PureCN - Quick Start

This tutorial provides a quick overview of the command line tools shipping with *PureCN*. For the R package and more detailed information, see the main vignette.

Update from previous stable versions

PureCN is fully backward compatible with input generated by versions 1.10 and 1.12. *PureCN* 1.10 introduced a new normal database format with several important improvements such as not splitting the database by sample sex for the normalization of autosomes. It is therefore necessary to at least re-run the NormalDB.R script described below when upgrading from version 1.8.

For upgrades from version 1.6, we highly recommend starting from scratch following this tutorial.

Installation

For the command line scripts described in this tutorial, we will need to install PureCN with suggested dependencies:

```
if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install("PureCN", dependencies = TRUE)
```

Alternatively, manually install the packages required by the command line scripts:

```
BiocManager::install(c("PureCN", "optparse",
    "TxDb.Hsapiens.UCSC.hg19.knownGene", "org.Hs.eg.db"))
```

(Replace `hg19` with your genome version).

Prepare environment and files

- Start R and enter the following to get the path to the command line scripts:

  ```
  system.file("extdata", package="PureCN")

  ## [1] "/tmp/Rtmp4DSmiG/Rinst5a0910ce8523/PureCN/extdata"
  ```

- Exit R and store this path in an environment variable, for example in BASH:

  ```
  $ export PURECN="/path/to/PureCN/extdata"
  $ Rscript $PURECN/PureCN.R --help
  Usage: /path/to/PureCN/inst/extdata/PureCN.R [options] ...
  ```

- Generate an interval file from a BED file containing baits coordinates:

  ```
  # specify path where PureCN should store reference files
  $ export OUT_REF="reference_files"
  $ Rscript $PURECN/IntervalFile.R --infile baits_hg19.bed \
      --fasta hg19.fa --outfile $OUT_REF/baits_hg19_intervals.txt \
  ```

```
--offtarget --genome hg19 \
--export $OUT_REF/baits_optimized_hg19.bed \
--mappability wgEncodeCrgMapabilityAlign100mer.bigWig \
--reptiming wgEncodeUwRepliSeqK562WaveSignalRep1.bigWig
```

Internally, this script uses *rtracklayer* to parse the `infile`. Make sure that the file format matches the file extension. See the *rtracklayer* documentation for problems loading the file. Check that the genome version of the baits file matches the reference. Do not include chrM baits in case the capture kit includes some.

The `-offtarget` flag will include off-target reads. Including them is recommended except for Amplicon data.

The `-genome` version is needed to annotate exons with gene symbols. Use hg19/hg38 for human genomes, not b37/b38. You might get a warning that an annotation package is missing. For hg19, install *TxDb.Hsapiens.UCSC.hg19.knownGene* in R.

The `-export` argument is optional. If provided, this script will store the modified intervals as BED file for example (again every *rtracklayer* format is supported). This is useful when the coverages are calculated with third-party tools like GATK.

The `-mappability` argument should provide a *rtracklayer* parsable file with a mappability score in the first meta data column. If provided, off-target regions will be restricted to regions specified in this file. On-target regions with low mappability will be excluded. For hg19, download the file from the UCSC website. Choose the kmer size that best fits your average mapped read length. For hg38, download recommended 76-kmer or 100-kmer mappability files through the courtesy of the Waldron lab from:

- https://s3.amazonaws.com/purecn/GCA_000001405.15_GRCh38_no_alt_analysis_set_76.bw

- https://s3.amazonaws.com/purecn/GCA_000001405.15_GRCh38_no_alt_analysis_set_100.bw

See the FAQ section of the main vignette for instruction how to generate such a file for other references.

Similarly, the `-reptiming` argument takes a replication timing score in the same format. If provided, GC-normalized and log-transformed coverage is tested for a linear relationship with this score and normalized accordingly. This is optional and provides only a minor benefit for coverage normalization, but can identify samples with high proliferation. Requires `-offtarget` to be useful.

Generation of this interval file is not needed when CNVkit is used to generate coverage data (covered at the end of this tutorial).

Create VCF files

PureCN does not ship with a variant caller. Use a third-party tool to generate a VCF for each sample.

