PureCN: Estimating tumor purity, ploidy, LOH and SNV status using hybrid capture NGS data

Markus Riester

July 7, 2016

Contents

1 Background

PureCN is a purity and ploidy aware copy number caller for cancer samples inspired by the *ABSOLUTE* algorithm [1]. It was designed for hybrid capture sequencing data, especially with medium-sized targeted gene panels without matching normal samples in mind.

It can be used to enhance existing segmentation and normalization algorithms. If the correct purity and ploidy solution was identified, *PureCN* can also help in classifying variants as germline vs. somatic or clonal vs. sub-clonal.

PureCN was further designed to integrate well with industry standard pipelines [2], but it is straightforward to generate input data from other pipelines.

2 Limitations

PureCN currently assumes a completely diploid normal genome. It tries to detect the sex of samples by calculating the coverage ratio of chromosomes X and Y and will then remove any non-diploid chromosomes (i.e. XY for males and Y for females)[1]. Non-diploid chromosomes provide only limited information for purity and ploidy selection, but are of interest for copy number calling. Future versions might support full copy number calling on the XY chromosomes.

PureCN currently also assumes a mostly clonal genome. For most clinical samples, this is reasonable, but very heterogenous samples are likely not possible to call without manual curation. Due to the lack of signal, manual curation is also often necessary in low purity samples or very quiet genomes.

In the absence of matched normals, the software currently requires some normal contamination to infer germline genotypes. Since cell lines are rarely matched, *PureCN* will likely not work well with cell line data.

3 Basic input files

The algorithm first needs to calculate copy number log-ratios of tumor vs. normal control. The for this calculation required coverage data needs to be provided in *GATK DepthOfCoverage* format:

```
Target  total_coverage  average_coverage    Sample1_total_cvg   Sample1_mean_cvg
Sample1_granular_Q1 Sample1_granular_median Sample1_granular_Q3 Sample1_._above_15
chr1:69091-70009    0   0   0   0   1   1   1   0
chr1:367659-368598  6358    9.25121153819759    6358    9.25121153819759    1   7   13  11.6
chr1:621096-622035  6294    9.16910019318401    6294    9.16910019318401    1   7   12  9.5
```

PureCN will only use data from the columns "Target", "total_coverage", and "average_coverage", all other columns are optional. If *GATK* is not available, then we refer to the *ExomeCNV* [3] package and documentation, which provides scripts for generating coverage files from BAM files.

Germline SNPs and somatic mutations are expected in a single VCF file. This VCF should contain a DB info flag for dbSNP membership. VCF files generated by *MuTect* [4] should work well and in general require no post-processing. *PureCN* can handle *MuTect* VCF files generated in both single and matched normal mode. If a matched normal is available, then somatic status information is currently expected in a SOMATIC info flag in the VCF.

4 Example data

We now load a few example files:

```r
gatk.normal.file <- system.file("extdata", "example_normal.txt",
    package="PureCN")
gatk.normal2.file <- system.file("extdata", "example_normal2.txt",
    package="PureCN")
gatk.normal.files <- c(gatk.normal.file, gatk.normal2.file)
gatk.tumor.file <- system.file("extdata", "example_tumor.txt",
    package="PureCN")
vcf.file <- system.file("extdata", "example_vcf.vcf", package="PureCN")
```

To obtain gene level copy number calls, we need an exon-to-gene mapping file. This is provided together with GC content via the gc.gene.file argument. The expected format:

[1] Loss of Y chromosome (LOY) can result in wrong female calls, especially in high purity samples or if LOY is in both tumor and contaminating normal cells. Since homozygous germline SNPs on the X chromosome are removed, this will not affect purity/ploidy selection. A future version might warn users in this case when germline SNPs are provided.'

```
    Target     gc_bias     Gene
    chr1:69091-70009     0.427638737758433     OR4F5
    chr1:367659-368598     0.459574468085106     OR4F29
    chr1:621096-622035     0.459574468085106     OR4F3
```

If *GATK* is available, then the *GCContentByInterval* tool can be used to generate the GC bias column. Otherwise, several *Bioconductor* annotation packages provide GC content information.

```
gc.gene.file <- system.file("extdata", "example_gc.gene.file.txt",
    package="PureCN")
```

5 GC bias

The algorithm works best when the coverage data is GC normalized. The steps below will not GC normalize, since the example data is already normalized. The function `correctCoverageBias`, which borrows normalization code from the *TitanCNA* package [5], can be used to correct for GC bias. We recommend correcting for GC bias, storing the corrected coverage files and then do the following steps on GC corrected data. All other assay-specific biases such as mappability or exon length are mitigated by using a control sample (GC bias is commonly library-specific).

```
gatk.normal.file <- system.file("extdata", "example_normal.txt",
    package = "PureCN")
gc.gene.file <- system.file("extdata", "example_gc.gene.file.txt",
    package = "PureCN")
coverage <- correctCoverageBias(gatk.normal.file, gc.gene.file, "example_normal_loess.txt")
```

6 Pool of Normals

6.1 Selection of normals for log-ratio calculation

For calculating copy number log-ratios of tumor vs. normal, *PureCN* requires coverage from a process-matched normal sample. Using a normal that was sequenced using a similar, but not identical assay, rarely works, since differently covered genomic regions result in too many log-ratio outliers. This section describes how to identify a good process-matched normal in case no matched normal is available or in case the matched normal has low or uneven coverage.

