BIOF3 组学数据分析

02 原始数据处理与 Cell Ranger

导出日期：2026年5月12日

02 原始数据处理与 Cell Ranger

测序下机拿到的是一堆 FASTQ 文件，里面按 read 顺序存着每条 reads 的碱基序列和质量值。要把这些原始 reads 变成"基因 × 细胞"的表达矩阵，需要经过三件事：识别每条 read 属于哪个细胞（barcode）、去除重复分子（UMI 去重）、比对到参考基因组并数每个基因有多少条 reads。对于 10x Chromium 技术平台，把这三件事打包起来做的软件就是 Cell Ranger。

本节介绍 10x 单细胞数据的基本结构、Cell Ranger 的工作流程和典型用法，以及产出之后如何用 Seurat / Scanpy 读进来。真正做分析需要约 64 GB 内存和若干小时 CPU 时间，如果你的机器达不到，这里贴的命令可以先放一放，直接用01 章下载好的 PBMC 3k 表达矩阵继续往下走。

10x 单细胞数据的基本结构

10x Chromium 用微流控芯片把细胞和含 barcode 的凝胶珠一起包进油滴（GEM），每个 GEM 里最多一个细胞和一颗珠子。同一细胞的所有 mRNA 都会被打上**相同的 cell barcode**，同一 mRNA 分子经过反转录后会携带**唯一的 UMI**。这两个标识让"哪条 reads 来自哪个细胞、哪条 reads 来自同一分子"变得可以在下游分析里重建出来。

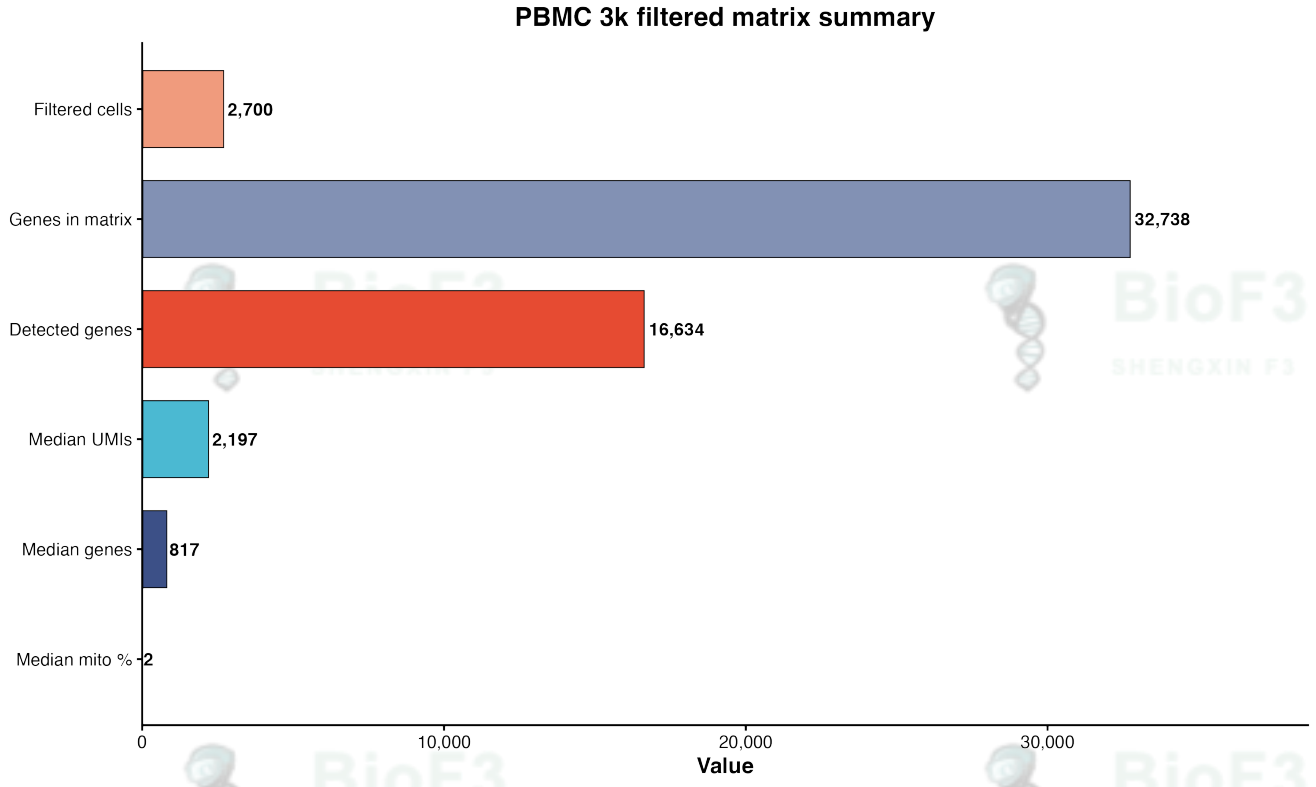

图 1：PBMC 3k 真实 filtered matrix 的关键统计，包括细胞数、矩阵基因数、检出基因数、中位 UMI、中位基因数和中位线粒体比例。

FASTQ 文件的组织方式

10x 的 FASTQ 常见命名是 `{sample}_S1_L001_R1_001.fastq.gz` 这种格式，三个角色：

- `R1`：16 bp 的 cell barcode + 10 bp 的 UMI（v3 chemistry；v2 是 14+10）+ Poly-T
- `R2`：测到的 cDNA 序列本体

