CGA WES Characterization Pipeline User Guide

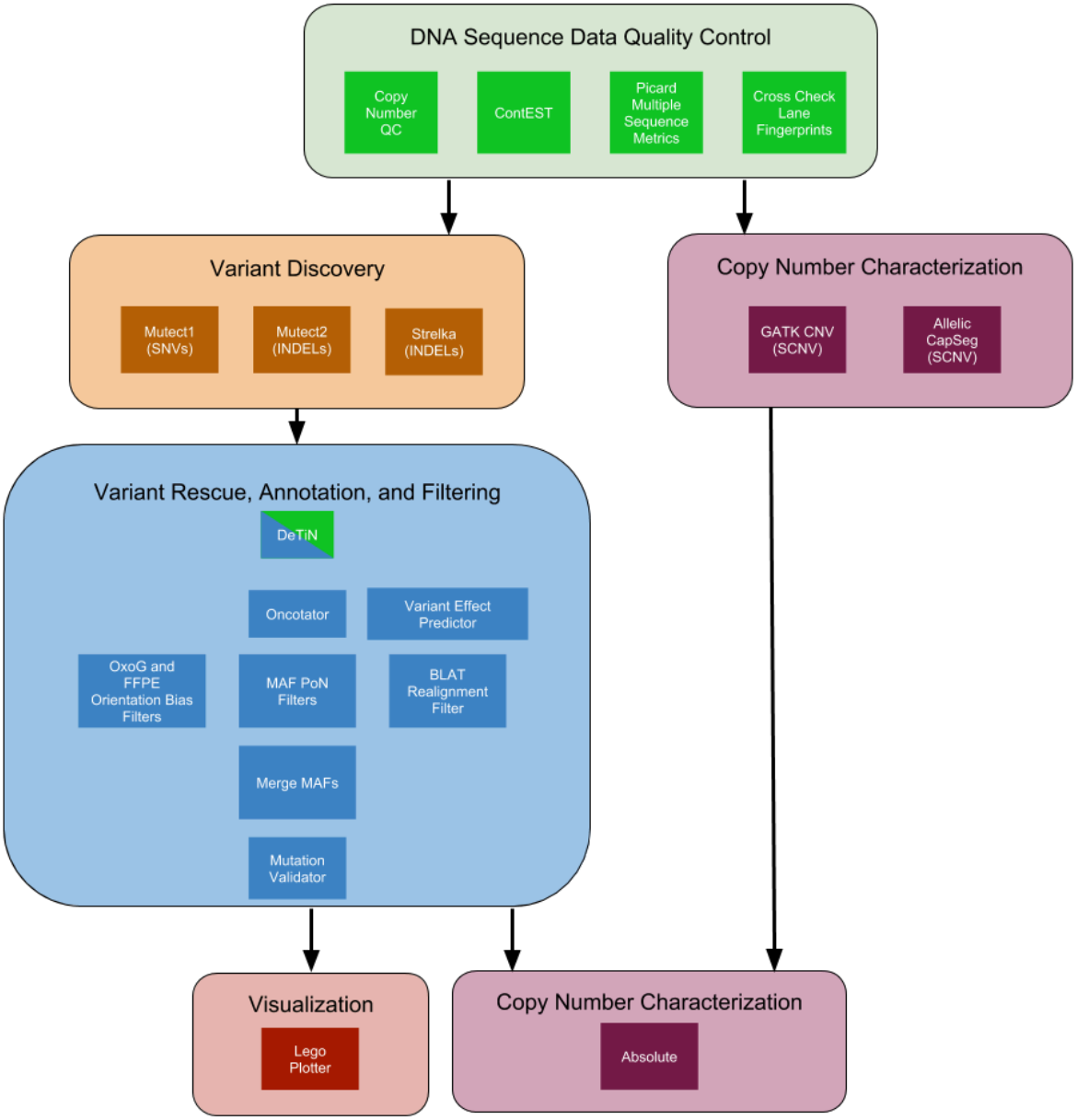
January 4, 2019

The CGA WES Characterization Pipeline is the standard computational workflow Getz Lab analysts employ when characterizing a tumor sample's somatic variants through contrastive computational analysis of matched tumor-normal WES BAMs. The pipeline includes state-of-the-art tools for quality control (QC) and characterization of paired (tumor/normal) whole exome sequencing data. The outputs of the pipeline include:

- QC reports that allow researchers to quickly identify quality issues with their whole exome sequencing data (e.g., contamination, sample or lane swaps).
- MAF (mutation annotation format) files containing annotated somatic variants that were called by the pipeline. While the pipeline will generate several versions of MAF files for each tumor/normal pair, each reflecting the results of a different filtering or annotation task, users will typically want the final validated maf, which only contains validated variants that have passed all of the pipeline's filtering tasks.
- Variant Effect Predictor (VEP) reports
- Copy Number and Allelic Copy Number reports
- Reports of sample purities and discrete (absolute) copy number profiles.
- Data visualizations, including
    - Lego plots of mutation spectra
    - Mutation allele fraction distributions and plots of tumor LOD values
    - Mutation breakdowns
    - Mutation coverage spectra

The pipeline is organized into five modules depicted in the figure below: (1) DNA Sequence Data Quality Control, (2) Variant Discovery, (3) Copy Number Characterization, (4) Variant rescue, Annotation and Filtering, and (5) Visualization. Each of these modules consists of multiple tools, many of which run in parallel. The tools are described below.

The current version of the pipeline only operates on hg19-aligned data.



DNA Sequence Quality Control

It is important to quantify and understand any sources of noise, artifacts and other irregularities in the data. Typically, analysts will run the pipeline through completion and then review the QC reports for any evidence of sample or lane swaps, contamination or sequencing issues. If there are quality issues, problematic sample pairs may be removed from downstream analysis, read groups may be blacklisted and the analysis pipeline re-run.

Sample and Lane Swaps

This category of problems can arise at different levels of sample handling, including collection, library preparation, annotation and informatics.

The ***CopyNumberQC*** tool compares DNA copy-number profiles of matched tumor and normal samples to identify potential tumor-normal swaps or identity mixups.

The ***PicardCrossCheckLaneFingerprints***[1,2] tool compares genotypes at common SNP sites to identify potential cross-individual sample or lane swaps and contamination.

Contamination

Sample contamination can appear in four ways: (i) cells from a different species (either representing biology, e.g., HPV in cervical cancer, xenografts, or lab contamination); (ii) cells from a patient's non-cancerous cells found in the tumor DNA sample (i.e. tumor "purity" or "cellularity"); (iii) tumor-in-normal contamination of the patient-matched normal sample (may occur in hematological malignancies or as the field effect in solid tumors); or (iv) contamination of cells from one individual with cells from other individuals. The pipeline incorporates several tools to assess these types of contamination in DNA and provide additional quality control (QC) metrics.

ContEst[3] estimates the amount of cross-sample contamination in next generation sequencing data. Its fraction contamination estimates are used downstream as inputs to Mutect1 and Mutect2 (two of the workflow's somatic mutation calling tools).

DeTiN[4] estimates Tumor-in-Normal (TiN) contamination level. Detection sensitivity for somatic variants is greatly reduced when the matched normal sample is contaminated with tumor cells. To overcome this limitation, we use DeTiN to estimate Tumor-in-Normal (TiN) contamination levels and, in cases affected by contamination, improve sensitivity by reclassifying events initially classified as germline due to the presence of tumor cells in the matched normal as somatic.

Capture efficiency, uniformness of coverage and other metrics

The pipeline runs several of the Picard metrics programs as part of the ***PicardMultipleMetrics***[1,5] Task (run on both tumor and normal samples). Below is a brief summary of the metrics run by this task. Details are available at https://broadinstitute.github.io/picard/command-line-overview.html#CollectMultipleMetrics.