The `createNormalDatabase` function builds a database of coverage files:

```
normalDB <- createNormalDatabase(gatk.normal.files)

## Allosome coverage appears to be missing, cannot determine sex.
## Allosome coverage appears to be missing, cannot determine sex.
```

Internally, this function determines the sex of the samples and trains a PCA that is later used for clustering a tumor file with all normal samples in the database. This clustering is performed by the `findBestNormal` function:

```
# get the best normal
gatk.best.normal.file <- findBestNormal(gatk.tumor.file, normalDB)
```

This function can also return multiple normal files that can be averaged into a single pool:

```
# get the best 2 normals and average them
gatk.best.normal.files <- findBestNormal(gatk.tumor.file, normalDB,
    num.normals=2)
pool <- poolCoverage(lapply(gatk.best.normal.files, readCoverageGatk),
     remove.chrs=c('chrX', 'chrY'))
```

Pooling is only recommended when the coverage in normals is significantly lower than in tumor. Otherwise the PCA will typically do a good job in selecting a normal with decent coverage and similar biases compared to tumor. But it is worth experimenting with different strategies. When pooling, it might be also necessary to remove certain principal components, most likely the first one, to avoid that coverage is the only selection criteria. The `plotBestNormal` function might be helpful in finding a good strategy. Note that this example removes sex chromosomes; if the normal database contains a sufficient number of samples with matching sex, `findBestNormal` will return only normal samples with matching sex.

6.2 Artifact filtering

It is important to remove as many artifacts as possible, since low ploidy solutions are typically punished more by artifacts than high ploidy solutions. High ploidy solutions are complex and usually find ways of explaining artifacts reasonably well. A large selection of normal files is helpful for the identification and removal of artifacts. We can for example use the normal coverage data to estimate the expected variance in coverage per exon. This will help the segmentation algorithm to filter noise. This step is optional, but recommended with the default segmentation function and large pool of normals.

```
exon.weight.file <- "exon_weights.txt"
createExonWeightFile(gatk.tumor.file, gatk.normal.files, exon.weight.file)
```

This function calculates exon-level copy number log-ratios for all normal samples provided in the `gatk.normal.files` argument. Assuming that all normal samples are in general diploid, an unusual high variance in log-ratio is indicative of an exon with either common germline alterations or frequent artifacts; high or low copy number log-ratios in those exons are unlikely measuring somatic copy number events.

For the log-ratio calculation, we provide a coverage file that is used as tumor in the log-ratio calculation. The corresponding `gatk.tumor.file` argument can also be an array of coverage files, in which case the exon coverage variance is averaged over all provided tumor files. As long as the coverage of the tumor files is even, only 1 or 2 tumor files are necessary, however. It is also safe to use a high coverage normal file instead of a tumor file in this step:

```
#createExonWeightFile(gatk.normal.files[1], gatk.normal.files[-1],
#    "exon_weights.txt")
```

We can also use the pool of normals to find SNPs with biased allelic fractions in low quality regions (very significantly different from 0.5 for heterozygous SNPs). We do not do this here for simplicity. This step is recommended for unpaired samples and optional for paired samples. If a matching normal is provided, most variants in low quality regions with biased allelic fractions are automatically ignored. Note that this is not meant to model non-reference bias leading in expected allelic ratio only slightly below 0.5 (typically around 0.48). This bias is typically not strong enough to influence selection of SNV states.

```
#mutect.normal.files <- dir("poolofnormals", pattern="vcf£", full.names=TRUE)
#snp.blacklist <- createSNPBlacklist(mutect.normal.files)
#write.csv(snp.bl[[1]], file="SNP_blacklist.csv")
#write.csv(snp.bl[[2]], file="SNP_blacklist_segmented.csv", row.names=FALSE,
#    quote=FALSE)
```

6.3 Artifact filtering without a pool of normals

By default, *PureCN* will exclude exons with coverage below 15X from segmentation. For SNVs, the same 15X cutoff is applied. *MuTect* applies more sophisticated artifact tests and flags suspicious variants. If *MuTect* was run in matched normal mode, then both potential artifacts and germline variants are rejected, that means we cannot just filter by the PASS/REJECT *MuTect* flags. The `filterVcfMuTect` function optionally reads the *MuTect* stats file and will keep germline variants, while removing potential artifacts. Without the stats file, *PureCN* will use only the filters based on read depths as defined in `filterVcfBasic`. Both functions are automatically called by *PureCN*, but can be easily modified and replaced if necessary.

