within subgraph

Visualizing ATAC-RNA pairs on the CCA UMAP

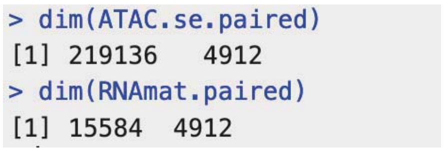
```
library(ggrastr)
plotPairs(ATAC = pairing$ATAC,
          RNA=pairing$RNA,
          max.show = 100,
          umap.df = umap.d)
```



Getting count object for the ATAC-RNA paired cells

```
ATAC.se.paired <- ATAC.se[,pairing$ATAC]
RNAmat.paired <- RNAmat[,pairing$RNA]

dim(ATAC.se.paired)
dim(RNAmat.paired)
```

4,912 cells included in pairs are assigned to each object



```
> dim(ATAC.se.paired)
[1] 219136   4912
> dim(RNAmat.paired)
[1] 15584   4912
```

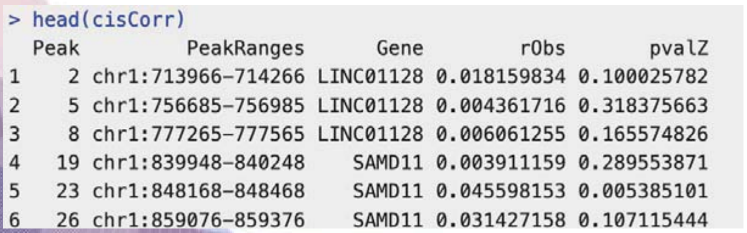
Peak-gene association testing

# Compute correlation between RNA expression and peak accessibility for peaks falling within a window around each gene.

# Do not run this code
```
#cisCorr <- runGenePeakcorr(ATAC.se = ATAC.se.paired,
#                            RNAmat = RNAmat.paired,
#                            genome = "hg19",
#                            nCores = 4,
#                            p.cut = NULL,
#                            n_bg = 100) # the number of background correlations to
                                          compute per gene-peak pair

cisCorr <- readRDS("./FigR_stim/control1h_PBMC_cisCor.rds")
```

# head for cisCorr
```
head(cisCorr)
```

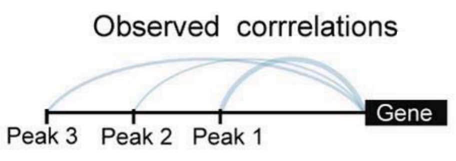
- This step use background peak correlations for significance testing.



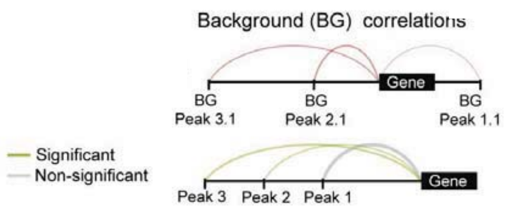
```
> head(cisCorr)
  Peak            PeakRanges       Gene        rObs       pvalZ
1    2 chr1:713966-714266 LINC01128 0.018159834 0.100025782
2    5 chr1:756685-756985 LINC01128 0.004361716 0.318375663
3    8 chr1:777265-777565 LINC01128 0.006061255 0.165574826
4   19 chr1:839948-840248    SAMD11 0.003911159 0.289553871
5   23 chr1:848168-848468    SAMD11 0.045598153 0.005385101
6   26 chr1:859076-859376    SAMD11 0.031427158 0.107115444
```

Peak-gene association testing

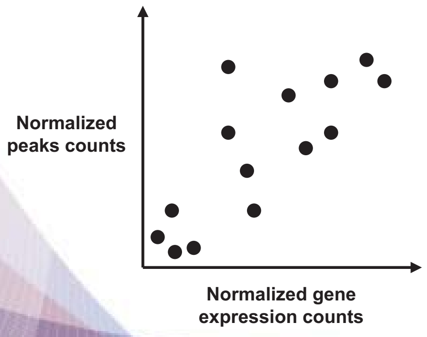
Compute peak-gene correlations



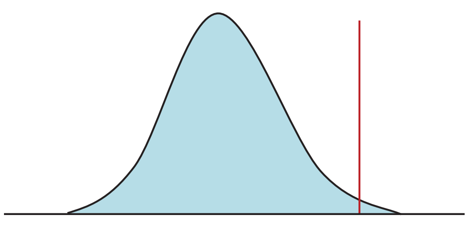
Determine significant peak-gene links



Peak-gene correlation
(Spearman correlation)