Important recommendations:

- Use *MuTect* 1.1.7 if possible

- Support for *MuTect 2* and *FreeBayes* is available for tumor-only VCFs, but currently poorly tested and only very limited artifact filtering will be performed for these callers. See the FAQ section in the main vignette for common problems and questions related to input data.

- Since germline SNPs are needed to infer allele-specific copy numbers, the provided VCF needs to contain both somatic and germline variants. Make sure that upstream filtering does not remove high quality SNPs, in particular due to presence in germline databases.

- Run the variant caller with a 50-75 base pair interval padding to increase the number of heterozygous SNPs

Run PureCN with internal segmentation

The following describes *PureCN* runs with internal copy number normalization and segmentation.

Coverage

For each sample, tumor and normal, calculate GC-normalized coverages:

```
# Calculate and GC-normalize coverage from a BAM file
$ Rscript $PURECN/Coverage.R --outdir $OUT/$SAMPLEID \
    --bam ${SAMPLEID}.bam \
    --intervals $OUT_REF/baits_hg19_intervals.txt


# GC-normalize coverage from a GATK DepthOfCoverage file
Rscript $PURECN/Coverage.R --outdir $OUT/$SAMPLEID \
    --coverage ${SAMPLEID}.coverage.sample_interval_summary \
    --intervals $OUT_REF/baits_hg19_intervals.txt
```

Similar to GATK, this script also takes a text file containing a list of BAM or coverage file names (one per line). The file extension must be .list:

```
# Calculate and GC-normalize coverage from a list of BAM files
$ Rscript $PURECN/Coverage.R --outdir $OUT \
    --bam normals.list \
    --intervals $OUT_REF/baits_hg19_intervals.txt \
    --cpu 4
```

Important recommendations:

- Only provide `-keepduplicates` or `-removemapq0` if you know what you are doing and always use the same command line arguments for tumor and the normals

NormalDB

To build a normal database for coverage normalization, copy the paths to all GC-normalized normal coverage files in a single text file, line-by-line:

```
ls -a normal*loess.txt | cat > example_normal.list

# From already GC-normalized files
$ Rscript $PURECN/NormalDB.R --outdir $OUT_REF \
    --coveragefiles example_normal.list \
    --genome hg19 --assay agilent_v6

# When normal panel VCF is available (highly recommended for unmatched samples)
$ Rscript $PURECN/NormalDB.R --outdir $OUT_REF \
    --coveragefiles example_normal.list \
    --genome hg19 --normal_panel $NORMAL_PANEL --assay agilent_v6
```

Important recommendations:

- Consider generating different databases when differences are significant, e.g. for samples with different read lengths or insert size distributions

- In particular, do not mix normal data obtained with different capture kits (e.g. Agilent SureSelect v4 and v6)

- Provide a normal panel VCF here to precompute mapping bias for faster runtimes. The only requirement for the VCF is an `AD` format field containing the number of reference and alt reads for all samples. See the example file `$PURECN/normalpanel.vcf.gz`.

- For ideal results, examine the interval_weights.png file to find good off-target bin widths. You will need to re-run IntervalFile.R with the `-offtargetwidth` parameter and re-calculate the coverages.

- The `-assay` argument is optional and is only used to add the provided assay name to all output files

- A warning pointing to the likely use of a wrong baits file means that more than 5% of targets have close to 0 coverage in all normal samples. A BED file with the low coverage targets will be generated in `-outdir`. If for any reason there is no access to the correct file, it is recommended to re-run the IntervalFile.R command and provide this BED file with `-exclude`.