- *CollectAligmentSummaryMetrics* - compares a BAM file to the reference file (FASTA), and provides statistics to assess the quality of the read alignments and the proportion of the reads that passed signal-to-noise threshold quality filters.

- *CollectInsertSizeMetrics* - collects metrics about the insert size distribution of a paired-end library.
- *QualityScoreDistribution* - Charts the distribution of quality scores.
- *MeanQualityByCycle* - Collects mean quality by cycle.This tool generates a data table and chart of mean quality by cycle from a BAM file. This metric gives an overall snapshot of sequencing machine performance. For most types of sequencing data, the output is expected to show a slight reduction in overall base quality scores towards the end of each read. Spikes in quality within reads are not expected and may indicate that technical problems occurred during sequencing.
- *CollectBaseDistributionByCycle* - Charts the nucleotide distribution per cycle in the BAM file in order to enable assessment of systematic errors at specific positions in the reads.
- *CollectSequencingArtifactMetrics* - Collects metrics to quantify single-base sequencing artifacts.
- *CollectQualityYieldMetrics* - Collects metrics about reads that pass quality thresholds and Illumina-specific filters. This tool evaluates the overall quality of reads within a bam file containing one read group.
- *CollectGcBiasMetrics* - This tool collects information about the relative proportions of guanine (G) and cytosine (C) nucleotides in a sample. Regions of high and low G + C content have been shown to interfere with mapping/aligning, ultimately leading to fragmented genome assemblies and poor coverage in a phenomenon known as 'GC bias'.
- *ConvertSequencingArtifactToOxoG* - This tool extracts 8-oxoguanine (OxoG) artifact metrics (e.g., OxoQ scores) from the output of CollectSequencingArtifactsMetrics.
- *CollectHSMetrics* - Collects hybrid-selection (HS) metrics.

Oxidative and chemical DNA damage and other artifact modes

Artifact modes may arise due to sample storage and DNA damage during library prep. A common class of artifacts occurs in a context and “orientation” specific way, meaning pre-existing damage on one of the DNA strands is locked in a specific conformation when adapters are added to the DNA fragments.

As a step in the ***PicardMultipleMetrics*** Task (see above) we run the Picard *CollectSequencingArtifactMetrics* tool, which collects metrics to quantify single-base sequencing artifacts. We then run the Picard *ConvertSequencingArtifactToOxoG* tool to extract and report on 8-oxoguanine (OxoG) artifact metrics (e.g., OxoQ scores) from the output of CollectSequencingArtifactsMetrics.

QC is an Ongoing Process Throughout Sequencing Data Analysis

Note that data QC is not simply the first step of an analysis, but rather an ongoing process at each step. For example, one of the QC steps outlined above (***DeTiN***) requires variant detection as input. Even the final steps of interpretation may reveal clear signs of artifacts. Catching

potential problems cannot always be done automatically, but a good analysis tool will provide clues to the analyst when QC issues require further attention, and perhaps additional filters.

Variant Discovery

The Variant Discovery module currently employs three callers: MuTect1 for detection of single-nucleotide variants (SNVs), and Strelka and MuTect2 for detection of insertions/deletions (INDELs). The Variant Discovery steps are arguably the pipeline's most important since callers' performance affects all downstream analyses and directly impacts any scientific conclusions drawn from the data. Our selection and tuning of callers will evolve over time as algorithms, sequencing technologies and benchmarking improve.

Currently we use ***MuTect1***[6] for somatic SNV discovery. MuTect1 has established itself as one of the leading somatic mutation callers, with low artifact rates and high sensitivity across the allele fraction spectra. We also use MuTect1 to "force-call" (evaluate read evidence) at known clinically relevant mutation sites and cancer mutation hot-spots.

We currently employ a hybrid strategy for the detection of INDELs. We use ***Strelka***[7] for our somatic INDEL discovery. Strelka uses realignment of reads after detection of candidate INDELs and a bayesian likelihood model to call INDELs. We then use ***MuTect2***[8] to independently call INDELs, and annotate the Strelka-called INDELs that are also called by MuTect2.

The focus of our benchmarking efforts to-date has been in the validation of the workflow's SNV calling. As we expand our benchmarking to include thorough validations of the workflows INDEL calling, we expect we will change the INDEL calling strategy to include new tools and procedures.

Variant Rescue, Annotation and Filtering

Following initial variant discovery, we process the resulting SNV and INDEL calls through a number of steps aimed at (1) rescuing somatic calls that were incorrectly classified as germline events due to tumor-in-normal contamination, (2) annotating SNVs and INDELs with information relevant to cancer researchers, and (3) removing false positives attributable to artifact modes, misinterpretation of germline variants as somatic, and misalignment.

Variant Rescue: DeTiN

As described above, DeTiN both estimates Tumor-in-Normal contamination and, in cases affected by contamination, improves sensitivity by reclassifying as somatic events initially classified as germline due to the presence of tumor cells in the matched normal.

Annotation: Oncotator and VEP

Oncotator[9] and **Variant Effect Predictor** (**VEP**)[10] are used to annotate SNVs and INDELs. Oncotator attaches Genomic Protein and Cancer Variant annotations to SNVs and INDELs. The annotations are drawn from several sources, including genomic annotations from GENCODE, site-specific protein annotations from UniProt, functional impact predictions from dbNSFP, cancer mutation frequency annotations from COSMIC, overlapping mutations from the Cancer Cell Line Encyclopedia, cancer variant annotations from ClinVar, and non-cancer variant annotations from dbSNP and 1000 Genomes. The VEP tool determines the effect of the detected somatic SNVs and INDELs on genes, transcripts, and protein sequences and reports its results in text and html files.

Post-Discovery Filtering

In the process of sample collection, preservation, DNA amplification and sequencing, there are a number of events that could introduce errors in DNA sequences leading to false positive calls coming out of Variant Discovery. Although built into MuTect1 are filters to eliminate false positives due to inaccurate read placement and non-independent sequencing errors, and a Panel of Normals (PoN) filter that rejects candidate variants present in two or more of the normal samples provided in the PoN (effectively a blacklist), we have found it beneficial to apply a number of post-discovery filters to remove false positive calls.

Orientation Bias Filters

Two known artifact types that the pipeline incorporates specific filters for are artifacts that arise from oxidation of Guanine to 8-Oxoguanine (“OxoG” artifacts) and artifacts that arise from formaldehyde deamination of Cytosines resulting in C-to-T transition mutations (formalin-fixing paraffin-embedding or “FFPE” artifacts). Though only seen in a low percentage of reads, these artifacts can have a significant impact on our ability to confidently call rare somatic mutations.

A key characteristic of these artifacts is an “orientation-bias” between read pairs supporting the variant - in terms of the sequenced order of the supporting reads within the pair (1 or 2) and the mapped strand of the read (F or R for forward and reverse). For the OxoG artifact, this presents itself as an excess of G>T sites in F1R2 read pairs and an excess of C>A sites in F2R1 read pairs. For FFPE, this presents itself as an excess of C>T sites in F1R2 read pairs and an excess of G>A in F2R1 read pairs.

The pipeline’s OxoG and FFPE ***Orientation Bias Filters***[11] use the characteristic read-pair orientation bias of these artifactual mutations to identify and filter out somatic SNVs likely to be attributable to these two artifact modes.

MAF PoN (Panel of Normals) Filters

The ***MAF Panel of Normals (PoN) Filter***[12] is a highly effective tool for filtering false positive germline variants and common artifacts from mutation calls.

