CNVnator多样本教程：独立ROOT、分箱与结果检查

以人类WGS常染色体为例，逐样本建立CNVnator ROOT文件，按tree、histogram、stat、partition、call顺序运行，并记录资源和空结果。

CNVnator根据比对读段的覆盖深度寻找拷贝数变异候选。批量分析最容易出错的地方是把不同样本写进同一个ROOT文件，或让同一个样本的多个步骤同时修改ROOT。下面以已完成比对质控的人类短读长WGS为例，每个样本独立建文件、顺序处理chr1至chr22。外显子捕获、肿瘤或非二倍体数据需要单独评估覆盖基线，不能直接照搬此例。

运行前核对	具体内容	留档
样本	BAM、样本ID、覆盖及比对参考	样本表与BAM校验值
参考	FASTA与BAM的组装、chr前缀一致	参考版本、目标染色体清单
参数	CNVnator/ROOT版本和bin size	命令、日志、资源峰值

准备样本清单和参考文件

samples.tsv无表头，每行两列TAB分隔：唯一样本ID和BAM绝对路径。示例参考目录需含chr1.fa至chr22.fa，且与BAM头中的染色体命名对应；文件名看起来一致并不能证明参考序列版本相同。运行前核对BAM索引和基本质控，记录调用版本、参考来源及样本覆盖。bin示例设为1000 bp，只是演示值；分箱越小不一定越好，分辨率、噪声和耗时都可能变化，项目内应统一并说明理由。

每个样本顺序运行完整流程

将程序路径、参考目录和染色体范围改成项目实际值。先在空工作目录运行一个代表性样本；确认ROOT内容与日志后，再逐步增加并发。CNVnator官方流程要求先生成读深度、直方图和统计，再partition与call；同一样本这几步必须使用同一ROOT和bin。

```
set -euo pipefail
cnv=/opt/cnvnator/cnvnator
ref=/ref/chromosome-fasta
bin=1000
chroms=(chr{1..22})
test -x "$cnv"
for chr in "${chroms[@]}"; do test -s "$ref/$chr.fa"; done
mkdir cnvnator-batch
sha256sum "$cnv" > cnvnator-batch/program.sha256
root-config --version > cnvnator-batch/root-version.txt
declare -A seen=(); n=0
while IFS=$'\t' read -r id bam extra || [[ -n "${id:-}${bam:-}${extra:-}" ]]; do
  [[ "$id" =~ ^[A-Za-z0-9][A-Za-z0-9_.-]*$ ]]
  [[ "$bam" = /* && -z "${extra:-}" ]]
  [[ -z "${seen[$id]+yes}" ]]; seen[$id]=1
  test -s "$bam"; samtools quickcheck -v "$bam"
  d="cnvnator-batch/$id"; mkdir "$d"; root="$d/$id.root"
  /usr/bin/time -v -o "$d/tree.time.txt" "$cnv" -root "$root" -tree "$bam" -chrom
"${chroms[@]}" > "$d/tree.log" 2>&1
  test -s "$root"
  /usr/bin/time -v -o "$d/his.time.txt" "$cnv" -root "$root" -his "$bin" -d "$ref" > "$d/
his.log" 2>&1
  /usr/bin/time -v -o "$d/stat.time.txt" "$cnv" -root "$root" -stat "$bin" > "$d/stat.log"
2>&1
  /usr/bin/time -v -o "$d/partition.time.txt" "$cnv" -root "$root" -partition "$bin" > "$d/
partition.log" 2>&1
  /usr/bin/time -v -o "$d/call.time.txt" "$cnv" -root "$root" -call "$bin" > "$d/calls.txt"
2> "$d/call.log"
  "$cnv" -root "$root" -ls > "$d/root-contents.txt" 2>&1
  du -b "$root" "$d/calls.txt" > "$d/retained-bytes.txt"
  printf 'COMPUTE_OK\n' > "$d/status.txt"
  n=$((n+1))
done < samples.tsv
test "$n" -gt 0
```

怎样验收候选和资源

逐样本检查tree、his、stat、partition和call日志，确认ROOT内部对象与本次样本、bin及染色体一致；ROOT文件存在不代表各阶段都成功。call结果写入标准输出，空文件可能表示没有候选，也可能是输入范围或覆盖异常，应检查日志、深度质控和ROOT内容，不能直接报成“全基因组无CNV”。

输出中的区间、类型、长度、归一化深度和统计值是候选证据。GC偏倚、低可比对性、重复区域和低覆盖都会影响信号；q0或单个e-value不应单独作为保留/剔除标准。对重要候选结合深度曲线、样本背景和独立证据复核；肿瘤纯度、性别和非整倍体也会影响基线解释。

记录每阶段耗时、峰值内存、ROOT和工作盘占用。结束后的文件大小不等于运行期间临时盘峰值；可用系统监控周期采样。多样本只有在单样本流程通过后再并行，每个样本各用ROOT和输出目录，避免同时写同一文件。归档清单应包括BAM来源与校验值、参考版本、bin、程序与ROOT版本、日志、calls和失败原因；重新计算用新目录保留旧结果。跨样本比较前先检查测序深度、GC偏倚和批次质量，统一参数并报告样本级QC，不能只比较call总数。若要改变bin size或染色体范围，另建结果目录并重新完成统计、分段和调用，避免把不同口径混在一个汇总表。先判断是输入、参考还是阶段耗时限制，再按实测配置CPU、内存和存储。

参考资料

CNVnator官方README：使用流程与参数 官方文档
CNVnator官方安装说明及依赖 官方文档