PureCN

Now that the assay-specific files are created and all coverages calculated, we run PureCN.R to normalize, segment and determine purity and ploidy:

```
mkdir $OUT/$SAMPLEID

# Without a matched normal (minimal test run)
$ Rscript $PURECN/PureCN.R --out $OUT/$SAMPLEID \
    --tumor $OUT/$SAMPLEID/${SAMPLEID}_coverage_loess.txt \
    --sampleid $SAMPLEID \
    --vcf ${SAMPLEID}_mutect.vcf \
    --normaldb $OUT_REF/normalDB_hg19.rds \
    --intervals $OUT_REF/baits_hg19_intervals.txt \
    --genome hg19

# Production pipeline run
$ Rscript $PURECN/PureCN.R --out $OUT/$SAMPLEID \
```

```
    --tumor $OUT/$SAMPLEID/${SAMPLEID}_coverage_loess.txt \
    --sampleid $SAMPLEID \
    --vcf ${SAMPLEID}_mutect.vcf \
    --statsfile ${SAMPLEID}_mutect_stats.txt \
    --normaldb $OUT_REF/normalDB_hg19.rds \
    --normal_panel $OUT_REF/mapping_bias_hg19.rds \
    --intervals $OUT_REF/baits_hg19_intervals.txt \
    --intervalweightfile $OUT_REF/interval_weights_hg19.txt \
    --snpblacklist hg19_simpleRepeats.bed \
    --genome hg19 \
    --force --postoptimize --seed 123

# With a matched normal (test run; for production pipelines we recommend the
# unmatched workflow described above)
$ Rscript $PURECN/PureCN.R --out $OUT/$SAMPLEID \
    --tumor $OUT/$SAMPLEID/${SAMPLEID}_coverage_loess.txt \
    --normal $OUT/$SAMPLEID/${SAMPLEID_NORMAL}_coverage_loess.txt \
    --sampleid $SAMPLEID \
    --vcf ${SAMPLEID}_mutect.vcf \
    --normaldb $OUT_REF/normalDB_hg19.rds \
    --intervals $OUT_REF/baits_hg19_intervals.txt \
    --genome hg19

# Recreate output after manual curation of ${SAMPLEID}.csv
$ Rscript $PURECN/PureCN.R --rds $OUT/$SAMPLEID/${SAMPLEID}.rds
```

Important recommendations:

- Even if matched normals are available, it is often better to use the normal database for coverage normalization. **When a matched normal coverage is provided with** `-normal` **then the pool of normal coverage normalization and denoising steps are skipped!**

- Always provide the normal coverage database to ignore low quality regions in the segmentation and to increase the sensitivity for homozygous deletions in high purity samples.

- The normal panel VCF file is useful for mapping bias correction and especially recommended without matched normals. See the FAQ of the main vignette how to generate this file. It is not essential for test runs.

- The *MuTect* 1.1.7 stats file (the main output file besides the VCF) should be provided for better artifact filtering. If the VCF was generated by a pipeline that performs good artifact filtering, this file is not needed.

- The `-postoptimize` flag defines that purity should be optimized using both variant allelic fractions and copy number instead of copy number only. This results in a significant runtime increase for whole-exome data.

- If `-out` is a directory, it will use the sample id as file prefix for all output files. Otherwise *PureCN* will use `-out` as prefix.

- The `-parallel` flag will enable the parallel fitting of local optima. See *BiocParallel* for details. This script will use the default backend.

- Defaults are well calibrated and should produce close to ideal results for most samples. A few common cases where changing defaults makes sense:

  High purity and high quality: For cancer types with a high expected purity, such as Ovarian cancer, AND when quality is expected to be very good (high coverage, young samples), `-maxcopynumber 8`

  Small panels with high coverage: `-padding 100` (or higher), requires running the variant caller with this padding or without interval file. Use the same settings for the panel of normals VCF so that SNPs in the flanking regions have reliable mapping bias estimates. The `-maxhomozygousloss` parameter might also need some adjustment for very small panels with large gaps around captured deletions.

  Cell lines: Safely skip the search for low purity solutions in cell lines: `-maxcopynumber 8`, `-minpurity 0.9`, `-maxpurity 0.99`. Add `-modelhomozygous` to find regions of LOH in samples without normal contamination (do not provide this flag when matched normal data are available in the VCF).

  cfDNA: `-minpurity 0.1`, `-minaf 0.01` (or lower) and `-error 0.0005` (or lower, when there is UMI-based error correction). Note that the estimated purity can be very wrong when the true purity is below 5-7%; these samples are usually flagged as non-aberrant.

  Amplicon data: `-model betabin` (Amplicon data is not officially supported)

  All assays: `-maxsegments` should set to a value so that with few exceptions only poor quality samples exceed this cutoff. For cancer types with high heterogenity, it is also recommended to increase `-maxnonclonal` to 0.3-0.4 (this will increase the runtime significantly for whole-exome data).