In a paired tumor-normal analysis it is statistically possible to miss germline variants in low coverage regions due to a variant going undetected in the normal sample; this can result in miscalling germline variants as somatic. Application of the PoN filter reduces the rate of false positive germline calls by effectively increasing the sequencing depth at these low coverage regions through the leveraging of a PoN's normal cohort.

Interestingly, although the MAF PoN Filter is an effective way to remove germline variants, most of the variants it flags for removal are in fact recurrent sequencing artifacts. The reason for this will become apparent in the following brief description of how the MAF PoN filter operates.

For each genomic position, a provided "token PoN" file encodes the empirical distribution of alternate allele read counts and alt allele fractions across all of the PoN's normal samples. For each variant call in the tumor sample, the filter models the variant's allelic fraction as a beta distribution parameterized by its alternate and reference read counts. A weighted sum of the discretized beta distribution and the PoN's empirical distribution at that variant's genomic position serves as a score for that position. If the score is above a certain threshold, the site gets flagged as being consistent with the empirical distribution observed in the PoN. For example, if a site recurrently harbors low-level sequencing noise in the PoN and it is called at low-allelic fraction by MuTect1, it's MAF PoN Filter score would reflect consistency between the read support for the allele in the tumor sample and the read support for the allele in the PoN, resulting in it being flagged for removal. If, however, the read support for the allele in the tumor sample were significantly higher than what was observed in the PoN, its likelihood score would be lower and the variant would be retained.

To be effective at filtering out germline variants and artifacts, the number of normal samples from which the PoN is constructed must be large: a hundred or more is viewed as satisfactory, but a thousand or more will significantly improve sensitivity to remove common artifacts. In order to be effective in filtering out very low allele count artifacts, large numbers of normals from which to construct the allele's empirical read count distribution are needed; in addition, the library prep and sequencing technology used for the PoN's normal samples would ideally mirror that used in the processing of the the matched tumor/normal pairs under study so as to capture sequencing artifacts characteristic of that library prep and sequencing platform.

This pipeline is designed to run an arbitrary number of instances of the MAF PoN filter, each with a distinct token PoN. These MAF PoN Filter instances are run in parallel, and their results are aggregated downstream (in order to be included in the final MAF, a variant must pass all filters). We typically run two instances of the MAF PoN filter: one employing a Token PoN assembled from 8334 TCGA controlled access normal samples, and a second employing a Token PoN assembled from a private collection of normal samples. We occasionally run a third instance using a token PoN assembled from the complete set of matched normals used within the study being analyzed; care must be taken, however, to ensure there is no tumor-in-normal contamination in those normal samples, and that the number of normals making up this token PoN is large enough (> 100) to be compatible with the default likelihood score threshold currently used by the MAF PoN Filter.

Configuring MAF PoN Filter Instances

The MAF PoN Filter instances are configured through a TSV file that is an input to the workflow (see the workflow-level `PONS_list` input parameter in Appendix A). This TSV file contains three columns: `pon_name`, `pon_url` and `pon_threshold`. `pon_name` is the label used for annotating the resulting MAF file with the filter instance's results; `pon_url` is the Google Cloud Storage URL pointing to the token pon file employed by the MAF PoN Filter instance; and `pon_threshold` for deciding whether an alternate allele site in the tumor sample is consistent with what is observed in the PoN (in which case it would be filtered out). This threshold value was determined by empirical means. We typically set it to -2.5 but have exposed as a configuration parameter. We do not recommend you alter this value. Below is an example PONS_list file.

pon_name	pon_url	pon_threshold
TCGA_8334	gs://getzlab-workflows-reference_files-tcga_ca/hg19/pon/maf_pon_filter/controlled_access_token_pon_from_tcga8000.final_summed_tokens.hist.bin	-2.5
GETZLAB_PRIVATE	gs://getzlab-workflows-reference_files_getzlab_ca/hg19/pon/mav_pon_filter/getzlab_controlled_access_token_pon_from_ice355.final_summed_tokens.hist.bin	-2.5

There is a separate workflow for constructing a token PoN from one's own collection of normal BAMs. This workflow id described in Appendix B.

BLAT Realignment Filter

Artifactual mutations also commonly occur due to poorly mapped reads (e.g., misaligned reads in homologous regions of the genome). This effect is amplified if other sources of artifacts alter the genomic sequence of the read as whole (e.g. oxoG or FFPE damage). These effects often cause reads with multiple candidate somatic calls, which in fact are simply misalignment errors. Since commonly used read alignment methods are bound to sacrifice some alignment accuracy for speed, we use a slower, but more sensitive and accurate alignment method on candidate mutations sites and their alternative reads to provide high quality confirmation of a mutation call.

We employ the ***BLAT Realignment Filter***, which applies BLAT[13] (BLAST-like Alignment Tool), a high sensitivity realigner, to each candidate mutation's alternative allele reads. For each read concordant with a somatic mutation call, the alternative alignments suggested by BLAT are examined. Mutations that are only supported by reads which are ambiguously mapped are removed. This approach allows fast filtering of large numbers of candidate mutation calls since only candidate variant-supporting reads need to be examined.

Mutation Validator[14]

The WES Characterization Pipeline employs a 'Mutation Validator' tool to identify validation evidence at variant sites using sequencing data from other sequencing experiments; for example, sequencing data from RNASeq, Whole Genome or Targeted Panel experiments.

For each validation data type, the corresponding normal sample data is parsed to estimate the maximum noise alternate allele fraction. (If, as in the case of RNA-seq data, the validation data type has no normal sample, the normal exome sample is used to estimate the noise.) Then a 99% upper limit alternate allele count in the tumor at this maximum noise allele fraction (`min_val_count`) is determined, which becomes the threshold validation read count in the tumor sample. See [14] for details.

In the current version of the pipeline, Mutation Validator results are used to annotate the SNVs and INDELs that passed all of the post-discovery filters; however, the results are not used to further filter the variant calls.

Copy Number Characterization

While coding somatic SNVs and INDELs are often strong drivers in cancer, somatic copy number alterations can be equally important in driving cancer formation and development. In some circumstances copy number aberrations are easier for the cell to acquire, and can often be founding events.
The ability to accurately determine copy number states of individual genomic regions in cancer cell populations allows researchers to accurately estimate cancer fractions of mutations and interpret copy number information in an unbiased way.

Tumor cell populations are commonly aneuploid which poses additional challenges when trying to estimate tumor cell population structure and abundance via next generation sequencing. To accurately determine the subclonal structure of a biopsy the local copy number for each SNV and INDEL is required.

The WES characterization pipeline includes tools that collect raw coverage, normalize the coverage by using a large panel of normal diploid samples and then estimate the allelic and absolute copy number profiles for each sample. Obtaining absolute copy number estimates (i.e. number of each of the homologous chromosome copies in a cell subpopulation) is critical in detecting homozygous deletion, LOH and copy-neutral LOH events, gains, whole genome duplications and other copy number events.

The pipeline employs the ***GATK3 CNV[15] pipeline*** for the generation of accurate relative copy-number profiles from the whole exome sequencing data and reference/alternate read counts at heterozygous SNP sites present in both the normal and tumor samples.

After accurate proportional coverage profiles are generated for a sample, the pipeline employs the ***Allelic CapSeg***[16] tool to generate a segmented allelic copy ratio profile.

Allelic copy number profile and mutational call data are then modeled jointly by ***ABSOLUTE***[17] to produce purity for the samples and a discrete (absolute) copy number profile. Individual mutation events are modeled based on this profile and cancer cell fractions and confidence intervals are assigned and reported for each variant call.