- `I1`（可选）：样本索引，多样本混合上机时区分样本

图 2：FASTQ 文件结构示意图。展示了 Read 1 和 Read 2 中各组成部分的长度和位置。

单条 FASTQ record 的内容就是 4 行格式：

```
@序列ID
ATCGATCGATCG...
+
IIIIIIIIIIII...
```

Cell Ranger 做什么

Cell Ranger 的 `cellranger count` 命令把一组 FASTQ 变成一份可直接读进 Seurat / Scanpy 的表达矩阵。过程大致是：识别 cell barcode 并做纠错 → 去除 UMI 重复 → 用 STAR 比对 R2 到参考基因组 → 根据比对结果给每个 (cell, gene) 累加 UMI 数 → 用 EmptyDrops 等算法判断哪些 barcode 对应的是真实细胞。输出里最重要的是 `filtered_feature_bc_matrix/` 和 `web_summary.html`。

系统要求

Cell Ranger 8 只在 Linux 下官方支持（没有 macOS/Windows 版本）。一个常规 PBMC 样本大约要 8 核 CPU、64 GB 内存和几百 GB 磁盘；大样本或同时跑多个按比例加码。

安装 Cell Ranger

Cell Ranger 本身需要到 10x Genomics 下载页登录索取下载链接（每个链接都有时效）。拿到 URL 后：

```
wget -O cellranger-8.0.0.tar.gz \
  "你从 10x 下载页拿到的带签名的 URL"

tar -xzvf cellranger-8.0.0.tar.gz
export PATH=/path/to/cellranger-8.0.0:$PATH

cellranger --version
```

参考基因组按物种下载对应版本：

```
# 人 GRCh38
wget https://cf.10xgenomics.com/supp/cell-exp/refdata-gex-GRCh38-2024-A.tar.gz

# 鼠 mm10
wget https://cf.10xgenomics.com/supp/cell-exp/refdata-gex-mm10-2024-A.tar.gz

tar -xzvf refdata-gex-GRCh38-2024-A.tar.gz
```