Run PureCN with third-party segmentation

If you already have a segmentation from third-party tools (for example CNVkit, EXCAVATOR2). For a test run:

```
Rscript $PURECN/PureCN.R --out $OUT/$SAMPLEID  \
    --sampleid $SAMPLEID \
    --segfile $OUT/$SAMPLEID/${SAMPLEID}.cnvkit.seg \
    --vcf ${SAMPLEID}_mutect.vcf \
    --intervals $OUT_REF/baits_hg19_intervals.txt \
    --genome hg19
```

See the main vignette for more details and file formats.

For a production pipeline run we provide again more information about the assay and genome. Here an CNVkit example (CNVkit runs without normal reference samples are not recommended):

```
# Provide a normal panel VCF to remove mapping biases, pre-compute
# position-specific bias for much faster runtimes with large panels
# This needs to be done only once for each assay
Rscript $PURECN/NormalDB.R --outdir $OUT_REF --normal_panel $NORMAL_PANEL \
    --assay agilent_v6 --genome hg19 --force


# Export the segmentation in DNAcopy format
```

```
cnvkit.py export seg $OUT/$SAMPLEID/${SAMPLEID}_cnvkit.cns --enumerate-chroms \
    -o $OUT/$SAMPLEID/${SAMPLEID}_cnvkit.seg

# Run PureCN by providing the *.cnr and *.seg files
Rscript $PURECN/PureCN.R --out $OUT/$SAMPLEID  \
    --sampleid $SAMPLEID \
    --tumor $OUT/$SAMPLEID/${SAMPLEID}_cnvkit.cnr \
    --segfile $OUT/$SAMPLEID/${SAMPLEID}_cnvkit.seg \
    --normal_panel $OUT_REF/mapping_bias_agilent_v6_hg19.rds \
    --vcf ${SAMPLEID}_mutect.vcf \
    --statsfile ${SAMPLEID}_mutect_stats.txt \
    --snpblacklist hg19_simpleRepeats.bed \
    --genome hg19 \
    --funsegmentation none \
    --force --postoptimize --seed 123
```

Important recommendations:

- The `-funsegmentation` argument controls if the data should to be re-segmented using germline BAFs (default). Set this value to `none` if the provided segmentation should be used as is.

- Since CNVkit provides all necessary information in the *.cnr output files, the `-inter vals` argument is not required.

Dx

Dx.R extracts copy number and mutation metrics from PureCN.R output.

```
# Provide a BED file with callable regions, for examples obtained by
# GATK CallableLoci. Useful to calculate mutations per megabase and
# to exclude low quality regions.
grep CALLABLE ${SAMPLEID}_callable_status.bed > \
    ${SAMPLEID}_callable_status_filtered.bed

# Only count mutations in callable regions, also subtract what was ignored
# in PureCN.R via --snpblacklist, like simple repeats, from the mutation per
# megabase calculation
# Also search for the COSMIC mutation signatures
# (http://cancer.sanger.ac.uk/cosmic/signatures)
Rscript $PureCN/Dx.R --out $OUT/$SAMPLEID/$SAMPLEID \
    --rds $OUT/SAMPLEID/${SAMPLEID}.rds \
    --callable ${SAMPLEID}_callable_status_filtered.bed \
    --exclude hg19_simpleRepeats.bed \
    --signatures

# Restrict mutation burden calculation to coding sequences
Rscript $PureCN/FilterCallableLoci.R --genome hg19 \
    --infile ${SAMPLEID}_callable_status_filtered.bed \
    --outfile ${SAMPLEID}_callable_status_filtered_cds.bed \
    --exclude '^HLA'
```

```
Rscript $PureCN/Dx.R --out $OUT/$SAMPLEID/${SAMPLEID}_cds \
    --rds $OUT/SAMPLEID/${SAMPLEID}.rds \
    --callable ${SAMPLEID}_callable_status_filtered_cds.bed \
    --exclude hg19_simpleRepeats.bed
```

Important recommendations:

- Run GATK CallableLoci with `-minDepth N` where N is roughly 20% of the mean target coverage of all samples.