7 Recommended run

Finally, we can run *PureCN* with all that information:

```
ret <-runAbsoluteCN(gatk.normal.file=pool, gatk.tumor.file=gatk.tumor.file,
    vcf.file=vcf.file, sampleid='Sample1', gc.gene.file=gc.gene.file,
#args.filterVcf=list(snp.blacklist=snp.blacklist, stats.file=mutect.stats.file),
    args.segmentation=list(exon.weight.file=exon.weight.file),
    post.optimize=FALSE, plot.cnv=FALSE)

## Loading GATK coverage files...

## Sex of sample:  ?

## Removing 7 small exons.

## Removing 15 low/high GC exons.

## Loading VCF...

## Assuming LIB-02240e4 is tumor in VCF file.

## Found 2331 variants in VCF file.

## Removing 0 non heterozygous (in matched normal) germline SNPs.

## Removing 62 SNPs with AF < 0.03 or AF >= 0.97 or less than 3 supporting reads or depth < 15.

## Found SOMATIC annotation in VCF. Setting somatic prior probabilities for somatic variants to 0.999
or to 1e-04 otherwise.

## Segmenting data...

## Exon weights found, will use weighted CBS.

## Removing 171 low coverage exons.

## Analyzing: Sample1
## Call:
## segment(x = smoothed.CNA.obj, weights = weights, alpha = alpha,
##     undo.splits = undo.splits, undo.SD = sdundo, verbose = ifelse(verbose,
##         1, 0))
##
##          ID chrom   loc.start   loc.end num.mark seg.mean
## 1  Sample1     1   1216044.5 248722319      943   0.3430
## 2  Sample1     2   1638036.0 231775198      717  -0.2557
## 3  Sample1     2 236403412.5 241737117       94   0.2467
## 4  Sample1     3  11832017.5 149470198      438   0.3071
## 5  Sample1     3 150264604.0 151542537       18   1.5703
## 6  Sample1     3 151545662.5 195938114       82   0.3166
## 7  Sample1     4    843512.0  70146580      134   0.2981
## 8  Sample1     4  75673305.5  77700146       39  -0.1927
## 9  Sample1     4  81188156.5 108831608       44  -1.1808
## 10 Sample1     4 110635592.5 113189433       38   0.2095
## 11 Sample1     4 119644740.5 119754793       22  -0.2127
## 12 Sample1     4 120419825.0 186611721       82   0.3210
## 13 Sample1     5    442758.0  10761154       38   0.3618
## 14 Sample1     5  38869183.5 180687408      362  -0.2751
## 15 Sample1     6   2623865.0 144219759      297  -0.2386
## 16 Sample1     6 144224235.5 170862274      119   0.3558
## 17 Sample1     7    938572.5  14028656       57   0.3387
## 18 Sample1     7  23286512.0  23313764       11   1.6629
## 19 Sample1     7  26232167.0 156469232      316  -0.2431
## 20 Sample1     8   6264200.0 145537891      339  -0.2669
## 21 Sample1     9    214953.0 139440208      376  -0.2136
## 22 Sample1    10    323391.5  72576624      234   0.2935
## 23 Sample1    10  72604313.0  72645621       16  -0.2824
## 24 Sample1    10  72648289.5  75000741       30   0.2128
## 25 Sample1    10  82300671.5  82403794        7  -1.2325
## 26 Sample1    10  85982056.5  88768888       13   0.8651
## 27 Sample1    10  91066426.0  99790218       36  -1.1378
## 28 Sample1    10 102283640.5 102289566        5   1.2228
## 29 Sample1    10 103541552.5 121214530       74  -1.1989
## 30 Sample1    10 124591880.5 134121207       24   0.1823
## 31 Sample1    11   2291272.0  34378690      107  -0.2953
## 32 Sample1    11  36614927.0  44081430       15   0.4236
## 33 Sample1    11  46880700.5  57947384       74  -0.1784
## 34 Sample1    11  60692156.0  65172438       76   0.2811
## 35 Sample1    11  65340286.5  66335024       33  -0.2330
## 36 Sample1    11  66335504.5  71209486       26  -1.2652
## 37 Sample1    11  71847083.0  82549523       78   0.3924
## 38 Sample1    11  82550385.5 134134828      152  -0.2167
## 39 Sample1    12   1740561.0  99126272      377  -1.1687
## 40 Sample1    12 113537804.0 124428836      214   0.3099
## 41 Sample1    13  20398996.5 114438189      325  -1.2344
## 42 Sample1    14  20757846.0 101349088      323   0.2810
## 43 Sample1    15  27216709.5  99926272      403   0.3263
## 44 Sample1    16    230533.5  31123514      225   0.2965
## 45 Sample1    16  56899289.0  56947247       26   0.9363
## 46 Sample1    16  57507348.5  57722319       20  -0.2207
## 47 Sample1    16  66918983.0  90038049      186   0.3409
## 48 Sample1    17   1399145.5  76832320      625   0.3161
## 49 Sample1    17  77768896.0  80559278       25  -0.2662
## 50 Sample1    18   5394737.5  71825664      133  -0.2608
## 51 Sample1    19   1481982.5  57301280      504   0.3169
## 52 Sample1    20    207959.0  62610776      331   0.2914
## 53 Sample1    21  11098731.0  47865219      178   0.3297
## 54 Sample1    22  17443695.0  45996257      151   0.3248
## 55 Sample1    22  50703417.0  51066096       20  -0.3827

## Mean standard deviation of log-ratios:  0.37

## Optimizing purity and ploidy.  Will take a minute or two...

## Local optima:  0.65/1.6, 0.5/2.4, 0.9/2.4, 0.65/3.2, 0.85/4.4, 0.45/3.4, 0.4/2.8, 0.75/4.8, 0.55/2.1,
0.7/2.6, 0.5/1.9, 0.9/3.8, 0.3/1.6

## Testing local optimum at purity 0.65 and total ploidy 1.6.

## Fitting SNVs for purity 0.65 and tumor ploidy 1.38.

## Analyzing: Sample1

## Optimized purity:  0.65

## Testing local optimum at purity 0.5 and total ploidy 2.4.

## Fitting SNVs for purity 0.48 and tumor ploidy 2.76.

## Analyzing: Sample1
## Optimized purity:  0.48
## Testing local optimum at purity 0.9 and total ploidy 2.4.
## Fitting SNVs for purity 0.95 and tumor ploidy 2.38.
## Analyzing: Sample1
## Optimized purity:  0.95
## Testing local optimum at purity 0.65 and total ploidy 3.2.
## Fitting SNVs for purity 0.64 and tumor ploidy 3.76.
## Analyzing: Sample1
## Optimized purity:  0.64
## Testing local optimum at purity 0.85 and total ploidy 4.4.
## Fitting SNVs for purity 0.94 and tumor ploidy 4.76.
## Analyzing: Sample1
## Optimized purity:  0.94
## Testing local optimum at purity 0.45 and total ploidy 3.4.
## Fitting SNVs for purity 0.47 and tumor ploidy 5.14.
## Analyzing: Sample1
## Optimized purity:  0.47
## Testing local optimum at purity 0.4 and total ploidy 2.8.
## Fitting SNVs for purity 0.38 and tumor ploidy 4.14.
## Analyzing: Sample1
## Optimized purity:  0.38
## Testing local optimum at purity 0.75 and total ploidy 4.8.
## Fitting SNVs for purity 0.72 and tumor ploidy 5.66.
## Analyzing: Sample1
## Optimized purity:  0.72
## Testing local optimum at purity 0.55 and total ploidy 2.1.
## Fitting SNVs for purity 0.56 and tumor ploidy 2.29.
## Analyzing: Sample1
## Optimized purity:  0.56
## Testing local optimum at purity 0.7 and total ploidy 2.6.
## Fitting SNVs for purity 0.69 and tumor ploidy 2.85.
## Analyzing: Sample1
## Optimized purity:  0.69
## Testing local optimum at purity 0.5 and total ploidy 1.9.
## Fitting SNVs for purity 0.51 and tumor ploidy 1.85.
## Analyzing: Sample1
## Optimized purity:  0.51
## Testing local optimum at purity 0.9 and total ploidy 3.8.
## Fitting SNVs for purity 0.64 and tumor ploidy 3.76.
## Analyzing: Sample1
## Optimized purity:  0.64
## Testing local optimum at purity 0.3 and total ploidy 1.6.
## Recalibrating log-ratios...
## Testing local optimum at purity 0.3 and total ploidy 1.6.
## Recalibrating log-ratios...
## Testing local optimum at purity 0.3 and total ploidy 1.6.
## Recalibrating log-ratios...
## Testing local optimum at purity 0.3 and total ploidy 1.6.
## Fitting SNVs for purity 0.57 and tumor ploidy 4.68.
## Analyzing: Sample1
## Optimized purity:  0.57
## Remember, posterior probabilities assume a correct SCNA fit.
```