Visualization

The pipeline's ***lego_plotter*** tool generates several visualizations of the pipeline's somatic variant calls. These include:

- Lego plot of the tumor sample's mutation rate spectrum
- Lego plot of the tumor sample's mutation count spectrum
- Lego plot of the tumor sample's normalized mutation count spectrum
- Lego plots of the tumor sample's mutation spectrum separated by forward and reverse strands
- Lego plots of the tumor sample's mutation count spectrum sliced by allele fraction
- Mutation allele fraction distribution
- Mutation LOD value distribution
- Mutation coverage spectrums

In addition, the copy number analysis produces several visualizations, including copy number profiles, of its results.

Pipeline Inputs

Workflow inputs are defined at both the workflow and task level. Workflow-level inputs are used by multiple tasks within the workflow; task-level inputs are specific to a single task. Appendix A contains tables listing the workflow- and task-level inputs.

The majority of the input files are reference files; most users of the pipeline will employ the same collection of reference files. We have placed copies of these reference files on google cloud storage. The public FireCloud workspaces we created to provide templates for the running of this workflow contain workspace attributes that point to these reference files and method configurations that reference these workspace attributes.

Workflow inputs that are dependent on the user's data model and experimental design are highlighted in Appendix A's tables. Most of the highlighted inputs are sample or pair specific, and hence will be drawn from FireCloud data model attributes. The exceptions are:

- workflow-level inputs `targetIntervals` and `baitIntervals`, which are dependent on the exome capture kit (e.g., Agilent, ICE)
- workflow-level input `PONS_list`, which contains a list of token PoN files; a separate instance of the MAFPonFilter task is run for each token PoN file
- optional task-level inputs `mutation_validator.tumorRNABam` and `mutation_validator.tumorRNABamIdx`

Reference Files

As mentioned in the previous section, copies of all reference data files have been placed on Google Cloud Storage and template workspaces contain workspace attribute definitions referencing these files. Appendix A's tables map workflow input values to workspace attributes, and hence reference files.

Panel-of-Normals Files

We have placed on google cloud storage a "TCGA Token PoN" file that we frequently employ within the Getz Lab. The Token PoN file is assembled from 8334 TCGA normal BAM files. Because TCGA BAM files are controlled access, so is this Token PoN; i.e., it may only be used by researchers who have dbGaP authorization to access TCGA controlled access data.

Users may construct Token PoN files from their own collections of normal BAMs. The larger the number of samples, the greater sensitivity the filter will have to the detection of low allele-fraction germline variants and artifacts. Sensitivity to detect artifacts also relies on matching the normal collection's library prep and sequencing methods with those use in the processing of the tumor/normal pair that is undergoing somatic characterization.

We provide a separate workflow for constructing a token PoN from one's own collection of normal BAMs. See Appendix B for details.

Pipeline Outputs

Data QC Reports

Copy Number QC

The report generated by the CopyNumberQC task identifies tumor-normal swaps or identity mixups. The pair-level attribute `copy_number_qc_report` references this report. See Copy Number QC for an explanation of how to interpret this report's contents.

Picard Cross Check Lane Fingerprints

The PicardCrossCheckLaneFingerprints task calculates a LOD (logarithm of odds) value for each distinct pair of read groups across a pair's tumor and normal BAM files. These values may be used to identify potential cross-individual sample or lane swaps, and contamination. The minimum LOD value calculated for a pair is written to the pair-level attribute `cross_check_fingprt_min_lod_value`. The task also generates a report whose GCS URL is written to the pair-level attribute `cross_check_fingprt_report`. See Picard Cross Check Lane Fingerprints for an explanation of how to interpret these outputs.

Contamination Estimation

The ContEst task writes an estimate a cross-individual contamination for each tumor sample to the pair-level attribute `fracCont`. See Contamination Estimation for an explanation of how to interpret this estimate.

DeTiN

DeTiN estimates tumor-in-normal (TiN) contamination level, and in cases affected by contamination, improves sensitivity by rescuing somatic variants that were initially discarded due to TiN contamination. TiN estimates are written to the pair-level attributes TiN and TiN_CI. See DeTiN for an explanation of how to interpret these estimates.

Picard Multiple Metrics

The PicardMultipleMetrics task is run on both tumor and normal samples. Results are written back to the data model as attributes on the corresponding pair entity, with tumor and normal results distinguished by the "`tumor_bam_`" and "`normal_bam_`" prefixes, respectively, attached to the attribute names. See PicardMultipleMetrics for a listing of the reports generated by this task and explanations of how to interpret their contents.

SNV and INDEL Calls

The final set of validated SNVs and INDELs called by this pipeline are written to a MAF file referenced by the pair-level attribute `mutation_validator_validated_maf`. This MAF file contains SNVs and INDELs that have passed ALL of the workflow's filtering tasks. All reported SNVs will have been called by the Mutect1 task. All reported INDELs will have been called by the Strelka task. If an INDEL was also call by Mutect2, that will be indicated by an entry in the MAF file's Other_Variant_Caller column.

In addition to this final MAF, the workflow generates 15 intermediate MAFs which are also written to the data model. These intermediate MAFs are useful for tracking the origin of a SNV or INDEL call, the SNVs and INDELs that were identified by MuTect1 and Strelka but then removed by the downstream filtering tasks, and the filtering tasks that were responsible for the

removal of the dropped variants. The most useful of these intermediate MAFs is the one referenced by the `after_filters_merged_maf` pair attribute. This MAF annotates each variant called by Mutect1 and Strelka with a list of "Passed_Filters" and a list of "Failed_Filters". The Failed_Filters annotation identifies the filters which were responsible for the removal of a variant call from the final validated maf; only those variant calls with an empty list of Failed_Filters will remain in the final validated maf.

Copy Number Variants

The gatk_acnv_only and absolute tasks creates several output files characterizing the tumor sample's somatic copy number variants. The most important of these are:

TBD

Lego Plots

Lego plots provide 3-D visualizations of single-base mutation rate and count spectra across a 3-base context, which consists of the base that is subject to mutation and it's two flanking bases (5' preceding and 3' trailing). Each lego plot displays the three base context on the x and y axes, and mutation rates or counts on the z axis, with different mutation modes (i.e, C>T, C>A, C>G, A>G, A>C, A>T) in different colors. Each mutation mode is displayed across a 4X4 grid in the x-y plane, representing the full range of flanking bases.

This workflow's lego_plotter task generates an html report (referenced by the pair-level attribute `mut_legos_html`) that contains multiple lego plots characterizing a tumor sample's single-base mutations. Note that the lego_plotter task was originally developed to run on sets of pairs and some of the plots included in the html report were meant to combine information from multiple pairs. Since we are running the task on single pairs, the combined view plots become redundant (they will be dropped in future updates of the workflow).

Here is a description of the report's plots, in their order of appearance:

A. Mutation rate spectrum - lego plot with the z-axis in representing normalized base coverage (mutation rate or mutations per Mb) in the region being evaluated, which in our case is the whole exome.
B. Mutation count spectrum - lego plot with z-axis in units of mutation counts
C. Smaller lego plots with the z-axis in units of mutation rates for each sample in the set; given we are running lego_plotter on a single T/N pair, this plot is just a smaller version of A.
D. Mutation sample normalized spectrum - normalized aggregation of the mutation count spectrums for all samples in the set, where the z-axis is the mutation count normalized to 1 for each pair in the set. This plot was designed to make each pair in a set contribute equally to a single lego plot, rather than letting hypermutators dominate the aggregated

lego. When the lego plotter is run on a single pair, as is our case, then this plot is just a normalized version of B.

E. Lego plots of mutation counts for two possible orientations. Any lack of symmetry between the two plots is evidence of an orientation bias artifact that has not been filtered.
F. Lego plots of weighted counts for the two possible orientations. Each mutation is given a weight of 1 that is divided between the two possible orientations. Lack of symmetry between the two plots is also evidence of an orientation bias artifact.
G. Lego plots of mutation counts in four allele fraction bins (`0.00≤AF<0.10; 0.10≤AF<0.25; 0.25≤AF<0.50; 0.50≤AF<1.00`).