Reference

Table 1: IntervalFile

Argument name	Corresponding PureCN argument	PureCN function
–fasta	reference.file	`preprocessIntervals`
–infile	interval.file	`preprocessIntervals`
–offtarget	off.target	`preprocessIntervals`
–targetwidth	average.target.width	`preprocessIntervals`
–mintargetwidth	min.target.width	`preprocessIntervals`
–smalltargets	small.targets	`preprocessIntervals`
–offtargetwidth	average.off.target.width	`preprocessIntervals`
–offtargetseqlevels	off.target.seqlevels	`preprocessIntervals`
–mappability	mappability	`preprocessIntervals`
–minmappability	min.mappability	`preprocessIntervals`
–reptiming	reptiming	`preprocessIntervals`
–reptimingwidth	average.reptiming.width	`preprocessIntervals`
–genome	txdb, org	`annotateTargets`
–outfile		
–export		`rtracklayer::export`
–version -v		
–force -f		
–help -h		

Table 2: Coverage

Argument name	Corresponding PureCN argument	PureCN function
–bam	bam.file	`calculateBamCoverageByInterval`
–bai	index.file	`calculateBamCoverageByInterval`
–coverage	coverage.file	`correctCoverageBias`
–intervals	interval.file	`correctCoverageBias`
–method	method	`correctCoverageBias`
–keepduplicates	keep.duplicates	`calculateBamCoverageByInterval`
–removemapq0	mapqFilter	`ScanBamParam`
–outdir		
–cpu		Number of CPUs to use when multiple BAMs are provided (deprecated)
–cores		Number of CPUs to use when multiple BAMs are provided
–parallel		Use default BiocParallel backend when multiple BAMs are provided
–seed		
–version -v		
–force -f		
–help -h		

Table 3: NormalDB

Argument name	Corresponding PureCN argument	PureCN function
–coveragefiles	normal.coverage.files	`createNormalDatabase`
–normal_panel	normal.panel.vcf.file	`calculateMappingBiasVcf`
–maxmeancoverage	max.mean.coverage	`createNormalDatabase`
–assay -a	Optional assay name	Used in output file names.
–genome -g	Optional genome version	Used in output file names.
–outdir -o		
–version -v		
–force -f		
–help -h		

Table 4: PureCN

Argument name	Corresponding PureCN argument	PureCN function
–sampleid -i	sampleid	`runAbsoluteCN`
–normal	normal.coverage.file	`runAbsoluteCN`
–tumor	tumor.coverage.file	`runAbsoluteCN`
–vcf	vcf.file	`runAbsoluteCN`
–rds	file.rds	`readCurationFile`
–normal_panel	normal.panel.vcf.file	`setMappingBiasVcf`
–normaldb	normalDB (serialized with `saveRDS`)	`calculateTangentNormal`, `filterTargets`
–segfile	seg.file	`runAbsoluteCN`
–sex	sex	`runAbsoluteCN`
–genome	genome	`runAbsoluteCN`
–intervals	interval.file	`runAbsoluteCN`
–statsfile	stats.file	`filterVcfMuTect`
–minaf	af.range	`filterVcfBasic`
–snpblacklist	snp.blacklist	`filterVcfBasic`
–error	error	`runAbsoluteCN`
–dbinfoflag	DB.info.flag	`runAbsoluteCN`
–popafinfofield	POPAF.info.field	`runAbsoluteCN`
–mincosmiccnt	min.cosmic.cnt	`setPriorVcf`
–funsegmentation	fun.segmentation	`runAbsoluteCN`
–alpha	alpha	`segmentationCBS`
–undosd	undo.SD	`segmentationCBS`
–maxsegments	max.segments	`runAbsoluteCN`
–intervalweightfile	interval.weight.file	`segmentationCBS`
–minpurity	test.purity	`runAbsoluteCN`
–maxpurity	test.purity	`runAbsoluteCN`
–minploidy	min.ploidy	`runAbsoluteCN`
–maxploidy	max.ploidy	`runAbsoluteCN`
–maxcopynumber	test.num.copy	`runAbsoluteCN`
–postoptimize	post.optimize	`runAbsoluteCN`
–bootstrapn	n	`bootstrapResults`
–modelhomozygous	model.homozygous	`runAbsoluteCN`
–model	model	`runAbsoluteCN`
–logratiocalibration	log.ratio.calibration	`runAbsoluteCN`
–maxnonclonal	max.non.clonal	`runAbsoluteCN`
–maxhomozygousloss	max.homozygous.loss	`runAbsoluteCN`
–outvcf	return.vcf	`predictSomatic`
–out -o		
–parallel	BPPARAM	`runAbsoluteCN`
–cores	BPPARAM	`runAbsoluteCN`
–seed		
–version -v		
–force -f		
–help -h		