Background peak-gene correlation by permutation

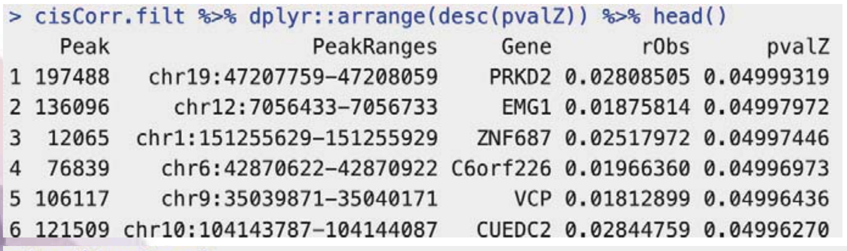


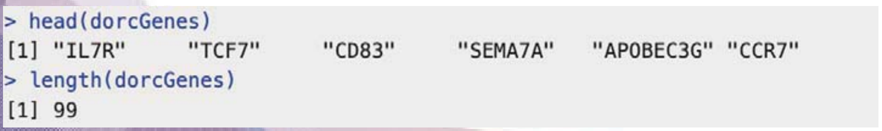
Determining DORCs

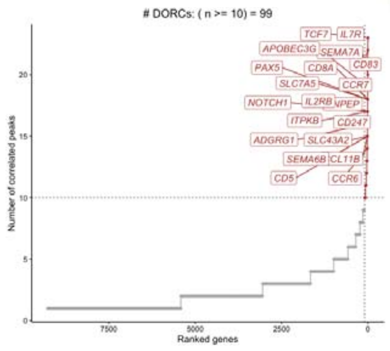
# Filtering DORCs by p-value
```
cisCorr.filt <- cisCorr %>% filter(pvalZ <= 0.05)
cisCorr.filt %>% dplyr::arrange(desc(pvalZ)) %>% head()
```

# Plotting DORCs
```
library(ggplot2)
dorcGenes <- dorcJPlot(dorcTab = cisCorr.filt,
                       cutoff = 10,
                       labelTop = 20,
                       returnGeneList = TRUE,
                       force=5)
head(dorcGenes)
length(dorcGenes)
```

- cutoff : the number of significant peaks needed to be called a DORC


```
> cisCorr.filt %>% dplyr::arrange(desc(pvalZ)) %>% head()
    Peak             PeakRanges    Gene       rObs      pvalZ
1 197488  chr19:47207759-47208059   PRKD2 0.02808505 0.04999319
2 136096    chr12:7056433-7056733    EMG1 0.01875814 0.04997972
3  12065  chr1:151255629-151255929  ZNF687 0.02517972 0.04997446
4  76839    chr6:42870622-42870922 C6orf226 0.01966360 0.04996973
5 106117    chr9:35039871-35040171     VCP 0.01812899 0.04996436
6 121509 chr10:104143787-104144087  CUEDC2 0.02844759 0.04996270
```

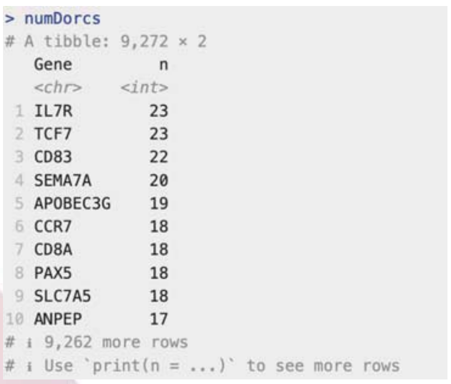

```
> head(dorcGenes)
[1] "IL7R"     "TCF7"     "CD83"     "SEMA7A"   "APOBEC3G" "CCR7"
> length(dorcGenes)
[1] 99
```



Summary of DORC

# To get the DORC accessibility scores, we can sum up the chromatin accessibility peak counts for peaks associated

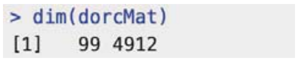
```
numDorcs <- cisCorr.filt %>% group_by(Gene) %>% tally() %>% arrange(desc(n))
numDorcs
```


```
> numDorcs
# A tibble: 9,272 × 2
   Gene         n
   <chr>    <int>
 1 IL7R        23
 2 TCF7        23
 3 CD83        22
 4 SEMA7A      20
 5 APOBEC3G    19
 6 CCR7        18
 7 CD8A        18
 8 PAX5        18
 9 SLC7A5      18
10 ANPEP       17
# ℹ 9,262 more rows
# ℹ Use `print(n = ...)` to see more rows
```