H-K. These are historical plots that were made of the analysis of oxoG artifacts and should be ignored. They will be removed in future versions of the workflow.

How to Use this Workflow

1. Clone a template workspace

We have created two public FireCloud template workspaces:

- `broad-fc-getzlab-workflows/CGA_WES_Characterization_TCGAControlledAccess`
- `broad-fc-getzlab-workflows/CGA_WES_Characterization_OpenAccess`

These workspaces each contain (1) a method configuration that may be used for configuring and launching the CGA WES Characterization pipeline on FireCloud and (2) workspace attribute definitions referenced by the method configuration. You will begin by cloning one of these workspaces.

The TCGA Controlled Access template workspace is in the TCGA-dbGaP-Authorized FireCloud Authorization Domain. This means you can only clone the workspace if you have dbGaP authorization for TCGA controlled access data. The pipeline in this workspace is configured to run an instance of the MAF PoN filter that uses the "TCGA_8334" Token PoN file described earlier. The workspace attribute MAF_PON_FILTER_LIST points to a TSV file that configures a single instance of the MAF PoN Filter, identified as TCGA_8334.

The Open Access template workspace is not in any authorization domain; TCGA controlled access data, including the TCGA_8334 Token PoN File, should not be accessed within the workspace. To use this template workspace, you must reference one or more Token PoNs that are not constructed from TCGA controlled access data. We provide a public workflow for creating your own Token PoN File (see Appendix B). You will need to create your own TSV file listing your Token PoNs and set the

MAF_PON_FILTER_LIST workspace attribute to the path of that TSV file on Google Cloud Storage.

If you do not have dbGaP authorization to access TCGA controlled access data, or you intend on sharing the workspace with colleagues who do not have the access, you should clone the Open Access workspace; otherwise clone the TCGA Controlled Access workspace as this will allow you to take advantage of the TCGA_8334 Token PoN.

2. **Upload your sample data to your workspace clone**

Using the gsutil utility or Google's Cloud storage browser, upload your data files to the clone's workspace bucket. Uploaded data files will include the Tumor BAM/BAI and Normal BAM/BAI files. If you have additional sequencing data generated using other experimental experimental strategies (e.g., RNASeq, WGS, Targeted Panel) that will be used by the Mutation Validator tool, upload these files as well.

See (howto) Upload files to your Google bucket - video and (howto) Upload files to your Google bucket - for instructions on how to upload your files.

3. **Create Annotations**

Populate the workspace's data model with participant, sample and pair entities. This should be done be creating their respective TSV-formatted load files and uploading them to your workspace.

Participant annotations should include at a minimum:

- `participant_id`

Sample annotations should include at a minimum:

- `sample_id`
- `Participant`: reference to associated participant_id
- `WXS_bam_path`: Google Cloud Storage URL (gs://<bucket name>/<file path on bucket> to sample WES BAM file
- `WXS_bai_path`: Google Cloud Storage URL to sample WES BAI file

If you have RNASeq data that you plan to pass to Mutation Validator, include their URLs in the data model:

- `RNASeq_bam_path`
- `RNASeq_bai_path`

Most likely, you will only have RNASeq data for you tumor samples.

Pair annotations should include at a minimum:

- `pair_id`
- `case_sample`: reference to pair's tumor sample_id)
- `control_sample`: reference to pair's normal sample_id)

See (how to) Import metadata for detailed instructions on creation of TSV load files and uploading them to your workspace.

4. **(Optional) Create one or more Token PoN Files and update method configuration to reference them**

   If you are unable to use the TCGA Token PoN File, or decide to run additional instances of the MAF PoN Filter with Token PoN files assembled from non-TCGA normal cohorts, follow the instructions in Appendix B for creating a Token PoN file. Once you have assembled a collection of Token PoN files to use in your pipeline, ensure there are copies of them on cloud storage, create a MAF PoN Filter list file (see the Configuring MAF PoN Filter Instances section), copy it to google cloud storage and then update the workspace attribute MAF_PON_FILTER_LIST to point to the path of your MAF PoN filter list file on cloud storage.

5. **Launch the Characterization workflow on your cohort's paired tumor/normal samples**

   A separate instance of the workflow will run for each tumor/normal pair. The workflolegogiw takes approximately 6 hours to run (assuming no cache hits) and costs anywhere between 2 cents and a dollar per run, depending on the size of your BAMs and your success running the workflow's tasks on preemptible VMs (preemptible VMs run at 20% the cost of non-preemptible VMs).

   If any of these runs fail, see Error Handling.

   We strongly recommend you run the workflow on a single tumor/normal pair before running it on a large cohort (pair set).

6. **Error Handling**

   Common configuration errors that lead to workflow failures include:
   - Method configuration references non-existent workspace attribute
   - Method configuration references non-existent entity attribute
   - Method configuration references non-existent object on cloud storage (either directly or through a workspace attribute)
   - Workspace user is not authorized to access a file object reference in the method configuration

In these cases, correct the configuration error and re-run the workflow.

Common runtime errors that may lead to workflow failures include:

- A job (running on a VM) was unexpectedly stopped before it had the opportunity to complete. This occurs rarely, but when it does it is typically due to an intermittent error condition in the cloud infrastructure. Typically re-running the failed workflow will result in success. The re-running of the workflow will take substantially less time due to call caching.
- An intermittent failure in the FireCloud platform led to the failure. Attempt to re-run the workflow; if the same error continues to occur, search the firecloud online forum (https://gatkforums.broadinstitute.org/firecloud/categories/ask-the-firecloud-team) for descriptions of similar failures; if you are unable to find your problem on the forum, report it in a new discussion thread.
- A failure in the workflow's software led to the failure. Search the Cancer Genome Analysis forum (https://gatkforums.broadinstitute.org/firecloud/categories/cancer-genome-analysis) for a description of the problem; if you are unable to find a report of a similar problem, report it to the forum in a new discussion thread. **Please note that the Getz Lab makes no commitment to provide user support for its published workflows. We will monitor the forum for user-reported bugs and do our best to release timely workflow updates with bug fixes.**

7. Review of QC Metrics

7.1. CopyNumberQC

Inspect the html report generated by the CopyNumberQC task (our default method config binds the html file to the pair-level attribute `copy_number_qc_report`). Text at the top of the file will indicate whether tumor-normal swaps or identity mixups were identified. If the report reads: "NO MIXUPS DETECTED", then you are OK. If MIXUPS are detected and they are restricted to specific lanes, report details will indicate which lanes are at fault (they will be highlighted in RED) You may then provide a custom readGroupBlackList with the offending lanes added to the list and re-run the pipeline.

7.2. PicardCrossCheckLaneFingerprints

The PicardCrossCheckLaneFingerprints task calculates a LOD (logarithm of odds) value for each distinct pair of read groups across both the supplied tumor and normal BAM files. The LOD value calculated for each read group pair is a measure of the relative likelihood the two read groups come from the same individual. A LOD value of 0 indicates there is equal likelihood that the two groups come from the same individual vs. coming from different individuals. If the LOD is negative value -N, then it is 10^N more likely that the read groups come from different individuals than the same individual; if the

LOD is a positive value N, then it is 10^N more likely that the read groups come from the same individual than from diffent individuals.

This task flags a pairing of read groups from the same individual as OK if their LOD score ≥ 15. If 1 ≤ LOD < 15, then there is a suspiciously low concordance between the two read groups, but not sufficiently low to identify it as a problem. If LOD < 1, then the read group pairing is flagged as problematic.