跑一次 cellranger count

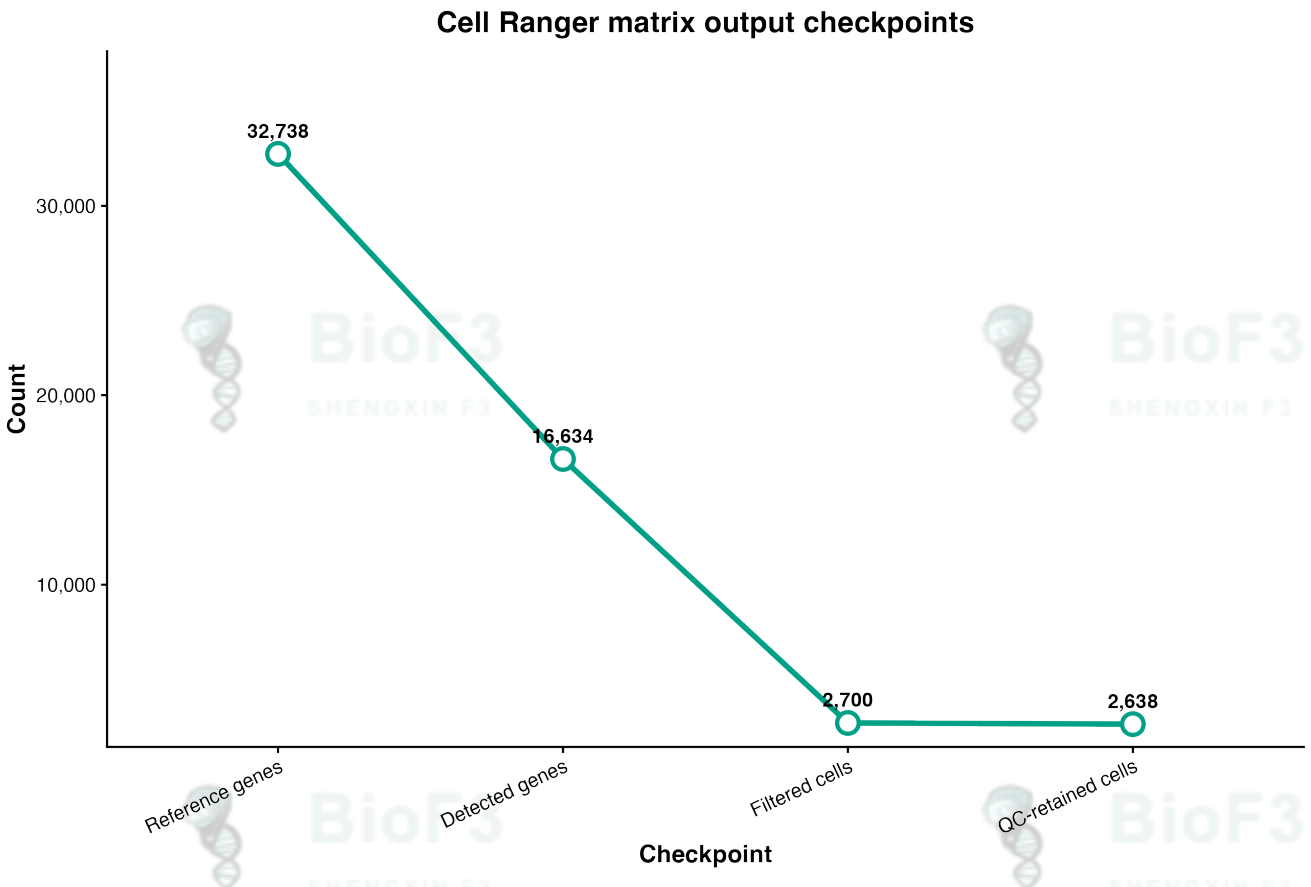

图 3：PBMC 3k 真实矩阵的输出检查点。图中展示参考基因数、实际检出基因数、filtered cell barcode 数和按本教程阈值保留的细胞数。

一个标准调用：

```
cellranger count \
  --id=PBMC_sample1 \
  --transcriptome=/data/refdata-gex-GRCh38-2024-A \
  --fastqs=/data/fastq/PBMC \
  --sample=PBMC_1 \
  --localcores=16 \
  --localmem=128
```

重要参数：

参数	作用	备注
`--id`	输出目录名	同时也是后续报告里的 sample ID
`--transcriptome`	参考基因组路径	必须和测序样本物种对得上
`--fastqs`	FASTQ 所在目录	多 lane 用逗号分隔，如 `/lane1,/lane2`
`--sample`	样本前缀	Cell Ranger 会匹配 `{sample}_S*_L*_R*_001.fastq.gz`
`--localcores`	CPU 核心数	通常 8-16，受机器限制
`--localmem`	内存上限（GB）	至少 64；高 UMI 样本更多
`--expect-cells`	预期细胞数	芯片说明书会给，不给默认按 EmptyDrops 自适应
`--chemistry`	chemistry 版本	大多数情况下可以留 `auto`

Cell Ranger 除了 `count` 还有 `aggr`（多样本合并）、`reanalyze`（改参数重分析）、`mkref`（自建参考）这几个子命令，日常最常用的还是 `count`。

输出目录结构

```
sample_01/
└── outs/
    ├── web_summary.html             # QC 网页报告（必看）
    ├── metrics_summary.csv          # web_summary 的数据版
    ├── filtered_feature_bc_matrix/ # 过滤后的表达矩阵（下游分析用）
    │   ├── barcodes.tsv.gz
    │   ├── features.tsv.gz
    │   └── matrix.mtx.gz
    ├── filtered_feature_bc_matrix.h5
    ├── raw_feature_bc_matrix/       # 未过滤矩阵，做 EmptyDrops 调参时用
    ├── raw_feature_bc_matrix.h5
    ├── possorted_genome_bam.bam     # 比对结果，可视化或 velocity 用
    ├── possorted_genome_bam.bam.bai
    ├── molecule_info.h5             # 每个 UMI 的分子级信息，aggr 要用
    └── cloupe.cloupe                # Loupe Browser 专用
```

`filtered_feature_bc_matrix/` 下是典型的 10x 三件套：

```
barcodes.tsv.gz          # 每行一个细胞 barcode
features.tsv.gz          # 每行一个基因：ENSEMBL_ID  symbol  Gene Expression
matrix.mtx.gz            # 稀疏矩阵（MatrixMarket 格式）
```

`filtered_feature_bc_matrix.h5` 是同样的信息压缩成单个 HDF5 文件，Seurat 和 Scanpy 都能直接读，对后续自动化脚本更方便。

