Vireo多批次拆样教程：输入检查、任务隔离与资源测量

完整操作教程 · 参考固定 · 往返恢复验收

逐批核对AD/DP矩阵、变异顺序与细胞清单，固定Vireo设置，保存独立供者分配与概率输出。

运行边界：示例代码需按安装版本和真实项目输入复核；未提供真实生物数据，语法或小样本检查不代表科学结果和性能验收。

Vireo批量拆样应以真实混样文库为单位，不能把不同批次的AD/DP矩阵按列直接拼接就当一个模型。位点、细胞及供者设计都要对应。本文使用当前vireoSNP文档的无已知基因型路线，先检查cellSNP目录，再独立运行每批。

先区分上游计数与Vireo模型

Vireo输入是变异×细胞的替代等位基因计数AD和总深度DP，不是基因表达矩阵。上游cellSNP-lite的参考、位点选择和过滤另作记录。每批固定供者数与随机种子，跨批的donor0不能直接解释为同一真实个体。

材料	一致性要求	不能替代
AD与DP	形状、位点行及细胞列相同，AD不大于DP	不是ALT与REF分别当AD/DP
样本清单	细胞数、顺序和唯一barcode	行数相同不证明顺序正确
位点VCF	与矩阵逐行对应、参考一致	同名染色体不等于同组装
供者设计	预期人数、已知基因型可用性	无基因型模式不自动给出真实姓名

读取计数前先检查基本结构

保存为check_cellsnp.py，传入一批cellSNP目录。需要已经安装SciPy和NumPy。该版本目录包含下列四个标准文件；检查会读入稀疏矩阵，也需要RAM，不是零成本或完整VCF验证。

```
import gzip, sys
from pathlib import Path
import numpy as np
from scipy.io import mmread

root = Path(sys.argv[1])
ad = mmread(root / "cellSNP.tag.AD.mtx").tocsr()
dp = mmread(root / "cellSNP.tag.DP.mtx").tocsr()
assert ad.shape == dp.shape and min(ad.shape) > 0
for matrix in (ad, dp):
    assert np.isfinite(matrix.data).all() and (matrix.data >= 0).all()
    assert (matrix.data == np.floor(matrix.data)).all()
assert ((dp-ad).data >= 0).all()
cells = (root / "cellSNP.samples.tsv").read_text().splitlines()
assert len(cells) == ad.shape[1] and all(cells) and len(set(cells)) == len(cells)
with gzip.open(root / "cellSNP.base.vcf.gz", "rt") as handle:
    n_variants = sum(not line.startswith("#") for line in handle if line.strip())
assert n_variants == ad.shape[0]
print("variants", ad.shape[0], "cells", ad.shape[1], "count_structure_ok")
```

通过只说明这些结构与计数关系，不验证矩阵行的具体位点真的按同一顺序。上游生成清单与校验仍须保留；基因型不足或AD/DP混用时先修来源，不删掉检查来让程序继续。

一批一目录，先用单子进程基线

CELL_DATA是该批目录，DONORS是实验预期正整数，OUT必须不存在；先运行上述检查。下例使用已核对CLI的-p 1及randSeed，按实际版本保存帮助。增加-p是增加子进程，并可能以更多内存换速度，不只按主机核心数拉满。

```
set -eu
: "${CELL_DATA:?[redacted]cellSNP[redacted]}"
: "${DONORS:?[redacted]}"
```

```
: "${OUT:?[redacted]}"
case "$DONORS" in ""|*[!0-9]*) exit 2;; esac
test "$DONORS" -ge 1 || exit 2
test ! -e "$OUT" && test ! -L "$OUT" || exit 2
python check_cellsnp.py "$CELL_DATA"
vireo -h > "$OUT.help.txt"
/usr/bin/time -v -o "$OUT.resources.txt" \
  vireo -c "$CELL_DATA" -N "$DONORS" -p 1 --randSeed 101 -o "$OUT" \
  > "$OUT.stdout.log" 2> "$OUT.stderr.log"
test -s "$OUT/donor_ids.tsv" && test -s "$OUT/summary.tsv" || exit 1
```