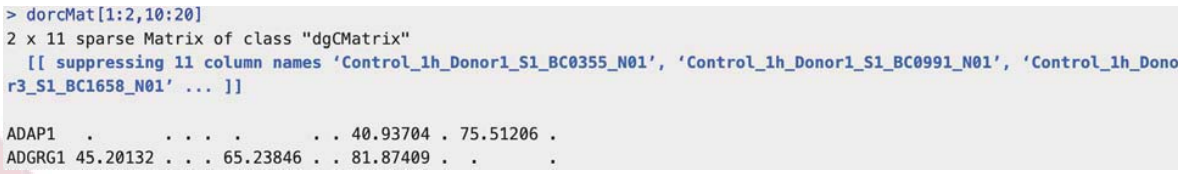
These genes are unstimulated (control) PBMCs, we expect most of these genes to be lineage-determining markers.

Calculating DORC scores

# Calculate DORC scores
```
dorcMat <- getDORCScores(ATAC.se = ATAC.se.paired,
                         dorcTab = cisCorr.filt,
                         geneList = dorcGenes,
                         nCores = 4)

dim(dorcMat)
dorcMat[1:2,10:20]
```


```
> dim(dorcMat)
[1]   99 4912
```

99 DORCs x 4,912 cells

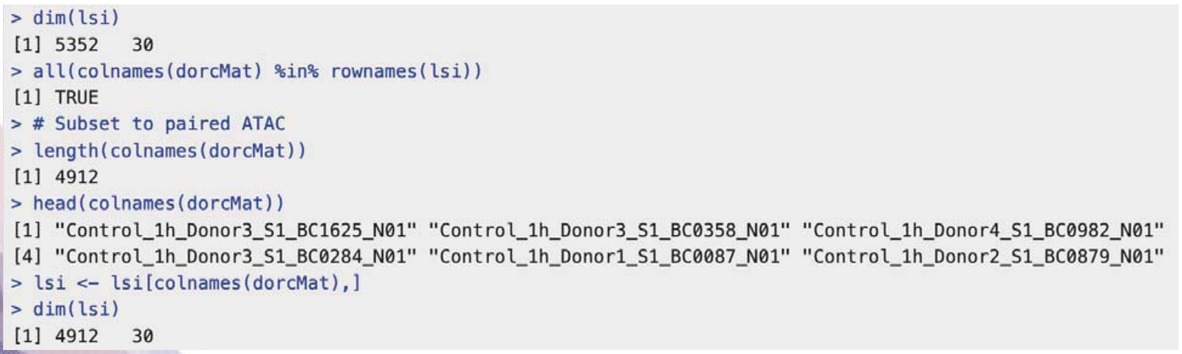

```
> dorcMat[1:2,10:20]
2 x 11 sparse Matrix of class "dgCMatrix"
  [[ suppressing 11 column names ‘Control_1h_Donor1_S1_BC0355_N01’, ‘Control_1h_Donor1_S1_BC0991_N01’, ‘Control_1h_Dono
r3_S1_BC1658_N01’ ... ]]

ADAP1   .        . . . .        . . 40.93704 . 75.51206 .
ADGRG1 45.20132 . . . 65.23846 . . 81.87409 .  .        .
```

Single cell DORC scores per gene were calculated as the sum of normalized scATAC-seq reads in peak counts (mean-centered) using the corresponding significantly correlated DORC-peaks for that gene.

Smoothing RNA using cell KNNs

```
lsi <- readRDS("./control1h_PBMC_atac_lsi.rds")
dim(lsi)
all(colnames(dorcMat) %in% rownames(lsi))
```

# Subset to paired ATAC
```
length(colnames(dorcMat))
head(colnames(dorcMat))
lsi <- lsi[colnames(dorcMat),]
lsi <- lsi[colnames(dorcMat),]
dim(lsi)
```


```
> dim(lsi)
[1] 5352   30
> all(colnames(dorcMat) %in% rownames(lsi))
[1] TRUE
> # Subset to paired ATAC
> length(colnames(dorcMat))
[1] 4912
> head(colnames(dorcMat))
[1] "Control_1h_Donor3_S1_BC1625_N01" "Control_1h_Donor3_S1_BC0358_N01" "Control_1h_Donor4_S1_BC0982_N01"
[4] "Control_1h_Donor3_S1_BC0284_N01" "Control_1h_Donor1_S1_BC0087_N01" "Control_1h_Donor2_S1_BC0879_N01"
> lsi <- lsi[colnames(dorcMat),]
> dim(lsi)
[1] 4912   30
```