web_summary 里看什么

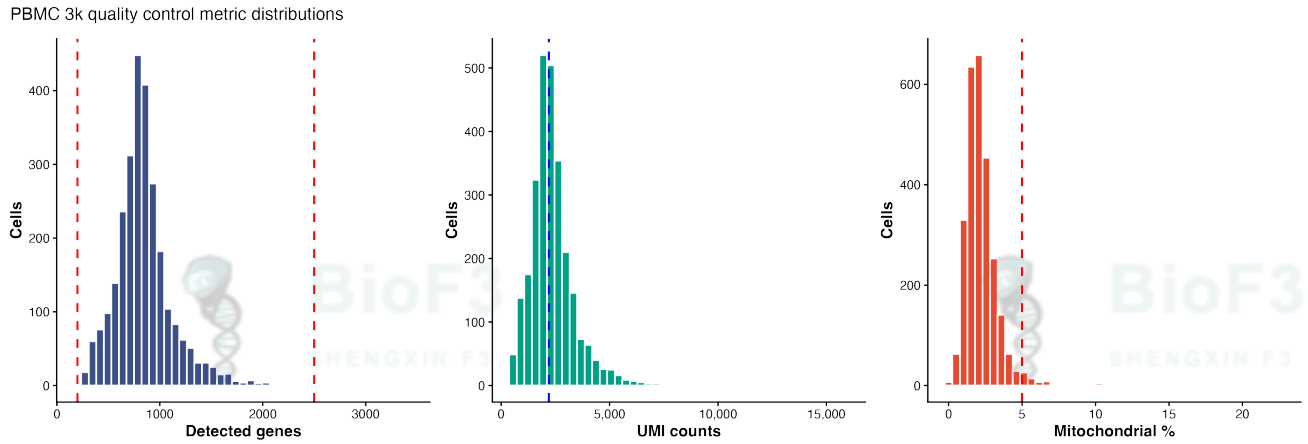

图 4：质量控制指标分布。展示了基因数、UMI 数和线粒体基因比例的分布情况，红色虚线表示质量控制阈值。

跑完之后第一件事是在浏览器里打开 `outs/web_summary.html`，看这几项：

指标	合理范围	过低提示	过高提示
细胞数（Estimated Number of Cells）	1,000 – 10,000（看上样）	细胞死亡、上样不足	双细胞、进样过浓
每细胞测序深度（Mean Reads per Cell）	20,000 – 100,000，最低 10,000	测序不足	—
中位基因数（Median Genes per Cell）	500 – 5,000	细胞质量差、深度不足	双细胞
总检测基因数（Total Genes Detected）	人/鼠 15,000 – 25,000	参考注释版本不匹配	—
测序饱和度（Sequencing Saturation）	50% – 80%	可以再加深度	继续测序收益递减
比对率（Reads Mapped to Genome）	> 80%	参考版本错、样本污染	—
线粒体基因比例	< 10%	—	细胞破损、应激

饱和度的公式是 `1 - unique UMIs / total reads`，意思是"再测下去还能看到多少新分子"。对 PBMC 这种相对稀疏的样本，60% 左右就可用；细胞图谱级项目允许更高，但回报递减。