First inspect the `cross_check_fingprt_min_lod_value` attribute attached to each pair in your cohort. If the min LOD value is less than 1 for a tumor/normal pair, inspect the html report generated by PicartCrossCheckLaneFingerprints for that pair (our default method config binds that report to the pair's `cross_check_fingprt_report` attribute), identify the problematic read groups (lanes), add them to your own readGroupBlackList and re-run the pipeline.

7.3. ContEst

Inspect the `fracCont` attribute that the characterization pipeline attaches to each pair in your cohort. For the analysis of TCGA data, Getz Lab analysts typically requested a resequencing of the tumor sample when fracCont exceeded 0.04; this, however, was an admittedly arbitrary threshold.

7.4. DeTiN

DeTiN estimates tumor-in-normal (TiN) contamination level, and in cases affected by contamination, improves sensitivity by rescuing somatic variants that were initially discarded due to TiN contamination. While deTiN's mutation recovery step improves detection sensitivity across all TiN values, at TiN values > 0.3, SSNV recovery is less effective with dramatic decreases in sensitivity; thus we recommend excluding from your analysis pairs with TiN estimates greater than 0.3. Alternatively, a highly contaminated normal may be used as a second tumor in a tumor-only analysis; however, tumor-only analysis is outside the scope of the pipeline described in this user guide.

The workflow writes the TiN estimates to the `TiN` and `TiN_CI` (confidence interval) pair entity attributes.

7.5. PicardMultipleMetrics

The PicardMultipleMetrics task is run on both tumor and normal samples. Results are written back to the data model as attributes on the corresponding pair entity, with tumor and normal results distinguished by the "`tumor_bam_`" and "`normal_bam_`" prefixes, respectively, attached to the attribute names. Analysts typically review the outputs of this task if downstream analysis results suggest a potential problem with the input BAMs.

7.5.1. {tumor_bam_ | normal_bam_} bam_validation

Results of Picard ValidateSamFile are written to files referenced by this pair of attributes. The file captures any formatting issues with the respecitive BAM file.

7.5.2. {tumor_bam_ | normal_bam_} allignment_summary_metrics

Results of Picard CollectAllignmentSummaryMetrics are written to files referenced by this pair of attributes. This file contains various metrics detailing the quality of the read alignments and the proportion of reads that passed machine signal-to-noise threshold quality filters. Quality filters are specific to Illumina data.

7.5.3. {tumor_bam_ | normal_bam_} bait_bias_summary, bait_bias_detail, pre_adapter_summary_metrics and pre_adapter_detail_metrics

Results of Picard CollectSequencingArtifactMetrics are written to files referenced by these four attributes. These files capture sequencing error artifacts arising from the hybrid selection protocols.

7.5.4. {tumor_bam_ | normal_bam_} base_distribution_by_cycle and base_distribution_by_cycle_metrics

Results of Picard CollectBaseDistributionByCycle are written to files referenced by these attributes. `base_distribution_by_cycle` references a chart of the nucleotide distribution per cycle in a bam file. It may be used to assess systematic errors at specific positions in the reads.

7.5.5. {tumor_bam_ | normal_bam_} gc_bias, gc_bias_summary_metrics and gc_bias_detail_metrics

Regions of high and low guanine and cytosine nucleotide content have been shown to interfere with mapping/aligning, ultimately leading to fragmented genome assemblies and poor coverage in a phenomenon known as 'GC bias'. The Picard CollectGcBiasMetrics collects information about the relative proportions of G + C in a sample. Results are written to files referenced by these attributes. `gc_bias` references a chart associated with the summary metrics table. The chart is a histogram of normalized coverage, the distribution of windows corresponding to GC percentages and base qualities corresponding to each %GC bin.

7.5.6. {tumor_bam_ | normal_bam_} insert_size_histogram and insert_size_metrics

Picard CollectInsertSizeMetrics provides metrics (in the file referenced by the `insert_size_metrics` attribute) for validating library construction, including insert size distribution and read orientation of paired-end libraries. The metrics file includes insert size statistics. A histogram and cumulative distribution of

insert sizes are plotted in the file reference by the `insert_size_histogram` attribute.

7.5.7. {tumor_bam_ | normal_bam_} quality_by_cycle and quality_by_cycle_metrics

Picard MeanQualityByCycle generates a table (`quality_by_cycle_metrics` attribute) and chart (`quality_by_cycle` attribute) of the mean quality score of a BAM for each cycle. Per cycle mean quality gives a snapshot of sequencing machine performance...spikes in mean quality within reads are unexpected and may be indicative of technical problems during sequencing.

7.5.8. {tumor_bam_ | normal_bam_} quality_by_distribution and quality_by_distribution_metrics

Picard QualityScoreDistribution plots a histgram (`quality_by_distribution` attribute) of base read quality scores (both original and modified through BQSR).

7.5.9. {tumor_bam_ | normal_bam_} quality_by_yield_metrics

Results from the Picard CollectQualityYieldMetrics tool are written to a file referenced by the `quality_by_yield_metrics` attribute. The file contains various metrics about the reads that passed quality thresholds and Illumina-specific filters.

7.5.10. {tumor_bam_ | normal_bam_} converted_oxog_metrics

Results from the Picard ConvertSequencingArtifactToOxoG tool are written to this attribute. The tool extracts 8-oxoguanine (OxoG) artifact metrics from the output of the Picard CollectSequencingArtifactsMetrics, which, in turn, is run as part of our CollectMultipleMetrics task.

7.5.11. {tumor_bam_ | normal_bam_} hybrid_selection_metrics

Results from the Picard CollectHsMetrics tool are written to this attribute. The metrics are specific for sequence datasets generated through hybrid-selection.

8. Retrieve and Review Results

   The results of the characterization workflow fall into three categories: (i) MAF files identifying a tumor sample's somatic SNVs and INDELs, (ii) data files (tsv) and segmented copy number plots identifying regions of somatic amplification and deletion in a tumor cell, and (iii) visualizations, in particular lego plots depicting somatic mutation spectrums in the context of flanking necleotides.

How to Cite this Workflow

Please cite in publications as follows (including individual constituent tools):

```
We have utilized the Getz Lab CGA WES Characterization pipeline
[https://github.com/broadinstitute/CGA_Production_Analysis_Pipeline] developed
```

at the Broad Institute to call, filter and annotate somatic mutations and copy number variation. The pipeline employs the following tools: MuTect[1], ContEst[2], Strelka[3], Orientation Bias Filter[4], DeTiN [5], AllelicCapSeg[6], MAFPoNFilter[7], RealignmentFilter, ABSOLUTE[8], GATK[9], PicardTools[10], Variant Effect Predictor [11], Oncotator [12].