检查概率、覆盖和批次边界

逐细胞核对donor_ids.tsv与输入清单，检查供者分配、未分配、双细胞、概率及有效位点数。不要为减少未分配而任意降低阈值，也不把分配率高当成拆样准确率。各批保持独立模型，身份匹配需额外基因型或实验信息。

若使用已知供者VCF，需另核对参考、位点交集和实际存在的GT、GP或PL字段，不能沿用无基因型示例后只加一个文件名。查询计数、基因型加载和模型各自记录资源，避免把上游计数耗时算成Vireo自身速度。

先单批稳定，再测允许的小并发，关注多个模型与计数矩阵重叠的RAM。失败保留日志并用新目录重试，必要批次未完成就不能交付全队列状态。保存参数、矩阵和结果对应关系后，再按真实资源缺口决定批量安排。

批次清单记录输入目录、位点数、细胞数、供者数、种子和结果位置。合并队列结果前加入文库标识，避免相同barcode或donor标签发生碰撞。若某批有效遗传信息明显不足，应记录为待复核，不能从其他批借用分配结果填空。已知基因型匹配与无基因型聚类的输出也要分别解释。

一批任务的manifest应该包含什么

除矩阵文件名外，为每批建立manifest，记录library_id、批次、输入目录、AD/DP和VCF校验值、细胞清单散列、参考基因组与位点来源、供者数N、随机种子、vireoSNP版本、命令、作业额度和输出目录。cellSNP矩阵的行序必须和VCF位点顺序一致，列序必须和细胞清单一致；单独比较维度无法证明顺序正确。若上游换了位点过滤或参考版本，应创建新的输入版本，不能复用旧批次manifest。

多个批次各自运行时，固定输出路径格式并保留原始barcode。合并细胞结果时使用library_id与barcode组成稳定键，而不是裸barcode。对相同供者数、不同供者数、不同测序深度批次分组比较时，应明确哪些差异来自设计而非算法。跨批次donor标签是批内推断编号，不能因为编号相同就合并到同一个人。

结果文件怎样验收

检查summary和donor assignment文件中的细胞数、列名和空值，并将输入细胞清单与输出assignment做集合比较，列出漏失和重复项。统计每个donor簇的细胞数、未分配数量和doublet相关输出；异常比例应回到测序深度、遗传位点覆盖和预期混样设计判断，而不是单独设一个通用阈值判失败。保存每批结果与参数，不要直接覆盖上一轮结果。

若运行成功但结果极不均衡，先检查AD/DP计数方向、位点来源及参考组装；再查看是否使用了匹配批次的VCF和细胞清单。结构守卫通过仅证明输入格式基本合理，不证明位点对齐正确。可抽取若干位点与细胞核对生成记录，并在已知基因型可用时按审批流程做独立比较。若无真值，报告应该描述为推断分群而非身份鉴定。

根据错误阶段判断资源需求

Matrix Market文件读取阶段内存不足，先确认矩阵是否被稠密化、Python进程是否重复载入矩阵，以及当前作业内存限制；检查器本身也会将AD和DP同时读入，所以预检也需考虑内存。若Vireo模型阶段失败，保存首个错误和资源时间序列，区分依赖、参数、输入问题与系统OOM。增加子进程可能加速特定阶段，也可能增加内存压力，不应直接把-p调到主机核心数。

资源试跑需使用代表性文库，包含相近细胞数、位点数和供者规模。记录CPU、峰值RSS、临时磁盘、运行时长和退出码，然后在计划的并发数下再次验证。不要把一次小样本运行的峰值简单乘以细胞比例，因为位点数、矩阵稀疏度和算法步骤可能同时影响内存。若问题是错位或错误参考，加RAM不会改善分群质量。

如何报告拆样结论

报告应分开写输入质量、算法输出和身份解释。输入质量包括细胞数、位点数、计数来源及参考；算法输出描述簇和分配状态；身份解释说明是否通过独立基因型或实验设计信息完成映射。没有已知基因型时，保留donor0等方法编号并标注推断，不要把它们替换成样本姓名。