Smoothing RNA using cell KNNs

# For 30 LSIs, get the nearest cell for each cell

```
cellkNN <- FNN::get.knn(lsi,k=30)$nn.index
dim(cellkNN)# 4912 cells, 30 LSIs
rownames(cellkNN) <- colnames(dorcMat)
```

# Smooth dorc scores using cell KNNs (k=30)

```
library(doParallel)
dorcMat.s <- smoothScoresNN(NNmat = cellkNN,mat = dorcMat,nCores = 4)
dim(dorcMat.s) # 99 DORCs, 4912 cells
```

	LSI1	…	LSI29	LSI30
Cell 1	2		4910	4911
…				
Cell 4912				

For 30 LSIs, the nearest cell for each cell

cellKNN (4,912 cells x 30 LSIs)

	Cell 1	…	Cell2	…	Cell 4910	Cell 4911	Cell 4912
DORC 1	2.34	…	4.12	…	1.34	4.66	3.63
…							
DORC 99							

Smooth score for {cell1, DORC1} = avg(4.12, … , 1.34, 4.66)

dorcMat (99 DORCs x 4,912 cells)

	Cell 1	…	Cell 4912
DORC 1	**avg(4.12, …, 1.34, 4.66)**	…	…
…			
DORC 99			

dorcMat.s (99 DORCs x 4,912 cells)

Smoothing RNA using cell KNNs

# Smooth RNA expression using cell KNNs (k=30) (Table1)

```
# Smooth RNA using cell KNNs
# This takes longer since it's all genes
colnames(RNAmat.paired) <- colnames(ATAC.se.paired)
RNAmat.s <- smoothScoresNN(NNmat = cellkNN,mat = RNAmat.paired,nCores = 4)

dim(RNAmat.s)
```

```
> dim(RNAmat.s)
[1] 15584  4912
```

Total RNA x cells

Visualizing DORC on UMAP

# Visualize DORC on pre-computed UMAP
# This is the ATAC UMAP shown in the paper (based on ATAC LSI)

```
umap.d <- as.data.frame(colData(ATAC.se.paired)[,c("UMAP1","UMAP2")])
```

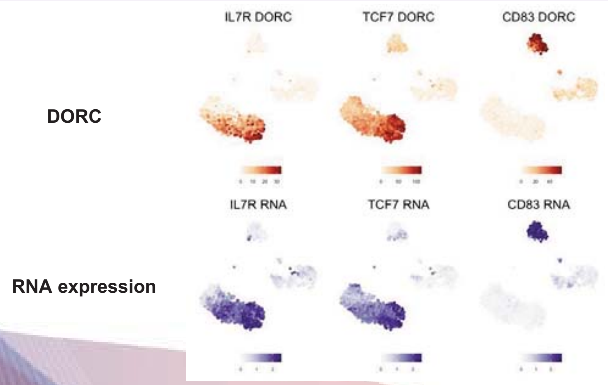
# DORC scores for top DORC(s)

```
myDORCs <- c("IL7R", "TCF7", "CD83")
dorcGGlist <- lapply(myDORCs,function(x) {
  plotMarker2D(umap.d,
               dorcMat.s,
               markers = x,
               maxCutoff = "q0.99",
               colorPalette = "brewer_heat"
  ) + ggtitle(paste0(x," DORC"))
})
```

Visualizing RNA expression for top DORCs

# Paired RNA expression of top DORCs
# Plot on the same reference ATAC UMAP

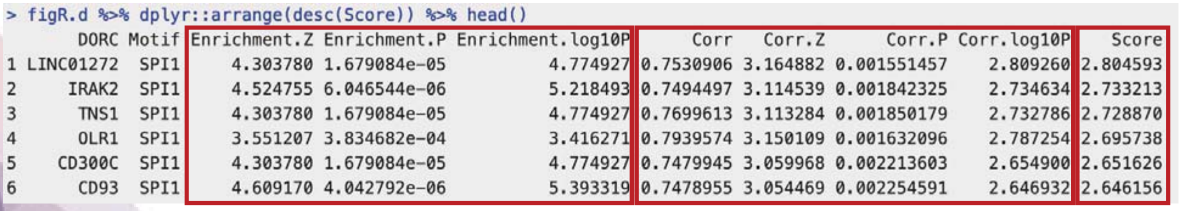
```
rnaGGlist <- lapply(myDORCs,function(x) {
  plotMarker2D(umap.d,
               RNAmat.s,
               markers = x,
               maxCutoff = "q0.99",
               colorPalette = "brewer_purple"
  ) + ggtitle(paste0(x," RNA"))
})

library(patchwork)
(dorcGGlist[[1]] + dorcGGlist[[2]] + dorcGGlist[[3]]) /  (rnaGGlist[[1]] +
rnaGGlist[[2]] + rnaGGlist[[3]])
```