The post.optimize flag will increase the runtime by a lot, but might be worth it if the copy number profile is not very clean. This is recommended with lower coverage datasets or if there are significant capture biases. For high quality whole exome data, this is typically not necessary. Note the uncommented line, which is recommended, but not used here for simplicity. The `segmentationPSCBS` function requires the *PSCBS* package [6] and sometimes gives better results than the default when the coverage data is relatively noisy, especially for whole exome data with a large number of heterozygous SNPs. The `plot.cnv` argument is set to `FALSE` for this vignette, but generates often helpful additional plots provided by the segmentation function.

We also need to create a few output files:

```
file.rds <- 'Sample1_PureCN.rds'
saveRDS(ret, file=file.rds)
pdf('Sample1_PureCN.pdf', width=10, height=12)
plotAbs(ret, type='all')
dev.off()

## pdf
##   2
```

The first plot is an overview plot which shows the purity and ploidy local optima, sorted by final likelihood score after fitting both copy number and allelic fractions:

```
plotAbs(ret, type="overview")
```

We now look at the main plots of the maximum likelihood solution in more detail.

```
plotAbs(ret, 1, type="hist")
```

Figure 2 displays a histogram of tumor vs. normal copy number log-ratios for the maximum likelihood solution (number 1 in Figure 1). The height of a bar in this plot is proportional to the fraction of the genome falling into the particular log-ratio copy number range. The vertical dotted lines and numbers visualize the, for the given purity/ploidy combination, expected log-ratios for all integer copy numbers from 0 to 7. It can be seen that most of the log-ratios of the maximum likelihood solution align well to expected values for copy numbers of 0, 1 and 2.

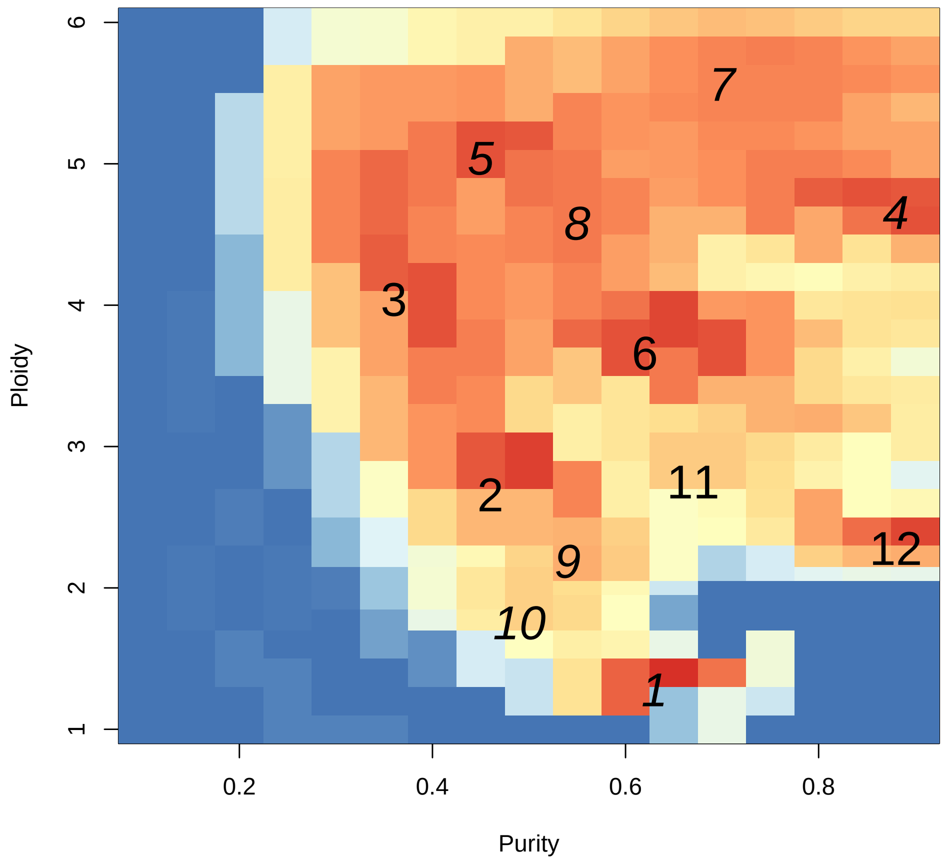

Figure 1: **Overview.** The colors visualize the copy number fitting score from low (blue) to high (red). The numbers indicate the ranks of the local optima.

```
plotAbs(ret, 1, type="BAF")
```

Germline variant data are informative for calculating integer copy number, because unbalanced maternal and paternal chromosome numbers in the tumor portion of the sample lead to unbalanced germline allelic fractions. Equations for calculating expected allelic fractions are given in the Supplemental Information of the *PureCN* manuscript. Figure 3 shows the allelic fractions of predicted germline SNPs. In the middle panel, the corresponding copy number log-ratios are shown. The lower panel displays the calculated integer copy numbers, corrected for purity and ploidy.

```
plotAbs(ret, 1, type="AF")
```

Finally, Figure 4 provides more insight into how well the variants fit the expected values. The left panel shows the correlation of expected and observed allelic fractions. The expected value is determined by the most likely state. High ploidy solutions have a lot of states, so this correlation is expected to be good for high ploidy solutions. The right panel plots the observed allelic fractions against copy number. The labels show the expected values for all called states; 2m1 would be diploid, heterozygous, 2m2 diploid, homozygous.