1. **MuTect1:** Cibulskis, K, Lawrence, MS & Carter, SL. Sensitive detection of somatic point mutations in impure and heterogeneous cancer samples. Nature … (2013). doi:10.1038/nbt.2514
2. **ContEst:** Cibulskis, K. et al. ContEst: estimating cross-contamination of human samples in next-generation sequencing data.Bioinformatics 27, 2601–2 (2011).
3. **Strelka:** Saunders, C. T. et al. Strelka: accurate somatic small-variant calling from sequenced tumor–normal sample pairs. Bioinformatics 28, 1811–7 (2012).
4. **Orientation Bias Filter (oxoG, FFPE):** Costello, M. et al. Discovery and characterization of artifactual mutations in deep coverage targeted capture sequencing data due to oxidative DNA damage during sample preparation. Nucleic Acids Res. 41, e67 (2013).
5. **DeTiN:** Taylor-Weiner, A. et al. DeTiN: overcoming tumor-in-normal contamination. Nat Methods 15, 531–534 (2018).
6. **AllelicCapSeg:** Landau, D. A. et al. Evolution and impact of subclonal mutations in chronic lymphocytic leukemia. Cell 152, 714–26 (2013).
7. **MAFPoNFilter:** Lawrence, M. *et al.* Discovery and saturation analysis of cancer genes across 21 tumour types. *Nature* **505**, 495 (2014).
8. **ABSOLUTE:** Carter, S. L. et al. Absolute quantification of somatic DNA alterations in human cancer. Nat. Biotechnol. 30, 413–21 (2012). doi: 10.1038/nbt.2203.
9. **GATK (Mutect2, somatic CNV):** McKenna, A. et al. The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data.Genome Res. 20, 1297–303 (2010).
10. **Picard Tools:** https://software.broadinstitute.org/gatk/documentation/tooldocs/4.0.1.0/picard_fingerprint_CrosscheckFingerprints.php https://software.broadinstitute.org/gatk/documentation/tooldocs/4.0.0.0/picard_analysis_CollectMultipleMetrics.php
11. **Variant Effect Predictor:** McLaren, W. et al. The Ensembl Variant Effect Predictor.Genome Biol. 17, 122 (2016).
12. **Oncotator:** Ramos, A. H. et al. Oncotator: cancer variant annotation tool.Hum. Mutat. 36, E2423–9 (2015).

License

References

1. McKenna, A. *et al.* The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data.*Genome Res.* **20,** 1297–303 (2010).
2. https://software.broadinstitute.org/gatk/documentation/tooldocs/4.0.1.0/picard_fingerprint_CrosscheckFingerprints.php
3. Cibulskis, K. *et al.* ContEst: estimating cross-contamination of human samples in next-generation sequencing data.*Bioinformatics* **27,** 2601–2 (2011).
4. Taylor-Weiner, A. *et al.* DeTiN: overcoming tumor-in-normal contamination. *Nat Methods* **15,** 531–534 (2018).
5. https://software.broadinstitute.org/gatk/documentation/tooldocs/4.0.0.0/picard_analysis_CollectMultipleMetrics.php
6. Cibulskis, K, Lawrence, MS & Carter, SL. Sensitive detection of somatic point mutations in impure and heterogeneous cancer samples. *Nature* … (2013). doi:10.1038/nbt.2514
7. Saunders, C. T. *et al.* Strelka: accurate somatic small-variant calling from sequenced tumor-normal sample pairs. *Bioinformatics***28,** 1811–7 (2012).
8. https://software.broadinstitute.org/gatk/documentation/tooldocs/3.8-0/org_broadinstitute_gatk_tools_walkers_cancer_m2_MuTect2.php
9. Ramos, A. H. *et al.* Oncotator: cancer variant annotation tool.*Hum. Mutat.* **36,** E2423–9 (2015).
10. McLaren, W. *et al.* The Ensembl Variant Effect Predictor.*Genome Biol.* **17,** 122 (2016).
11. Costello, M. *et al.* Discovery and characterization of artifactual mutations in deep coverage targeted capture sequencing data due to oxidative DNA damage during sample preparation.*Nucleic Acids Res.* **41,** e67 (2013).
12. Lawrence, M. *et al.* Discovery and saturation analysis of cancer genes across 21 tumour types. *Nature* **505,** 495 (2014).
13. Kent, W. BLAT—The BLAST-Like Alignment Tool. *Genome Res* **12,** 656–664 (2002).
14. Ellrott, K. *et al.* Scalable Open Science Approach for Mutation Calling of Tumor Exomes Using Multiple Genomic Pipelines.*Cell Syst* **6,** 271–281.e7 (2018).
15. http://www.genomics.broadinstitute.org/call-somatic-copy-number-variants-using-gatk-cnv
16. Landau, D. A. *et al.* Evolution and impact of subclonal mutations in chronic lymphocytic leukemia. *Cell* **152,** 714–26 (2013).
17. Carter, S. L. *et al.* Absolute quantification of somatic DNA alterations in human cancer. *Nat. Biotechnol.* **30,** 413–21 (2012).

Appendix A: Workflow Input Parameters

Workflow-level Inputs

These are inputs that are used by two or more of the workflow's tasks. The majority of these workflow-level inputs are reference data files.

Input variable name	category	type	description	template workspace attribute(s)
`cga_pipeline_config`	Config	File	Configuration json file with optional parameters for the workflow and its constituent tasks. We recommend not changing any of the values in this file.	CGA_PIPELINE_CONFIG
`tumorBam`	Sample data	File	Tumor sample BAM file (see https://samtools.github.io/hts-specs/SAMv1.pdf).	
`normalBam`	Sample data	File	Normal sample BAM file (see https://samtools.github.io/hts-specs/SAMv1.pdf).	
`tumorBamIdx`	Sample data	File	Tumor sample BAI file (BAM indexed) (see samtools index command http://www.htslib.org/doc/samtools.html).	
`normalBamIdx`	Sample data	File	Normal sample BAI file (BAM indexed) (see samtools index command http://www.htslib.org/doc/samtools.html).	
`caseName`	Sample data	String	Name attached to the tumor sample under analysis. This name will be used in the naming of some of the workflow's output files.	
`ctrlName`	Sample data	String	Name attached to the normal sample under analysis. This name will be used in the naming of some of the workflow's output files.	
`pairName`	Sample data	String	Name attached to the tumor-normal pair under analysis. This name will be used in the naming of some of the workflow's output files.	

`readGroupBlackList`	Reference data	File	A file containing blacklisted readgroups. A readgroup is added to the blacklist if the CrossCheckLaneFingerprint task determines that reads in the readgroup do not come from the same individual as the reads in other readgroup's within the sample and tumor bam. We provide a default readGroupBlackList file containing blacklisted readgroups from the TCGA project. Clearly these readgroups will not apply to non-TCGA data, but no harm is done by their inclusion in the default blacklist.	READ_GROUP_BLACKLIST
`refFasta`	Reference data	File	The FASTA file for the appropriate genome build (reference sequence file). This workflow has only been tested and benchmarked with hg19.	REF_FASTA
`refFastaIdx`	Reference data	File	The FASTA file index for the reference genome (see http://www.htslib.org/doc/faidx.html).	REF_FASTA_IDX
`refFastaDict`	Reference data	File	The FASTA file dictionary for the reference genome (see https://broadinstitute.github.io/picard/command-line-overview.html#CreateSequenceDictionary). This file is required by several of the workflow's tasks.	REF_FAST_DICT
`targetIntervals`	Reference data	File	An interval list file that contains the locations of the targets used for the whole exome sequencing. We have placed two target intervals files on Google Cloud Storage: one for use with hg19 WES data generated using the agilent exome capture kit, a second for use with hg19 WES data generated using the Illumina ICE capture kit. If your target intervals file is in BED format, you must convert it to the Picard interval_list format using Picard's BedToInterval tool.	If analyzing data generated using the Agilent exome capture kit: AGILENT_TARGET_INTERVALS If analyzing data generated using the Illumina exome capture kit (ICE): ILLUMINA_TARGET_INTERVALS
`baitIntervals`	Reference data	File	An interval list file that contains the locations of the baits used for the whole exome sequencing. We have placed	If analyzing data generated using the Agilent exome capture kit: AGILENT_BAIT_INTERVALS

