computeMatrix大任务教程：区域规模、分组拆分与内存监控

完整操作教程 · 参考固定 · 往返恢复验收

估算computeMatrix矩阵维度，固定缺失与排序规则，正确验收区域分片及rbind合并。

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。

computeMatrix把多个bigWig按区域汇总成矩阵，峰值会随区域数、轨迹数和bin数量增长。压缩输出只有几十MB，不表示运行时只需相同内存。先明确分析窗口和区域集合，再决定是否分组试跑，不能把删区域或降低分辨率当作结果完全不变的优化。

先算矩阵主体的数量级

以reference-point中心上下各3kb、bin50为演算，每条轨迹每区域约120个位置。若10万个区域、8条轨迹，双精度值主体约0.768十进制GB；这只是数值部分，元数据、并行副本、掩码和中间计算还会增加峰值，不能直接当作服务器RAM要求。

检查项	固定内容	验收记录
区域	参考组装、BED6坐标与唯一ID	总行数、范围、链和重复
轨迹	同一顺序的bigWig清单	染色体及长度、信号定义
窗口	reference-point中心、上下游与bin	预期位置数和矩阵维度
拆分	每区域只进入一个分片	行数守恒、组标签与最终排序

先固定一份完整协议

以下要求regions.bed为已验收BED6，无表头、track或分组注释行，ID唯一且染色体与bigWig一致。A、B均为已正确生成的信号轨迹。代码保留原始区域，不使用skipZeros或missingDataAsZero；缺失和零信号的含义需要明确，不能为减少内存混用。

```
#!/usr/bin/env bash
set -euo pipefail
test -s regions.bed && test -s A.bw && test -s B.bw || exit 2
mkdir matrix-full-001
computeMatrix --version > matrix-full-001/version.txt 2>&1
/usr/bin/time -v -o matrix-full-001/resources.txt \
  computeMatrix reference-point --referencePoint center \
  --beforeRegionStartLength 3000 --afterRegionStartLength 3000 \
  --binSize 50 --numberOfProcessors 2 --sortRegions keep \
  --regionsFileName regions.bed --scoreFileName A.bw B.bw \
  --outFileName matrix-full-001/matrix.gz \
  --outFileSortedRegions matrix-full-001/regions.bed \
  > matrix-full-001/run.log 2>&1
test -s matrix-full-001/matrix.gz
gzip -t matrix-full-001/matrix.gz
computeMatrixOperations info -m matrix-full-001/matrix.gz
```

输出区域文件用于追踪实际保留与排序，不应只拿输入行号猜结果对应。先抽查正负链、染色体边界和有/无信号区域，确认缺失处理符合研究目标。运行成功也可能出现大量缺失或整条轨迹参考不匹配，不能只检查压缩文件可解开。

需要分片时保持样本与方法一致

可按区域分片控制一次任务大小。示例每片2万行，分片前必须完成上述BED6验收并保存输入校验；样本bigWig顺序、窗口、bin、排序及过滤参数全部相同。逐片执行与全量同类命令，仅替换区域和独立输出路径，每片日志、资源与实际输出区域单独保存。

```
set -euo pipefail
mkdir region-parts-001
split -l 20000 -d -a 4 regions.bed region-parts-001/part-
# ■■■■■■■■■■■■■■■■■■■■■■■■
computeMatrixOperations rbind \
  -m part-0000.matrix.gz part-0001.matrix.gz \
  -o joined.matrix.gz
computeMatrixOperations info -m joined.matrix.gz
```

上面只展示两片的合并形状，必须把实际全部分片列入清单；只有一片时无需合并。不要把split文件当成已经计算的矩阵，也不要用普通cat拼gzip代替工具操作。rbind要求同样本且顺序一致；合并本身仍需内存，分片并不保证最终热图和聚类能在原机器完成。