TF-gene associations

```
# Determine TF-gene associations and inferring a regulatory network based on DORCs
# To determine TFs that are putative regulators (activators or repressors) of DORCs, a built-in reference motif database is used
# Do not run this code
#figR.d <- runFigRGRN(ATAC.se = ATAC.se.paired,
#                      dorcTab = cisCorr.filt, # Filtered peak-gene associations
#                      genome = "hg19",
#                      dorcMat = dorcMat.s,
#                      dorcK = 5,
#                      rnaMat = RNAmat.s,
#                      nCores = 4)

figR.d <- readRDS("./FigR_stim/control1h_PBMC_figR.rds")
```

Regulation score represents the intersection of motif-enriched and RNA-correlated TFs.



```
> figR.d %>% dplyr::arrange(desc(Score)) %>% head()
      DORC Motif Enrichment.Z Enrichment.P Enrichment.log10P      Corr   Corr.Z      Corr.P Corr.log10P    Score
1 LINC01272  SPI1     4.303780 1.679084e-05          4.774927 0.7530906 3.164882 0.001551457    2.809260 2.804593
2     IRAK2  SPI1     4.524755 6.046544e-06          5.218493 0.7494497 3.114539 0.001842325    2.734634 2.733213
3      TNS1  SPI1     4.303780 1.679084e-05          4.774927 0.7699613 3.113284 0.001850179    2.732786 2.728870
4      OLR1  SPI1     3.551207 3.834682e-04          3.416271 0.7939574 3.150109 0.001632096    2.787254 2.695738
5    CD300C  SPI1     4.303780 1.679084e-05          4.774927 0.7479945 3.059968 0.002213603    2.654900 2.651626
6      CD93  SPI1     4.609170 4.042792e-06          5.393319 0.7478955 3.054469 0.002254591    2.646932 2.646156
```

(a) Enrichment test using TF motif and DORC

(b) Correlation between TF RNA expression and DORC score

Regulation score

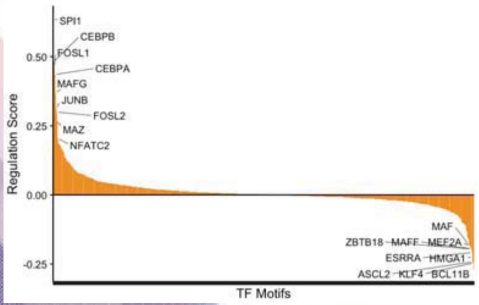
Regulation score calculated by combining (a) and (b)

$$Regulationscore = sign(Correlation)^{*} - \log 10^{*} \left[1 - (1 - P_{Enrichment})^{*} (1 - P_{Correlation})\right].$$

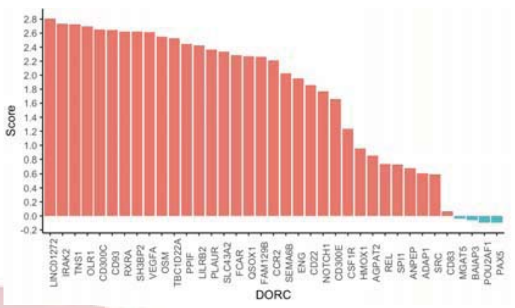
Visualizing FigR results

```
# Rank TF motifs by mean of regulation scores
rankDrivers(figR.d,rankBy = "meanScore")
# SPI1 gene
SPI1 <-figR.d %>% dplyr::filter(Motif == "SPI1") %>%
  dplyr::arrange(desc(Score)) %>%
  dplyr::filter(Score !=0)
SPI1$DORC <- factor(SPI1$DORC, levels = SPI1$DORC)
SPI1 <- SPI1 %>% dplyr::mutate(color = case_when(Score > 0 ~ "pos",
                                                  Score < 0 ~ "neg"))
SPI1$color <- factor(SPI1$color, c("pos", "zero", "neg"))
SPI1 %>% ggplot(aes(x= DORC, y=Score, fill = color)) +
  geom_bar (stat="identity",position = position_dodge(0.9)) +
  scale_y_continuous(breaks = seq(-2,5,0.2)) + theme_classic() +
  theme(axis.text.x = element_text(angle = 90))
```

Top 1 TF is SPI1.



In resting immune cells, SPI1 positively regulated LINC01272, IRAK2, etc.



Thank you!