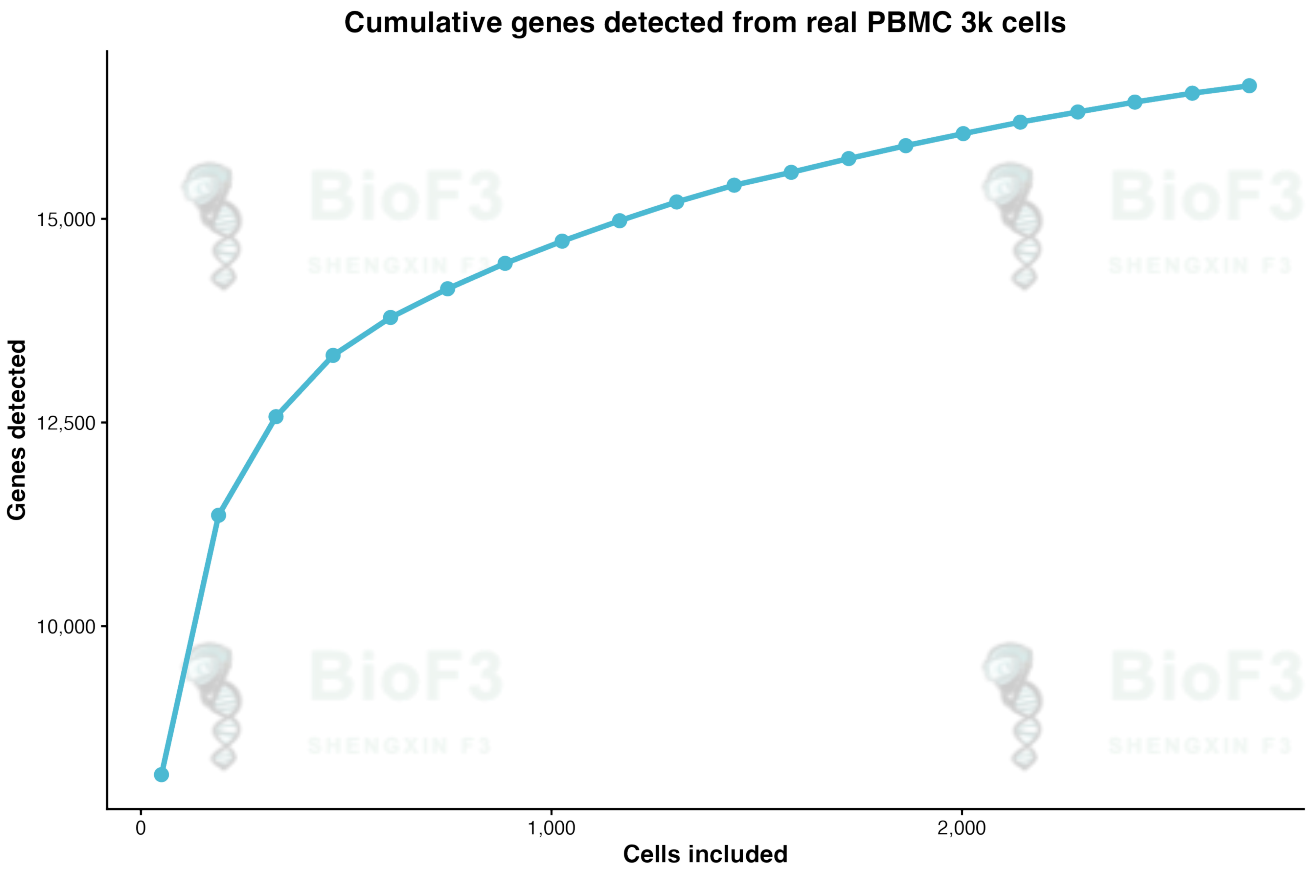

图 5：真实 PBMC 3k 细胞逐步纳入后累计检出的基因数。filtered matrix 不包含原始 reads，因此这里不伪造测序饱和度，而是展示可由矩阵直接计算的基因检出趋势。