验收合并边界与后续负担

核对每片输出区域数、ID和组标签，确保没有遗漏、重复或不兼容参数。各片局部排序不是全局排序；需要全局排序或聚类时应统一处理，并重新测量那一步资源。对比一份能完整计算的小集合与其分片合并结果，检查数值与缺失分布，确认操作符合当前版本行为。

内存失败先减少同时运行的实例与处理进程，观察单任务峰值；区域或bin必须改变时另存方法协议。轨迹数、区域长度和过滤规则变化后重新估算，不能按旧压缩文件大小线性规划。保留原始bigWig、区域清单与矩阵，画图是否截断色标也应单独说明，避免把显示变化误作生物学差异。

区域文件和轨迹清单先做完整校验

BED6区域文件应确认染色体命名、0-based坐标、start小于end、参考长度、唯一ID及strand列合法。computeMatrix可以处理合法BED输入，但对错误坐标有时只产生警告或丢弃区域；正式任务前应独立统计无效和越界行。若区域来自不同来源，先统一基因组版本并记录转换工具与链方向处理。

bigWig清单应固定样本顺序，并比较chromosome名称和长度。某个文件缺染色体或使用不同组装时，不能靠跳过缺失自动拼成可比矩阵。抽取同一已知区间确认信号存在和尺度合理。所有bigWig、区域集和参数都写入manifest并计算散列，确保矩阵行列对应清楚。

矩阵规模估算与内存边界

主体元素数量可以按区域数×轨迹数×bin数粗估，但还要考虑矩阵的数据类型、排序副本、区域元数据和多进程工作区。区域窗口越大或bin越小，每区域位置数越多。压缩后的matrix.gz体积与解压后的矩阵峰值并不相同；根据代表性任务实测内存，而不是使用压缩文件大小申请RAM。

numberOfProcessors会影响同时处理的轨迹和临时对象数量，线程越多并不总是越省时。先用一份完整且具有代表性的区域集测量1–2个线程的运行，记录峰值RSS和墙钟。随后试验目标并发数，确认作业内存限制和共享存储能力。失败发生于读取bigWig、矩阵生成还是写盘，应分别判断。

分片前后确保行一一对应

按区域拆分时，将唯一region_id作为关键字段保留，记录每片起止行和输入散列。检查所有分片的区域集合没有重叠或遗漏，所有样本bigWig顺序、窗口、bin和missing/zero策略相同。每片输出排序后不能依靠原始行号拼接；应按region_id重新排序并核对总行数。

拆分运行可能减少单次内存，但增加重复加载bigWig和调度开销。选择分片大小时先做试跑，观察峰值、运行时间和临时盘。过小分片会让总耗时更长；过大片段仍可能超出额度。遇到失败只重跑该片并保存attempt编号，不覆盖其他已验收片。

缺失值与零值不能混用

bigWig中未定义的区间可能显示为缺失，真实测得的零信号则是有效数值。missingDataAsZero、skipZeros等选项会改变矩阵内容和区域保留范围。选择前先明确缺失代表什么：轨迹未覆盖、染色体不匹配还是生物学上无信号。不要为得到无NA矩阵而把缺失自动填零。

抽查正负链、边界区域、已知高信号区及无信号区，比较输入区域和输出排序区域。检查每条轨迹缺失比例及区域是否被过滤；大量缺失可能是组装/染色体命名问题。即使gzip -t通过，也只证明压缩流可读，不证明样本顺序、矩阵列数和区域含义正确。

矩阵导出与下游分析

computeMatrix输出包含轨迹值和区域元数据，导出成普通表时需要明确轨迹顺序、bin位置、缺失策略和区域ID。下游聚类或热图的样本分组应与实验设计一致，避免把同一实验技术重复当作独立生物重复。若进行组间统计，矩阵可用于特征整理，但应使用适合的统计模型和重复结构。

报告中说明分析窗口、reference point或scale-region方法、binSize、排序策略、过滤和线程数。将全量结果与探索性分片结果区分保存，附运行manifest、日志、输入散列和矩阵维度。图形应标注轨迹和组别，避免只交热图而没有可追溯的原始矩阵。