Table 5: Dx

Argument name	Corresponding PureCN argument	PureCN function
–rds	file.rds	`readCurationFile`
–callable	callable	`callMutationBurden`
–exclude	exclude	`callMutationBurden`
–maxpriorsomatic	max.prior.somatic	`callMutationBurden`
–signatures		`deconstructSigs::whichSignatures`
–out		
–version -v		
–force -f		
–help -h		

Session Info

- R version 3.6.1 (2019-07-05), `x86_64-pc-linux-gnu`

- Locale: `LC_CTYPE=en_US.UTF-8`, `LC_NUMERIC=C`, `LC_TIME=en_US.UTF-8`, `LC_COLLATE=C`, `LC_MONETARY=en_US.UTF-8`, `LC_MESSAGES=en_US.UTF-8`, `LC_PAPER=en_US.UTF-8`, `LC_NAME=C`, `LC_ADDRESS=C`, `LC_TELEPHONE=C`, `LC_MEASUREMENT=en_US.UTF-8`, `LC_IDENTIFICATION=C`

- Running under: `Ubuntu 18.04.3 LTS`

- Matrix products: default

- BLAS: `/home/biocbuild/bbs-3.9-bioc/R/lib/libRblas.so`

- LAPACK: `/home/biocbuild/bbs-3.9-bioc/R/lib/libRlapack.so`

- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, stats4, utils

- Other packages: Biobase 2.44.0, BiocGenerics 0.30.0, BiocParallel 1.18.1, Biostrings 2.52.0, DNAcopy 1.58.0, DelayedArray 0.10.0, GenomeInfoDb 1.20.0, GenomicRanges 1.36.0, IRanges 2.18.2, PureCN 1.14.3, Rsamtools 2.0.0, S4Vectors 0.22.0, SummarizedExperiment 1.14.1, VariantAnnotation 1.30.1, XVector 0.24.0, matrixStats 0.54.0

- Loaded via a namespace (and not attached): AnnotationDbi 1.46.1, BSgenome 1.52.0, BiocManager 1.30.4, BiocStyle 2.12.0, DBI 1.0.0, GenomeInfoDbData 1.2.1, GenomicAlignments 1.20.1, GenomicFeatures 1.36.4, Matrix 1.2-17, R6 2.4.0, RColorBrewer 1.1-2, RCurl 1.95-4.12, RSQLite 2.1.2, Rcpp 1.0.2, Rhdf5lib 1.6.0, VGAM 1.1-1, XML 3.98-1.20, assertthat 0.2.1, backports 1.1.4, biomaRt 2.40.4, bit 1.1-14, bit64 0.9-7, bitops 1.0-6, blob 1.2.0, colorspace 1.4-1, compiler 3.6.1, copynumber 1.24.0, crayon 1.3.4, data.table 1.12.2, digest 0.6.20, dplyr 0.8.3, evaluate 0.14, formatR 1.7, futile.logger 1.4.3, futile.options 1.0.1, ggplot2 3.2.1, glue 1.3.1, grid 3.6.1, gridExtra 2.3, gtable 0.3.0, highr 0.8, hms 0.5.1, htmltools 0.3.6, httr 1.4.1, knitr 1.24, labeling 0.3, lambda.r 1.2.3, lattice 0.20-38, lazyeval 0.2.2, magrittr 1.5, memoise 1.1.0, munsell 0.5.0, pillar 1.4.2, pkgconfig 2.0.2, plyr 1.8.4, prettyunits 1.0.2, progress 1.2.2, purrr 0.3.2, reshape2 1.4.3, rhdf5 2.28.0, rlang 0.4.0, rmarkdown 1.15, rtracklayer 1.44.3, scales 1.0.0, splines 3.6.1, stringi 1.4.3, stringr 1.4.0, tibble 2.1.3, tidyselect 0.2.5, tools 3.6.1, vctrs 0.2.0, xfun 0.9, yaml 2.2.0, zeallot 0.1.0, zlibbioc 1.30.0