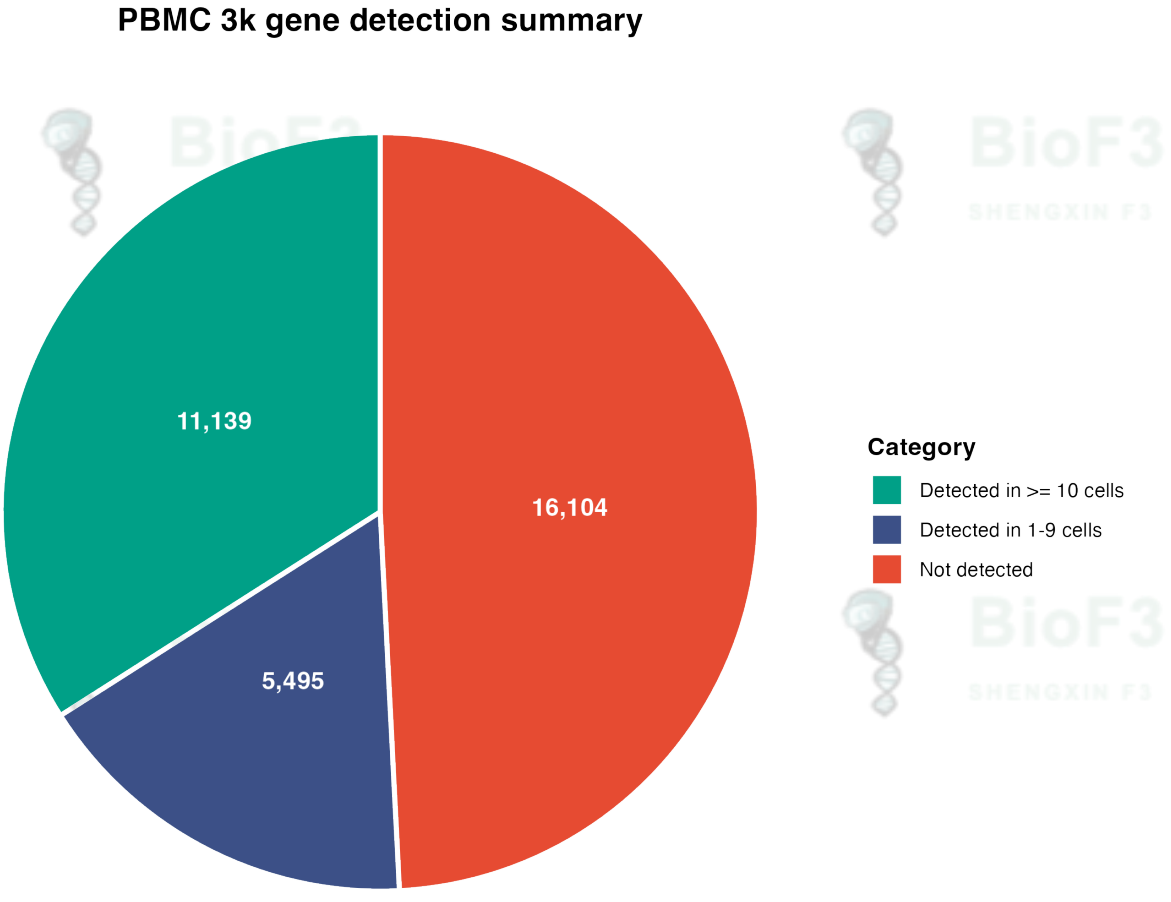

图 6：PBMC 3k 矩阵中的基因检出统计。filtered matrix 不包含比对日志，因此这里展示真实可追溯的基因检出类别。

把输出读进 Seurat / Scanpy

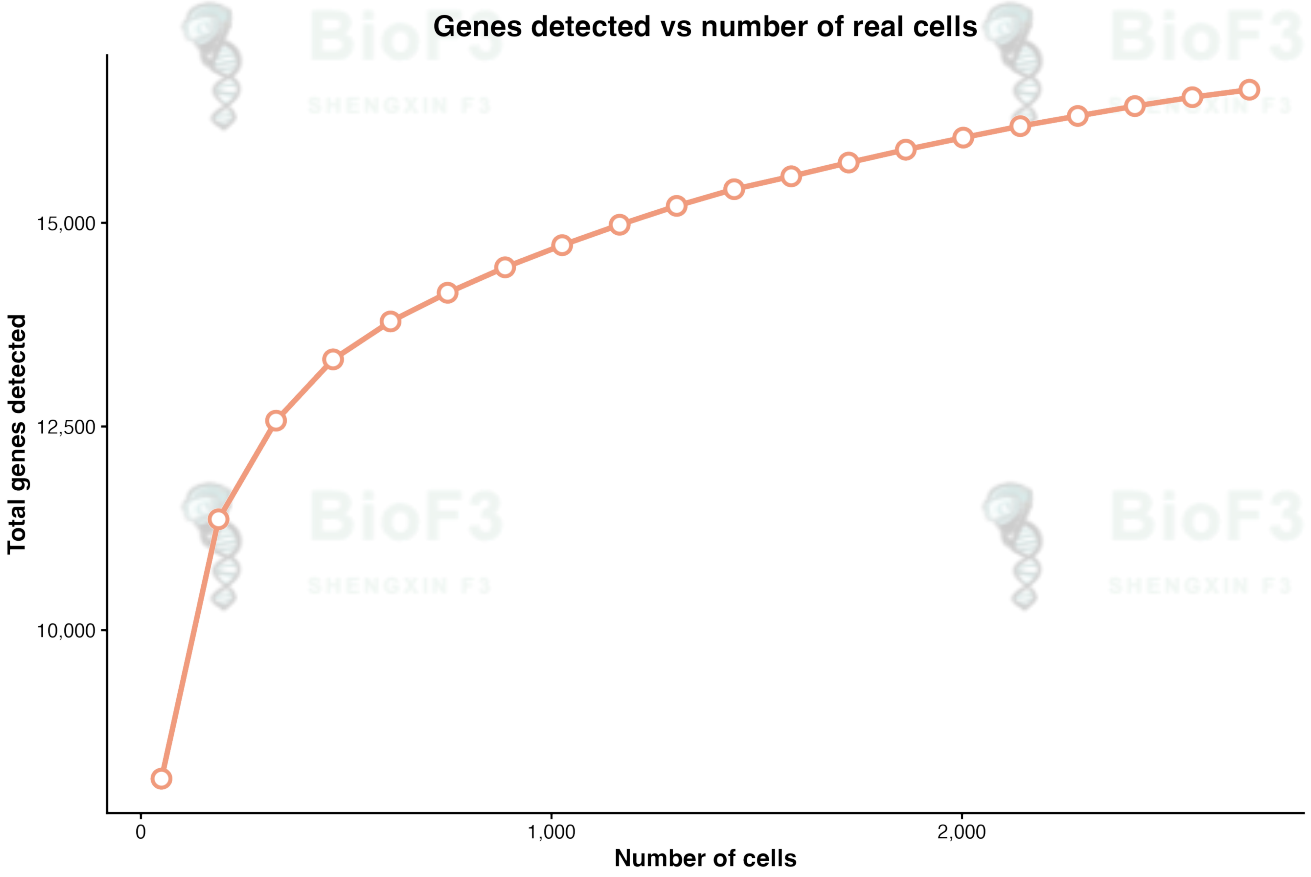

图 7：细胞数量与基因检测关系。展示了随着细胞数量增加，检测到的基因总数的变化趋势。

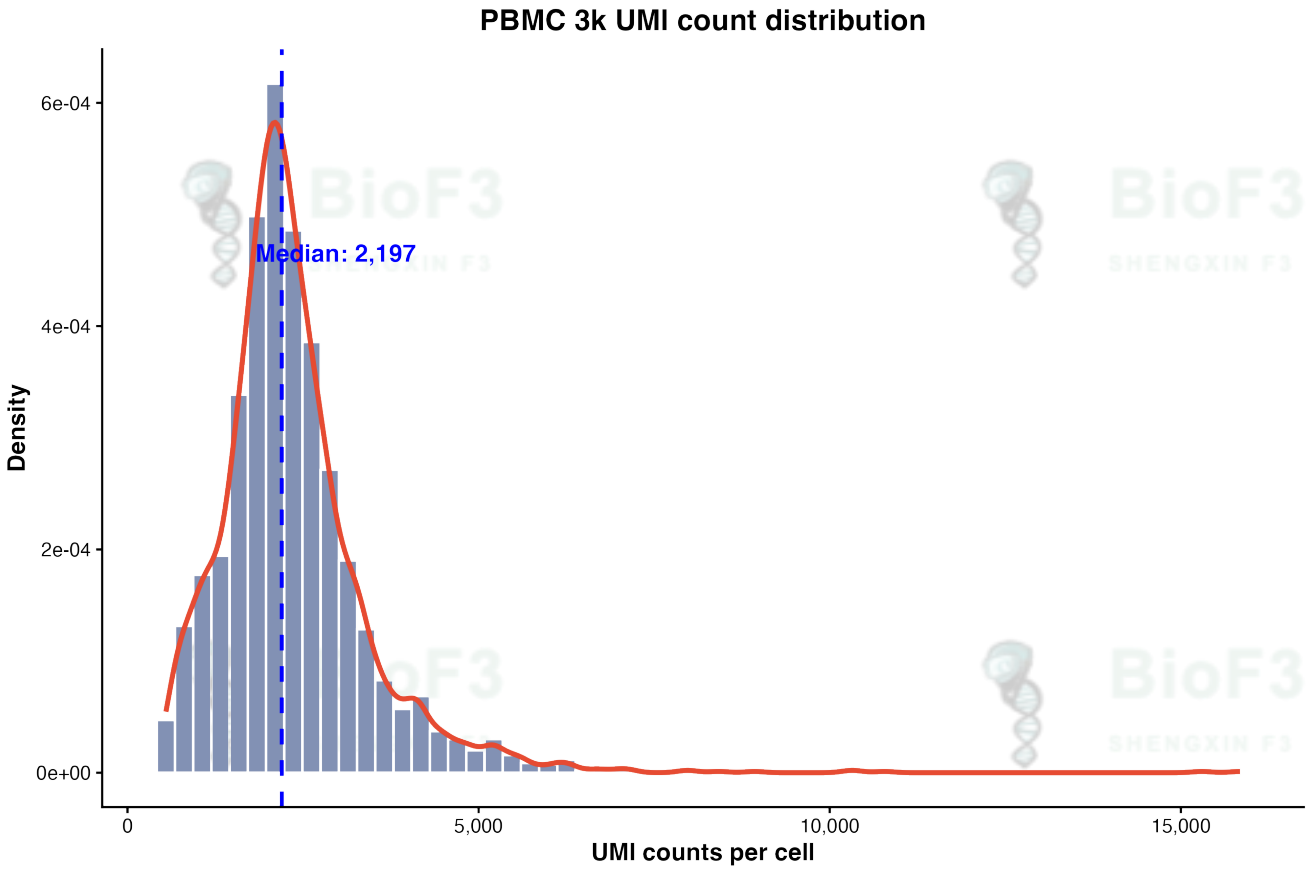

图 8：UMI 计数分布。展示了每个细胞的 UMI 计数分布，蓝色虚线表示中位数。

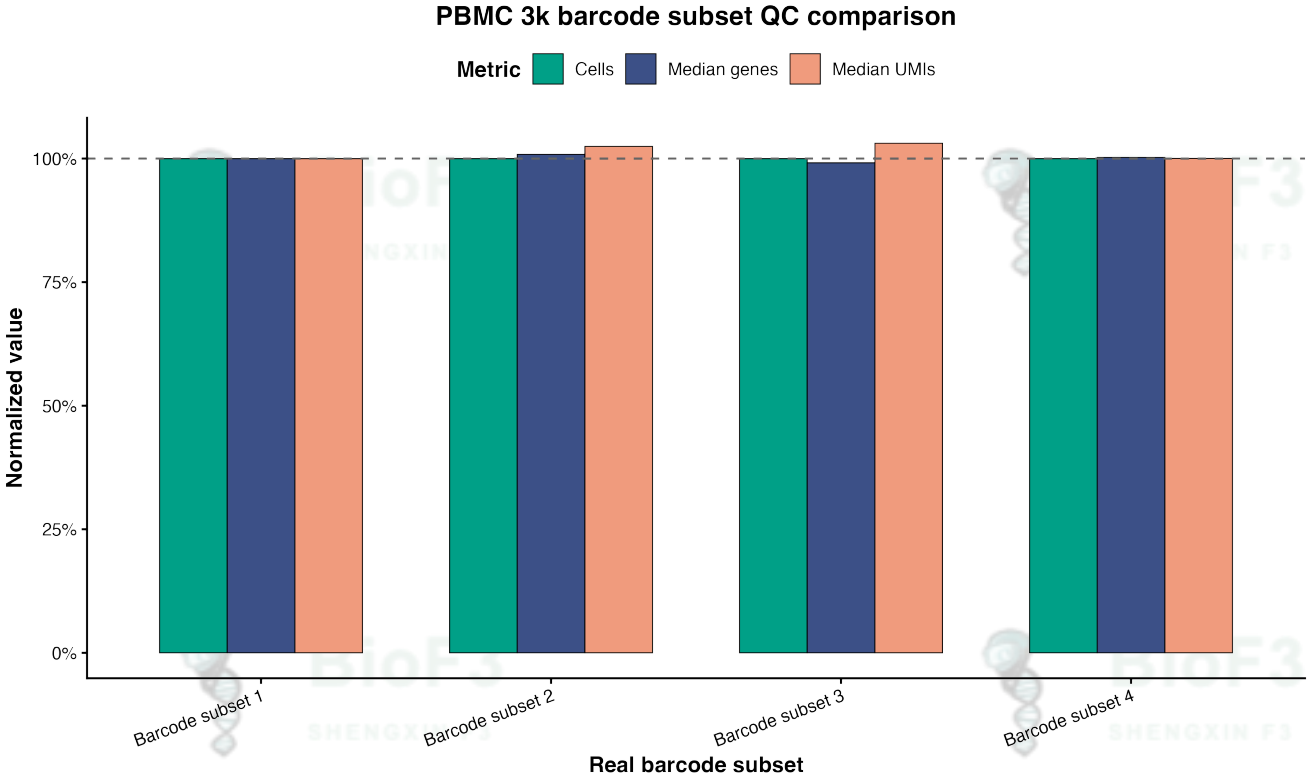

图 9：PBMC 3k 真实细胞按 barcode 顺序分成四个子集后的 QC 对比。它用于演示分组比较图形，不代表真实多样本批次。

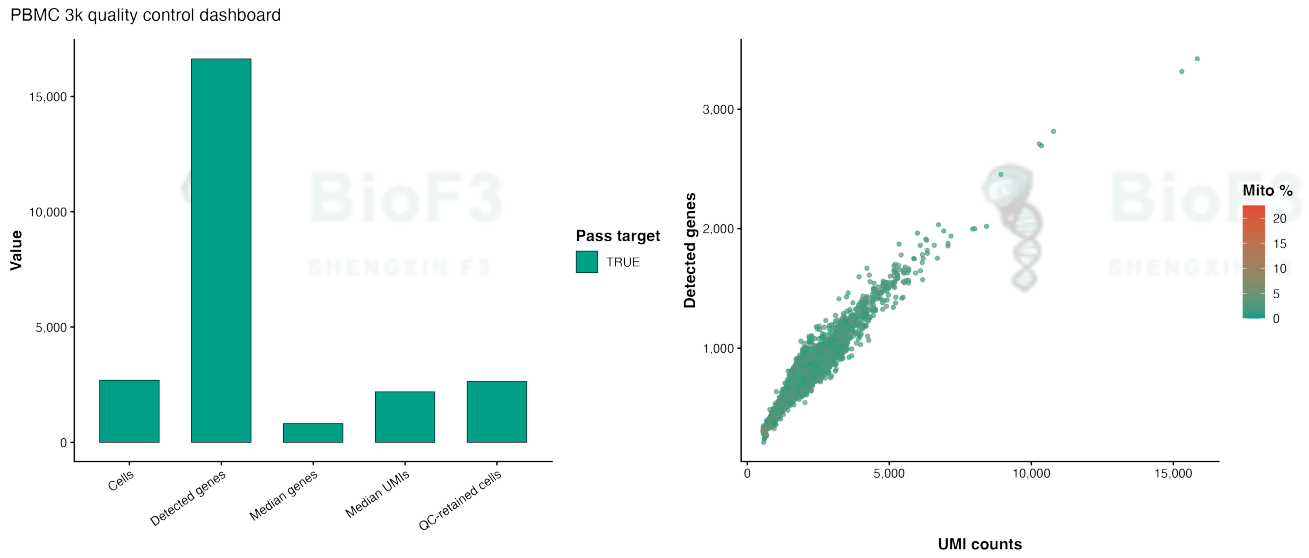

图 10：质量控制仪表盘。综合展示了关键质量指标和细胞质量分布的散点图。