			two bait intervals files on Google Cloud Storage: one for use with hg19 WES data generated using the agilent exome capture kit, a second for use with hg19 WES data generated using the Illumina ICE capture kit. If your bait intervals file is in BED format, you must convert it to the Picard interval_list format using Picard's BedToInterval tool.	If analyzing data generated using the Illumina exome capture kit (ICE): ILLUMINA_BAIT_INTERVALS
DB_SNP_VCF	Reference data	File	VCF format dbSNP file, input to the PicardMultipleMetrics, Mutect1 and MutectFC tasks. The PicardMultipleMetrics task uses this VCF to exclude regions around known polymorphisms from analysis by several of the tool's metrics programs. Mutect1 and MutectFC uses this VCF when classifying a detected mutation as somatic or germline. For details, see the discussion of variant classification in https://www.nature.com/articles/nbt.2514.pdf.	DB_SNP_VCF
DB_SNP_VCF_IDX	Reference data	File	Index file of VCF file of DB SNP variants.	DB_SNP_VCF_IDX
cosmicVCF	Reference data	File	COSMIC catalogue of known somatic mutations in VCF format. (See https://cancer.sanger.ac.uk/cosmic/). The workflow's Mutect1 and MutectFC tasks use the cosmic VCF. The COSMIC VCF serves two purposes:(1) Sites that are in both the dbSNP and COSMIC VCFs do *NOT* use the prior as a site being germline during somatic classification. This is because dbSNP contains a number of sites that are common somatic events which were deposited into dbSNP in the past. We want to counteract this effect and not make these sites harder to call (dbSNP sites that are not in COSMIC have their LOD threshold raised). (2) Sites in COSMIC are exempt from Mutect1's and MutectFC's "Panel of Normals" filter.	COSMIC_VCF
normalPanel	Panel of	File	Panel of normals (PoN) in VCF	KGENOMES_MUTECT_PON

	Normals		format used by mutect1's, mutectFC's and mutect2's internal PoN filters. See https://www.nature.com/articles/nbt.2514 for details.) We provide a PoN VCF contructed from the 1000 Genomes germline data set. This is an open access data file.	
`cytoBandFile`	Reference data	File	TSV file of chromsomal annotations (chr, start, end, band, stains) used by the workflow's MAFPoNFilter instances.	CYTO_BAND
`GATK4_JAR`	Toolkit executable	File	GATK4 Jar file	GATK4_JAR
`hasPicardMetrics_tumor*`	Workflow control	Boolean	Indicates whether tumor sample's picard metrics have been calculated ahead of running workflow. Default value is False.	
`hasPicardMetrics_normal*`	Workflow control	Boolean	Indicates whether normal sample's picard metrics have been calculated ahead of running workflow. Default value is False.	
`forceComputePicardMetrics_tumor*`	Workflow control	Boolean	Indicates whether to force calculation of tumor sample's picard metrics, even if they had been calculated outside of workflow. Default value is True. The PicardMultipleMetrics task tends to be one of the longer running tasks in the workflow, thus if the results have been generated outside of the workflow (e.g., the Broad Genomics Platform typically computes these metrics after sequencing and alignment and delivers the results along with the BAM files).	
`forceComputePicardMetrics_normal*`	Workflow control	Boolean	Indicates whether to force calculation of normal sample's picard metrics, even if they have been calculated outside of workflow. Default value is True. The PicardMultipleMetrics task tends to be one of the longer running tasks in the workflow, thus if the results have been generated outside of the workflow (e.g., the Broad	

			Genomics Platform typically computes these metrics after sequencing and alignment and delivers the results along with the BAM files).	
`PONS_list`	Config	File	Lists the instances of the MAFPonFilter task to run (one instance per token pon file) and the GCS URL to the corresponding token pon files.	If you are dbGaP authorized for TCGA controlled access data, you may use the PON list referenced by, MAF_PON_FILTER_LIST This list references a single token pon file constructed from 8334 TCGA normal samples. Note that to use this Token PoN, your workspace MUST be in the TCGA-dbGaP-Authorized authorization domain, regardless of whether your sample data is controlled access. If you are using a controlled access MAF PoN Filter token file, you must treat the resulting output files as controlled access. If you are not dbGaP authorized, or you do not want to place your workspace under the TCGA-dbGaP-Authorized Authorization domain, you CANNOT use the default PON_list file and Token PON File that we make available to TCGA dbGa Authorized users.. You will need to create your own Token PON File(s) and PON_list.

* optional

Task-level Inputs

These are inputs that are used by a single workflow task.

Task and input variable name	Category	type	Description	template workspace attribute(s)
`blat.hg19_bit`	Reference file	File	hg19 reference in the .2bit format (see http://genome.ucsc.edu/FAQ/FAQformat.html#format7)	BLAT_REALIGNER_HG19_BIT
`ContEST_Ta`	Reference file	File	The population allele	HAP_MAP_VCF

`sk.HapMapVCF`			frequencies for each SNP in HapMap	
`ContEST_Task.SNP6Bed`	Reference file	File	BED file containing interval list for SNP6 population sites.	SNP6BED
`CopyNumberReportQC_Task.captureNormalsDBRCLZip`	Reference file	File	DB containing binned read counts for a large collection of normal samples; is used for normalizing read counts in CopyNumberQC	CAP_NORM_DB_ZIP
`CrossCheckLaneFingerprints_Task.HaplotypeDBForCrossCheck`	Reference file	File	Haplotype DB used for fingerprinting.	HAP_DB_FOR_CC
`GatherAndDeTiN_Task.exac_pickle`	Reference file	File	Pickle file (serailized python object structure) containing ExAC sitles with minor allele fraction > 0.01.	EXAC_PICKLE
`gatk_acnv_only.common_snp_list`	Reference file	File	A picard interval-list file indicating common heterozygous sites.	GOOD_HETS
`gatk_acnv_only.gatk_protected`	Toolkit executable	File	GATK3 Jar file	GATK_PROTECTED_JAR
`gatk_acnv_only.PoN`	Panel-of-Normals	File	A panel-of-normals file for use with GATK3 allelic CNV.	ACNV_PON
`lego_plotter_task.mut_categs`	Reference file	File	File defining mutation catgories used by the lego plotter task.	MUT_CATEGORIES
`mutation_validator*.tumorRNABam`	Sample data	File	RNA Bam provided to mutation validator. (See https://www.sciencedirect.com/science/article/pii/S2405471218300966?via%3Dihub for a description of mutation validator tool.)	
`mutation_validator*.tumorRNABamIdx`	Sample data	File	RNA Bam Index file	
`MutectFC_Task.mutectIntervals`	Reference file	File	Interval list defines set of specified loci that will be explicitly searched for variants by the MutectFC task.	MUTECT_FC_INTERVALS
`Oncotate_task.oncoDB`	Reference file	File	Tar ball containing database used by the Oncotate task.	ONCO_TARBALL

`TarBall`				
`Strelka.config`	Configuration File	File	Config file used with the Strelka task	STRELKA_CONFIG
`VEP_Task.VEP_File`	Reference file	File	Zip file containing database used by the VEP (variant effect predictor) task.	VEP_ZIP_FILE

* optional

Appendix B: MAF PoN Filter Token PoN Maker

We have made publicly available a workflow for creating a custom Token PoN for use by the CGA WES Characterizaton Pipeline's MAF PoN Filter task. The workflow is available in the FireCloud Method Repository under the name:
`getzlab/CGA_Token_PoN_Maker_v0.1_Jan2019`

We have also published a companion method config. The method config references two workspace attributes:

```
UCSC_ANNOTATION_DB
TOKEN_PON_MAKER_CONFIG
```

Both of these workspace attributes are defined in the open and controlled access template workspaces.

The method configuration defines pair_set as the root entity type. From each pair in the set it constructs an array of bam files and an array of bai files from the control (i.e., normal) samples within that pair. You can also create a method configuration that defines as its root entity type a sample set, and construct the bam and bai arrays from all of the samples in that set.

Note that in order to employ an instance of the MAF PoN Filter that uses a custom Token PoN created with this workflow, you will need to provide a MAF PoN Filter configuration file that references you custom Token PoN. See Configuring MAF PoN Filter Instance for details.