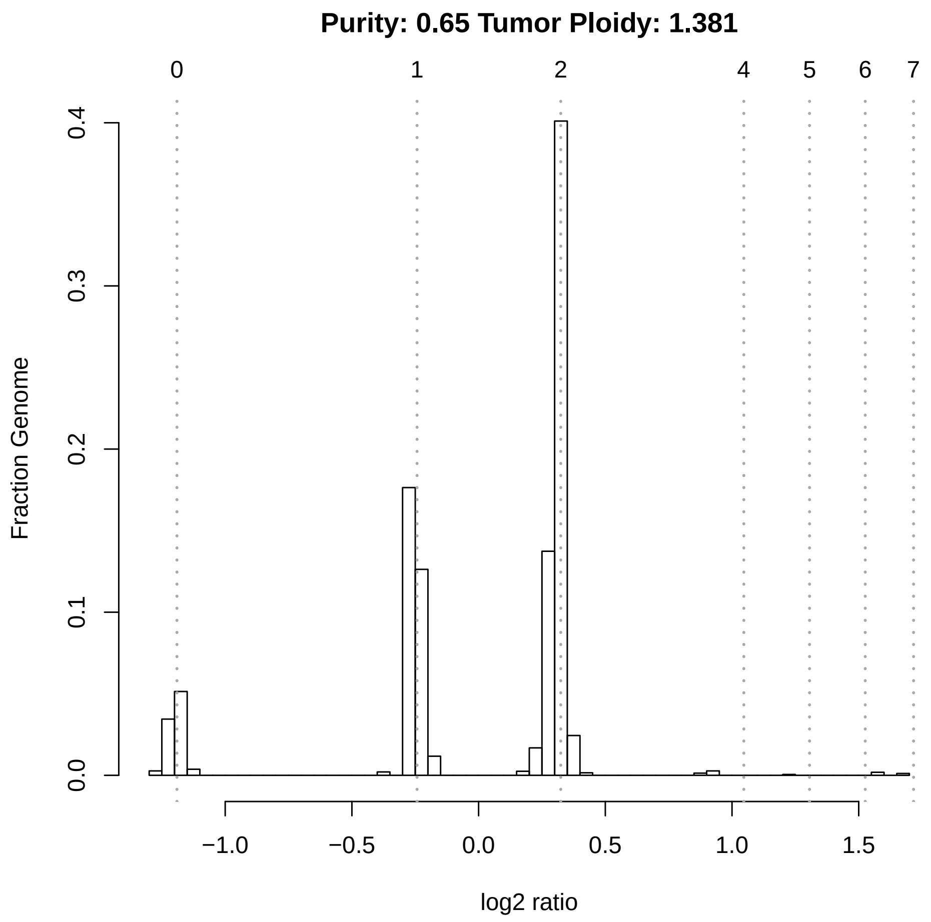

Figure 2: **Log-ratio histogram.**

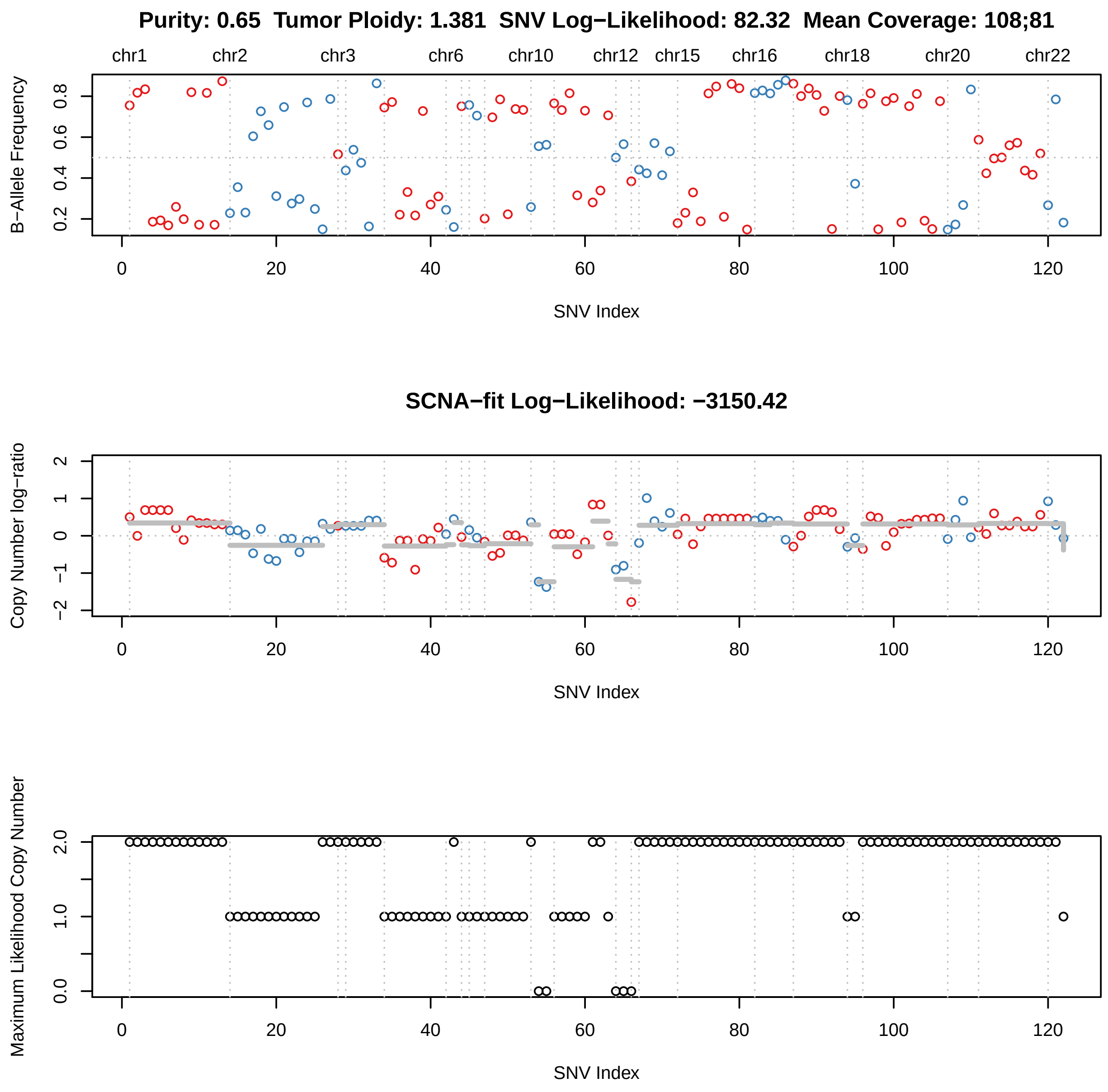

Figure 3: **B-allele frequency plot.** Each dot is a (predicted) germline SNP.

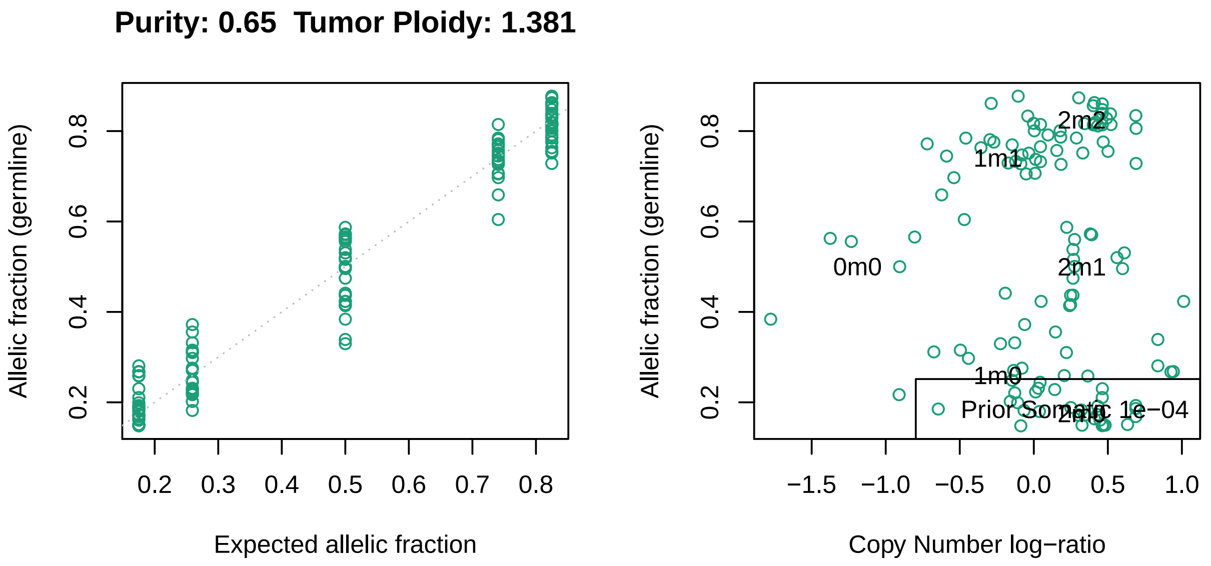

Figure 4: **Allele fraction plots.** Each dot is again a (predicted) germline SNP. This plot normally also shows somatic mutations in two additional panels. This toy example contains only germline SNPs however.

8 Manual curation

For prediction of SNV status (germline vs. somatic, sub-clonal vs. clonal, homozygous vs. heterozygous), it is important that both purity and ploidy are correct. We provide functionality for curating results:

```
createCurationFile(file.rds)
```

This will generate a CSV file, in which the correct purity and ploidy values can be manually entered. It also contains a column "Curated", which should be set to TRUE, otherwise the file will be overwritten when re-run.

Then in R, the correct solution (closest to the combination in the CSV file) can be loaded with the `readCurationFile` function:

```
ret <- readCurationFile(file.rds)
```

This function has various handy features, but most importantly it will re-order the local optima so that the curated purity and ploidy combination is ranked first. This means `plotAbs(ret,1,type="hist")` would show the plot for the curated purity/ploidy combination, for example.

The default curation file will list the maximum likelihood solution:

```
read.csv('Sample1_PureCN.csv')

##   Sampleid Purity   Ploidy Flagged Failed Curated                      Comment
## 1  Sample1   0.65 1.380783    TRUE  FALSE   FALSE RARE KARYOTYPE;EXCESSIVE LOH
```

PureCN currently only flags samples with warnings, it does not mark any samples as failed. The `Failed` column in the curation file can be used to manually flag samples for exclusion in downstream analyses.

9 Custom segmentation

By default, we will use *DNAcopy* [7] to segment the log-ratio. It is straightforward to replace the default with other methods and the `segmentationCBS` function can serve as an example. The `segmentationPSCBS` function is another example which uses the *PSCBS* package [6].