R 端用 Seurat：

```
library(Seurat)

data_dir <- "sample_01/outs/filtered_feature_bc_matrix/"
data <- Read10X(data.dir = data_dir)

seurat_obj <- CreateSeuratObject(
  counts       = data,
  project      = "PBMC",
  min.cells    = 3,    # 只保留至少在 3 个细胞里表达的基因
  min.features = 200   # 只保留至少检测到 200 个基因的细胞
)
seurat_obj
```

Python 端用 Scanpy：

```
import scanpy as sc

adata = sc.read_10x_mtx(
    'sample_01/outs/filtered_feature_bc_matrix/',
    var_names='gene_symbols',
    cache=True,
)
print(adata)
```

用 H5 文件更简单：

```
# Seurat
library(Seurat)
data <- Read10X_h5("sample_01/outs/filtered_feature_bc_matrix.h5")
seurat_obj <- CreateSeuratObject(counts = data)
```

```python
# Scanpy
import scanpy as sc
adata = sc.read_10x_h5('sample_01/outs/filtered_feature_bc_matrix.h5')
```

读进来之后就可以进入下一章的 QC、标准化、聚类流程。

下载资源

module03_complete_sci.R
16 KB

下载图表生成脚本 ↗

下一步

继续学习：03 质量控制、聚类与细胞类型注释

参考资源

- Cell Ranger 官方文档
- 10x Genomics 支持中心
- 单细胞分析最佳实践
- Seurat
- Scanpy

扫码关注微信公众号【生信F3】

获取文章完整内容，分享生物信息学最新知识。