对于失败文库，写明卡在哪一步、重跑次数和当前状态，并保留日志。批次汇总表同时列出通过、待复核、失败和未启动的文库，避免只报告成功项造成覆盖率偏高。交付前由另一位分析者抽查输入散列、输出细胞集合及簇解释；未完成复核的结果应保持待复核标签。

跨批次合并前做三项核对

首先核对每批细胞ID命名空间是否唯一，建议构造library_id:barcode键；其次核对每批位点清单的组装、VCF过滤和位点顺序；最后核对donor编号是否仅在批内有效。任一项不一致时都应停止合并，先修正映射表。输出合并后的细胞数应等于各批通过审核细胞数的总和，差异必须能追溯到明确过滤记录。

如果同一个供者参与多批实验，只有独立基因型匹配或项目认可的样本信息能够支持跨批映射。遗传簇外观相似、簇编号相同或文库时间接近都不是充分证据。需要跨批对齐时，建立单独的身份推断表，记录参考来源、比对规则、置信信息和复核人，主拆样结果保留原始簇编号。

使用资源咨询清单

向计算管理员或服务器服务方提供实际的AD/DP维度、矩阵非零数、VCF位点数、细胞数、供者数、软件版本、当前-p并行度和失败日志。说明失败发生在矩阵读取、模型推断还是文件写出，并附实际作业内存额度。这样才能判断是磁盘读取、稀疏矩阵展开、线程设置还是单批规模限制，而不是仅根据总细胞数估一个RAM数字。

如果样本属于受控数据，先确认存储位置、访问账号、备份和临时文件清理策略。处理后的donor assignment可能仍具有敏感性，不要把原始基因型或带身份映射的结果上传到未授权空间。服务器可提供计算和隔离环境，但数据使用权、共享范围及保存期限仍按项目和机构规定执行。

简易项目复核记录

复核表可按“文件结构、位点一致性、细胞覆盖、模型输出、身份解释”五栏填写。每栏写具体证据路径，而不是只勾选通过。例如位点一致性附VCF和上游清单散列，细胞覆盖附漏失/重复ID列表，身份解释注明有无外部基因型。审阅者发现问题时写明影响的是计算可运行性还是科学解释。

若团队没有专职计算管理员，记录当前conda环境或容器摘要、vireoSNP安装来源和可用磁盘位置。下次换节点时先运行帮助命令和小数据结构检查，确认依赖完整后再提交大批次。失败日志里可能包含文件路径或样本标识，对外分享前注意脱敏。

复用结果前的判断

同一输入和配置已通过检查的批次可以作为缓存结果复用，但应核对manifest散列、软件版本和输出完整性。任何输入文件、位点筛选、供者数、随机种子或软件升级变化都应创建新运行ID。文件夹存在或summary非空不足以证明它来自当前任务；要将关键输出散列写回任务台账，形成可比对的证据。

长期项目应保留失败和待复核状态的原因，避免每次重新启动都把旧问题当新问题排查。定期检查归档路径是否仍可访问、校验值是否匹配，以及元数据是否缺少字段。对敏感数据按机构要求进行备份与清理，不将临时文件生命周期交给脚本默认行为。

批次状态表还应记录负责人、提交时间和最近一次检查时间。任务长期停在running时，先确认主进程及子进程是否仍在运行，再更新状态；不能因为日志不再增长就直接删除目录。对未完成批次保留未决原因，下一位接手者可以从manifest和日志判断是否需要重跑。

完成数据复核后，再决定是否共享跨批汇总。拆样结果与身份映射表应分开管理，普通分析者可以查看去身份化簇标签，而身份对应信息按最小权限保存。

配套练习包

本PDF对应的ZIP包含本文代码块、主题练习模板或合成数据、README和SHA256校验文件。请从同一教程页面下载练习包；先核对校验值，再在独立目录复现。

参考资料

Vireo官方手册与输入格式 · 来源资料

Vireo当前CLI、nproc和随机种子 · 来源资料

Vireo当前cellSNP读取及结果输出 · 来源资料

操作建议：先在合成小样本上验证流程，再依据项目参考、标签合同、存储策略与获配资源验收真实数据。