It is also possible to provide already segmented data, which we however only recommend when matched SNP6 data is available. Otherwise it is usually better to customize the segmentation function as described above, since the algorithm then has access to the raw log-ratio distribution. The expected file format for already segmented copy number data is:

```
ID   chrom   loc.start   loc.end num.mark    seg.mean
Sample1   1   61723   5773942 2681    0.125406444072723
Sample1   1   5774674 5785170 10  -0.756511807441712
```

10 Output

The `plotAbs()` call above will generate the main plots shown in the manuscript. The R data file (file.rds) contains gene level copy number calls, SNV status and LOH calls. The purity/ploidy combinations are sorted by likelihood and stored in `ret$results`.

```
names(ret)

## [1] "candidates" "results"    "input"

head(ret$results[[1]]$gene.calls, 3)
```

```
##         chr   start     end C seg.mean seg.id number.exons gene.mean    gene.min gene.max
## SCNN1D chr1 1216042 1226991 2    0.343      1           18 0.3557916 -0.14729185 1.442539
## CEP104 chr1 3731966 3768972 2    0.343      1           21 0.4352325 -0.03504123 1.262746
## GPR153 chr1 6309398 6314966 2    0.343      1            5 0.1947442 -0.75719595 1.098475
##        focal num.snps.loh.segment percentage.loh.in.loh.segment
## SCNN1D FALSE                    0                             0
## CEP104 FALSE                    0                             0
## GPR153 FALSE                    0                             0
```

This data.frame also contains gene level LOH information. The SNV posteriors:

```
head(ret$results[[1]]$SNV.posterior$beta.model$posteriors, 3)
```

```
##                  seqnames     start       end SOMATIC.M0   SOMATIC.M1   SOMATIC.M2
## chr1114515871xxx     chr1 114515871 114515871          0 1.542640e-34 1.474571e-05
## chr1150044293xxx     chr1 150044293 150044293          0 1.502780e-35 8.313631e-09
## chr1158449835xxx     chr1 158449835 158449835          0 6.896309e-57 1.195370e-12
##                     SOMATIC.M3 SOMATIC.M4 SOMATIC.M5 SOMATIC.M6 SOMATIC.M7  GERMLINE.M0
## chr1114515871xxx 1.807752e-214          0          0          0          0 5.771584e-64
## chr1150044293xxx 1.040274e-216          0          0          0          0 1.166806e-59
## chr1158449835xxx 3.492470e-218          0          0          0          0 2.025737e-98
##                   GERMLINE.M1 GERMLINE.M2   GERMLINE.M3 GERMLINE.M4 GERMLINE.M5
## chr1114515871xxx 2.079033e-10   0.9999853 4.604243e-214           0           0
## chr1150044293xxx 9.691973e-14   1.0000000 3.208862e-212           0           0
## chr1158449835xxx 1.059012e-23   1.0000000 5.013249e-212           0           0
##                  GERMLINE.M6 GERMLINE.M7 GERMLINE.CONTHIGH GERMLINE.CONTLOW ML.SOMATIC
## chr1114515871xxx           0           0      1.019206e-47    7.460949e-277      FALSE
## chr1150044293xxx           0           0      3.547817e-25    1.039791e-233      FALSE
## chr1158449835xxx           0           0      4.262958e-33     0.000000e+00      FALSE
##                  ML.M ML.C ML.AR       AR CN.Subclonal    Log.Ratio Prior.Somatic
## chr1114515871xxx    2    2 0.825 0.755183        FALSE  0.501074161   9.90099e-05
## chr1150044293xxx    2    2 0.825 0.817078        FALSE -0.001780057   9.90099e-05
## chr1158449835xxx    2    2 0.825 0.834266        FALSE  0.688988158   9.90099e-05
##                  Prior.Contamination ML.LOH num.snps.loh.segment
## chr1114515871xxx                0.01   TRUE                   13
## chr1150044293xxx                0.01   TRUE                   13
## chr1158449835xxx                0.01   TRUE                   13
##                  percentage.loh.in.loh.segment
## chr1114515871xxx                             1
## chr1150044293xxx                             1
## chr1158449835xxx                             1
```

This lists all posterior probabilities for all possible SNV states. M0 to M7 are multiplicity values, i.e. the number of chromosomes harboring the mutation (e.g. 1 heterozygous, 2 homozygous if copy number C is 2). Columns with the ML prefix indicate maximum likelihood estimates, e.g. ML.AR is the expected allelic ratio of the most likely state, AR is the observed allelic ratio. GERMLINE.CONTHIGH and GERMLINE.CONTLOW are the two contamination states. The former are homozygous germline SNPs that were not filtered out because reference alleles from another individual were sequenced, resulting in allelic fractions smaller than 1. The latter are non-reference alleles only present in the contamination.

11 Prediction of somatic status and cellular fraction

The SNV posteriors above provide posterior probabilities for all possible states. The `predictSomatic` function adjusts these posterior probabilities by excluding germline states that do not correspond to the maternal and paternal chromosome numbers (since *PureCN* only considers heterozygous variants, SNPs are either from the maternal or paternal chromosome). For predicted somatic mutations, this function also provides cellular fraction estimates, i.e. the fraction of cells with mutation. Fractions significantly below 1 indicate sub-clonality[2].

```
head(predictSomatic(ret), 3)

##                   seqnames     start       end SOMATIC.M0   SOMATIC.M1   SOMATIC.M2
## chr1114515871xxx      chr1 114515871 114515871          0 1.542640e-34 1.474571e-05
## chr1150044293xxx      chr1 150044293 150044293          0 1.502780e-35 8.313631e-09
## chr1158449835xxx      chr1 158449835 158449835          0 6.896309e-57 1.195370e-12
##                     SOMATIC.M3 SOMATIC.M4 SOMATIC.M5 SOMATIC.M6 SOMATIC.M7  GERMLINE.M0
## chr1114515871xxx 1.807752e-214          0          0          0          0 5.771584e-64
## chr1150044293xxx 1.040274e-216          0          0          0          0 1.166806e-59
## chr1158449835xxx 3.492470e-218          0          0          0          0 2.025737e-98
##                  GERMLINE.M1 GERMLINE.M2 GERMLINE.M3 GERMLINE.M4 GERMLINE.M5 GERMLINE.M6
## chr1114515871xxx           0   0.9999853           0           0           0           0
## chr1150044293xxx           0   1.0000000           0           0           0           0
## chr1158449835xxx           0   1.0000000           0           0           0           0
##                  GERMLINE.M7 GERMLINE.CONTHIGH GERMLINE.CONTLOW ML.SOMATIC ML.M ML.C
## chr1114515871xxx           0      1.019206e-47    7.460949e-277      FALSE    2    2
## chr1150044293xxx           0      3.547817e-25    1.039791e-233      FALSE    2    2
## chr1158449835xxx           0      4.262958e-33     0.000000e+00      FALSE    2    2
##                  ML.AR       AR CN.Subclonal    Log.Ratio Prior.Somatic
## chr1114515871xxx 0.825 0.755183        FALSE  0.501074161   9.90099e-05
## chr1150044293xxx 0.825 0.817078        FALSE -0.001780057   9.90099e-05
## chr1158449835xxx 0.825 0.834266        FALSE  0.688988158   9.90099e-05
##                  Prior.Contamination ML.LOH num.snps.loh.segment
## chr1114515871xxx                0.01   TRUE                   13
## chr1150044293xxx                0.01   TRUE                   13
## chr1158449835xxx                0.01   TRUE                   13
##                  percentage.loh.in.loh.segment Cellfraction
## chr1114515871xxx                             1           NA
## chr1150044293xxx                             1           NA
## chr1158449835xxx                             1           NA
```

