Somatic Copy Number Variation

Coming soon in GATK4 alpha:
New implementation of ReCapSeg
GATK development roadmap

Alpha GATK 4: cloud-friendly and more scalable (Apache Spark) + extended functionality (CNVs, Picard)

https://github.com/broadinstitute/gatk

First new feature in GATK 4 is a reimplementation of ReCapSeg
Copy Number Variations (=Alterations) can be dramatic

<table>
<thead>
<tr>
<th>per genome (3 x 10^9 bp)</th>
<th>per Mbp</th>
<th>somatic alteration type</th>
<th>acronym</th>
</tr>
</thead>
<tbody>
<tr>
<td>1000s to 10,000s</td>
<td>0.33 to &gt;1000</td>
<td>single nucleotide variations</td>
<td>SNV or SNP</td>
</tr>
<tr>
<td>100s to 1,000s</td>
<td>&lt; 1</td>
<td>small insertions &amp; deletions</td>
<td>Indel</td>
</tr>
<tr>
<td>100s to 1,000s</td>
<td>&lt; 1</td>
<td>structural variations</td>
<td>SV</td>
</tr>
<tr>
<td>100s to 1,000s</td>
<td>&lt; 1</td>
<td>copy number alterations</td>
<td>CNA or CNV</td>
</tr>
</tbody>
</table>

Spectral karyotyping paints each chromosome pair with a color
Alterations can vary dramatically between cancers and within cancers
Overview of the somatic CNV discovery workflow

Underlying method (CapSeg) has been around for years and has been cited in many papers.

**Start:**
- Genome reference
- Reads (BAM file)
- Targets (exome or gene panel)
- Panel of normals (PoN)

**End:** Amplification/deletion calls and copy ratio for each segment
Step 1: Collect proportional coverage

```java
java -jar GATK4.jar ExomeReadCounts \
-R <ref_genome> \
-I <input_bam_file> \
-O <pcov_output_file_path> \
-exome <target_BED> \
-transform PCOV -exonInfo FULL \
-groupBy SAMPLE -keepdups
```
Collecting proportional coverage is just a matter of counting reads

First collects coverage per read group at each target:

![Diagram of dark region as target, purple reads counted, green are not]

Then divides coverage by total number of reads per sample.

Proportional coverage file

```
##fileFormat = tsv
##commandLine =
org.broadinstitute.gatk.tools.exome.ExomeReadCounts ...snip...
##title       = Read counts per target and sample
CONTIG  START   END     NAME    SAMPLE1
1       12200   12275   target1    1.150e-05
1       13505   13600   target2    1.500e-05
1       31000   31500   target3    7.000e-05
....snip....
```
Step 2: Create normalized coverage profile

```
java -jar GATK4.jar NormalizeSomaticReadCounts \
   -I <pcov_input_file_path> \
   -T <target_BED> \
   -pon <pon_file> \
   -O <output_target_cr_file> \
   -FNO <output_target_fnt_file> \
   -BHO <output_beta_hats_file> \
   -PTNO <output_pre_tangent_normalization_cr_file>
```

Also requires -Djava.library.path=<hdf_jni_native_dir> (omitted for simplicity)
The PoN is used for normalization

- The PoN is created by collecting proportional coverage from each Normal sample

  ExomeReadCounts (on each Normal) -> CombineReadCounts -> CreatePanelOfNormals

- PoN stores median proportional coverage per target across the panel (target factors)

- Case sample is normalized against the target factors

  => First estimate at copy ratio per target in a case sample
Normalized coverage profiles are still very noisy

- Some substructure contributed by sequencing artifacts
  => would lead to hypersegmentation

- Lower coverage $\rightarrow$ increased noise
Important to choose appropriate Normal samples for PoN

• Use separate PoNs for different sequencing platforms
  – Protocol change in sample prep
  – Target region change

• Identify the samples
  – Blood normals without large events (see PoN QC)
  – Young subjects without bloodborne cancers
  – How many?
    • Recommend 40 or more
    • Depends on noisiness of data – more never hurts

• QC steps included to eliminate samples with large CNVs
Second round of (tangent) normalization cleans them up

Normalized coverage profile

```
#fileFormat = tsv
#commandLine = ....snip....
#title = ....snip....
name    contig  start   stop    SAMPLE1
target1  1      12200   12275   -0.5958351605220968
target2  1      13505   13600   -0.2855054918109098
target3  1      31000   31500   -0.11450116047248263
....snip....
```
Step 3: Segment coverage profile

java -jar GATK4.jar PerformSegmentation \
-S <sample_name> \
-T <normalized_coverage_file> \
-O <output_seg_file> \
-log
Segmentation produces... segments!

<table>
<thead>
<tr>
<th>Sample</th>
<th>Chromosome</th>
<th>Start</th>
<th>End</th>
<th>Num_Probes</th>
<th>Segment_Mean</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAMPLE1</td>
<td>1</td>
<td>12200</td>
<td>70000</td>
<td>18</td>
<td>0.841235</td>
</tr>
<tr>
<td>SAMPLE1</td>
<td>1</td>
<td>300600</td>
<td>163000</td>
<td>337</td>
<td>1.23232323</td>
</tr>
</tbody>
</table>

Segment boundaries

Segmented coverage file

X axis = targets
Step 4: Plot segmented coverage profile

```
java -jar GATK4.jar PlotSegmentedCopyRatio \
  -S <sample_name> \
  -T <normalized_coverage_file> \
  -P <pre_normalized_coverage_file> \
  -seg <segmented_coverage_file> \
  -O <output_seg_plot> \
  -log
```
Step 5: Call segments

Normalized coverage  \(\text{from step 2}\) \rightarrow \text{CallSegments} \rightarrow \text{Called segments} \rightarrow \text{Segmented coverage}  \(\text{from step 3}\) \rightarrow \text{Sample name}  \(\text{from SM tag}\)

\text{java -jar GATK4.jar CallSegments} \\-T <normalized\_coverage\_file> \\-S <seg\_file> \\-O <output\_called\_seg\_file> \\-sample <sample\_name>
Each segment is given a call (3 possibilities)

<table>
<thead>
<tr>
<th>Call</th>
<th>X axis = targets</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Call</th>
<th>X axis = targets</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Call</th>
<th>X axis = targets</th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
</tr>
<tr>
<td>-</td>
<td></td>
</tr>
</tbody>
</table>

### Called segments

<table>
<thead>
<tr>
<th>Sample</th>
<th>Chromosome</th>
<th>Start</th>
<th>End</th>
<th>Num_Probes</th>
<th>Segment_Mean</th>
<th>Segment_Call</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAMPLE1</td>
<td>1</td>
<td>12200</td>
<td>70000</td>
<td>18</td>
<td>0.841235</td>
<td>-</td>
</tr>
<tr>
<td>SAMPLE1</td>
<td>1</td>
<td>300600</td>
<td>1630000</td>
<td>337</td>
<td>1.23232323</td>
<td>0</td>
</tr>
</tbody>
</table>

....snip....
Recap of the CNV calling workflow

Reference → Reads

Targets → Collect proportional coverage

Proportional coverage profile

Normalize → Segment

Segmented coverage profile

Call segments → Called segments

Plot segmented coverage → Plots
Further reading

http://gatkforums.broadinstitute.org/categories/recapseg-documentation