分片矩阵的列对齐与合并

同一片区域在各轨迹中的bin列顺序应固定，bigWig列表顺序也必须一致。生成矩阵后检查metadata中的sample_labels、group_labels和区域ID，防止不同分片使用不同轨迹排序。合并分片时按region_id和预先定义的窗口位置恢复顺序，不要直接拼接gzip文件；computeMatrix格式需要通过支持工具合并或重新生成。

分片处理后统计输入区域数、每片成功数、输出区域数、过滤数和重复ID。总行数不等时列出差异ID并解释。排序策略设置为keep时仍需核对输出区域文件与输入，其他排序选项则应保存排序结果。行顺序错位会使热图看起来正常但区域注释错误。

边界区域和异常轨迹

距离染色体端点过近的区域可能无法覆盖完整上下游窗口，deepTools可能调整或丢弃部分区域。提前根据窗口宽度筛查边界，明确是否保留并报告规则。负链区域应按所选模式正确反向；抽查几个正负链已知基因，确认上游/下游方向没有颠倒。

某条轨迹在某些染色体缺失时，矩阵可能包含大量缺失或跳过区域。分别统计每条bigWig的覆盖和missing比例，确保缺失不是被默认转成零。零信号可以是有效观测；缺失可能是无覆盖或参考不一致，两者在下游统计上不可随意互换。

何时分片，何时重新设计矩阵

如果完整区域集合在相同参数下超过作业内存，先试区域分片并测量单片峰值和全批总耗时。若主要问题是分辨率太高或窗口过宽，可与研究者讨论调整分析设计，但需将调整作为新的方法版本并说明丢失的空间细节。不能为了满足硬件限制而把改过分辨率的结果伪装成原协议。

采样区域的试跑不能用于估算所有轨迹的最坏情况，特别是大文件或高信号样本。用代表性及较大样本测上界，观察磁盘I/O与内存变化。若计算时间主要在读取bigWig，增加RAM可能帮助有限；如果矩阵展开占主导，再依据进程峰值评估更大内存或不同处理策略。

区域级验收

对输入regions.bed与输出sorted regions按唯一ID比较，记录丢失、重复和排序变化。抽查矩阵数值是否与bigWig查询结果对应，并按样本顺序核对轨迹列段。区域数正确不能证明样本列未颠倒。

查看matrix.gz元数据中的窗口、bin和轨迹标签。下游脚本若移除行名或忽略缺失掩码，可能导致列错配。将读取脚本、分组规则和热图参数一起归档。

资源申请材料

申请算力时提供区域数、轨迹数、窗口位置数、binSize、线程、实测峰值RSS、临时盘和失败阶段。说明是否可区域分片以及如何保证行顺序与缺失值处理。由此判断增加内存、降低并发或分片哪种更合适。

零信号与缺失不能混用。报告每条bigWig的缺失比例及参考兼容性；如果大量缺失源于染色体命名或组装不一致，应修复输入，而不是用更多RAM继续计算。

若输出区域数减少，先检查边界截断、bigWig覆盖和deepTools过滤选项，再判断是否符合预期。记录每一步被跳过或缺失的region ID，不能仅凭最终矩阵可读就忽略覆盖差异。

热图的聚类、排序和颜色缩放也会影响视觉解释。保存绘图输入矩阵、样本分组、颜色范围和区域注释；若按信号排序，说明排序依据并保留未排序结果用于追踪。

若上下游长度或binSize改变，矩阵的列数及每列代表的碱基范围都会变化。下游图表或统计脚本应读取对应元数据，不可沿用旧版本列名解释新矩阵。

任务记录还应标注matrix.gz对应的区域清单散列、轨迹清单散列和命令版本。下游分析前先核对散列，确保热图引用的是同一次生成的区域和bigWig组合。

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

computeMatrix区域、排序与缺失参数 · 来源资料

computeMatrixOperations合并约束 · 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。