Segment 1 in this toy example has clearly copy number 2 and LOH, so heterozygous mutations are impossible and the posterior probabilities for `GERMLINE.M1` are set to 0.

References

[1] Scott L. Carter, Kristian Cibulskis, Elena Helman, Aaron McKenna, Hui Shen, Travis Zack, Peter W. Laird, Robert C. Onofrio, Wendy Winckler, Barbara A. Weir, Rameen Beroukhim, David Pellman, Douglas A. Levine, Eric S. Lander, Matthew Meyerson, and Gad Getz. Absolute quantification of somatic DNA alterations in human cancer. *Nature biotechnology*, 30(5):413–421, May 2012.

[2] Aaron McKenna, Matthew Hanna, Eric Banks, Andrey Sivachenko, Kristian Cibulskis, Andrew Kernytsky, Kiran Garimella, David Altshuler, Stacey Gabriel, Mark Daly, and Mark A. DePristo. The Genome Analysis Toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome research*, 20(9):1297–1303, Sep 2010.

[2]This number can be above 1 when the observed allelic fraction is higher than expected for a clonal mutation. This maybe due to random sampling, wrong copy number, sub-clonal copy number events, or wrong purity/ploidy estimates.

[3] Jarupon Fah Sathirapongsasuti, Hane Lee, Basil A. J. Horst, Georg Brunner, Alistair J. Cochran, Scott Binder, John Quackenbush, and Stanley F. Nelson. Exome sequencing-based copy-number variation and loss of heterozygosity detection: ExomeCNV. *Bioinformatics (Oxford, England)*, 27(19):2648–2654, Oct 2011.

[4] Kristian Cibulskis, Michael S. Lawrence, Scott L. Carter, Andrey Sivachenko, David Jaffe, Carrie Sougnez, Stacey Gabriel, Matthew Meyerson, Eric S. Lander, and Gad Getz. Sensitive detection of somatic point mutations in impure and heterogeneous cancer samples. *Nature biotechnology*, 31(3):213–219, Mar 2013.

[5] Gavin Ha, Andrew Roth, Jaswinder Khattra, Julie Ho, Damian Yap, Leah M. Prentice, Nataliya Melnyk, Andrew McPherson, Ali Bashashati, Emma Laks, Justina Biele, Jiarui Ding, Alan Le, Jamie Rosner, Karey Shumansky, Marco A. Marra, C. Blake Gilks, David G. Huntsman, Jessica N. McAlpine, Samuel Aparicio, and Sohrab P. Shah. TITAN: inference of copy number architectures in clonal cell populations from tumor whole-genome sequence data. *Genome research*, 24(11):1881–1893, Nov 2014.

[6] Adam B. Olshen, Henrik Bengtsson, Pierre Neuvial, Paul T. Spellman, Richard A. Olshen, and Venkatraman E. Seshan. Parent-specific copy number in paired tumor-normal studies using circular binary segmentation. *Bioinformatics (Oxford, England)*, 27(15):2038–2046, Aug 2011.

[7] E. S. Venkatraman and Adam B. Olshen. A faster circular binary segmentation algorithm for the analysis of array CGH data. *Bioinformatics (Oxford, England)*, 23(6):657–663, Mar 2007.

Session Info

- R version 3.3.1 (2016-06-21), `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`
- Base packages: base, datasets, grDevices, graphics, methods, parallel, stats, stats4, utils
- Other packages: Biobase 2.32.0, BiocGenerics 0.18.0, Biostrings 2.40.2, DNAcopy 1.46.0, GenomeInfoDb 1.8.2, GenomicRanges 1.24.2, IRanges 2.6.1, PureCN 1.0.4, Rsamtools 1.24.0, S4Vectors 0.10.2, SummarizedExperiment 1.2.3, VariantAnnotation 1.18.3, XVector 0.12.0
- Loaded via a namespace (and not attached): AnnotationDbi 1.34.4, BSgenome 1.40.1, BiocParallel 1.6.2, BiocStyle 2.0.2, DBI 0.4-1, GenomicAlignments 1.8.4, GenomicFeatures 1.24.4, RColorBrewer 1.1-2, RCurl 1.95-4.8, RSQLite 1.0.0, XML 3.98-1.4, biomaRt 2.28.0, bitops 1.0-6, chron 2.3-47, data.table 1.9.6, evaluate 0.9, formatR 1.4, highr 0.6, knitr 1.13, magrittr 1.5, rtracklayer 1.32.1, stringi 1.1.1, stringr 1.0.0, tools 3.3.1, zlibbioc 1.18.0