# Using genomicInstability, a package for inferring the genomic instability of cells from scRNA-Seq data

Pasquale Laise[1,2] and Mariano J. Alvarez[1,2]

[1]DarwinHealth Inc, 333 West 39th Street, New York, NY 10018
[2]Department of Systems Biology, Columbia University, 1130 St. Nicholas Ave., New York, NY 10032

October 27, 2021

## 1 Overview of genomicInstability

Cellular heterogeneity is a hallmark of cancer[1]. Transcriptional analysis of individual cells (single-cell RNA sequencing, scRNA-Seq) has significantly improved our understanding of intratumor heterogeneity, showing that many tumors are composed by normal and neoplastic cells in different phenotypic states—cellular subtypes[2, 3]. Consequently, the analysis of tumor-related transcriptional programs requires an accurate classification of each individual cell as putative neoplastic or normal.

Computational approaches for the identification of putative malignant and normal cells from single cell gene expression profiles typically rely on unsupervised cluster analysis, followed by cell type identification based on the expression of known marker genes, and methods based on copy number variation (CNV) inference from single-cell gene expression data[2, 3, 4, 5, 6, 7]. Cluster based analyses are affected by multiple factors, including the choice of similarity metric and cluster algorithm, and inclusion/exclusion criteria of individual marker genes used for cluster annotation. CNV-based methods, instead, are typically computationally intensive, often requiring low-level sequencing data (BAM files) as input[7], as well as prior biological knowledge, such as a tumor-matched normal reference samples[2] or a single-nucleotide variant (SNV) call set[6], which are not always available.

Here, we present the implementation of a new genomic instability analysis (GIA) method for identifying malignant cells from single-cell gene expression data, which does not require any prior biological knowledge or matched normal samples. GIA uses the aREA algorithm[8] to quantitatively estimate the association between gene expression and chromosomal location by performing enrichment analysis of contiguously coded genes (loci-blocks) on the single cell gene expression profiles. Such associations, which are purely the product of transcriptional co-regulation in normal cells, can also be generated by genomic alterations typical of neoplastic cells, like amplification and deletion of chromosomal regions, collectively referred to as CNVs. In a mixture of normal and tumor cells, as it is usually the case when tumor samples are profiled at the single-cell level, it is to expect that the genomic aberrations in the tumor cells would account for a globally higher level of loci-block enrichment. In particular, a higher genomic instability—more CNVs–would be reflected on a larger dynamic range for the enrichment of the loci-blocks on the expression profile. GIA uses the variance of such enrichment, across all loci-bloks, to quantitatively estimate the level of genomic instability for each cell (Genomic Instability Score, GIS).

The package `genomicInstability` provides an implementation of the GIA algorithm for the R-system environment, including functions to quantify the association between expression and loci-blocks (`inferCNV()`); functions to estimate the genomic instability score of each cell (`genomicInstabilityScore()`); functions to estimate the likelihood of each of the analyzed cells to be genomically unstable (`giLikelihood()`); as well as

functions for the graphical representation of the results, including heatmaps for the loci-blocks and density distribution plots for the genomic instability estimates.

# 2 Installation of *genomicInstability* package

The *genomicInstability* package requires the R-system, the checkmate package, the mixtools package, and the ExperimentHub, SingleCellExperiment and pROC packages to run the examples in this vignette. After installing R, all required components can be obtained with:

```
> if (!requireNamespace("BiocManager", quietly=TRUE))
+     install.packages("BiocManager")
> install.packages("checkmate")
> BiocManager::install("mixtools")
> BiocManager::install("SingleCellExperiment")
> BiocManager::install("ExperimentHub")
> BiocManager::install("genomicInstability")
> install.packages("pROC")
```

# 3 Use-case example for the analysis of HNSC and normal cells

For the example analysis in this vignette, we will use the Head and Neck Squamous Carcinoma (HNSC) scRNA-Seq dataset by Puram et. al [3]. This dataset is available from Gene Expression Omnibus (GEO) as GSE102233.

## 3.1 Getting started

As first step, we have to load the genomicInstability environment with:

```
> library(genomicInstability)
```

## 3.2 Estimating the association between gene expression and loci-blocks

For convenience, we will obtain the data and metadata for this example case as a *SingleCellExperiment*-class data object using the package `ExpressionHub`.

The access number for GSE102233 in ExperimentHub is "EH5419" and we can access the data and metadata as a SingleCellExperiment-class (SCE) object with:

```
> library(SingleCellExperiment)
> library(ExperimentHub)
> eh <- ExperimentHub()
> dset <- eh[["EH5419"]]
```

The data in this SCE-class object is a Transcripts-per-million (TPM) matrix for 21,341 known genes (EntrezID) across 5,902 cells. The metadata includes information regarding the enzyme used for reverse transcription, whether the sample was isolated from lymph nodes or is a primary tumor, and whether the cells are classified as tumor or normal, as well as the type of normal cell according to Puram et. al[3]. To reduce execution time, we will limit the analysis in this vignette to a subset of 500 cells, selected uniformly at random—we fix the random seed for reproducibility purposes—from the 5,902 represented in the dataset.

```
> tpm_matrix <- assays(dset)$TPM
> set.seed(1)
> pos <- sample(ncol(tpm_matrix), 500)
```

```
> tpm_matrix <- tpm_matrix[, pos]
> metadata <- colData(dset)
```

The association between gene expression and loci-blocks is estimated by the function `inferCNV()`. The minimum input for this function is a matrix of gene expression. If available, a matrix of gene expression for a normal (non-neoplastic) samples can also be provided. The matrix of normal samples is not essential nor strictly required by the algorithm, but it provides a better prior for the estimation of CNVs and the tumor vs. normal likelihood (see below).

By default, `inferCNV()` generates loci-blocks composed by 100 contiguous genes and with 75% overlap between blocks. This parameters can be modified by the `species`, `k` and `skip` arguments of the `inferCNV()` function, which define the species to be analyzed—currently "human" and "mouse" are implemented—, the number of genes for each loci-block, and the displacement, in number of genes, between loci-blocks, respectively.

For this example, we are going to use only the gene expression matrix and default parameters. The output of the `inferCNV` function is an inferCNV-class object with slots `nes`, `null` and `param`. The enrichment of each loci-block on the gene expression profile of each cell/sample is expressed as Normalized Enrichment Scores (NES), and is stored in the `nes` slot.

```
> cnv <- inferCNV(tpm_matrix)
> class(cnv)

[1] "inferCNV"

> names(cnv)

[1] "nes"   "null"  "param"

> dim(cnv$nes)

[1] 926 500
```

### 3.3 Estimating the Genomic Instability Score (GIS)

The Genomic Instability Score (GIS) is estimated as the $\log_2$ of the NES variance for each cell, and the computation is performed by the function `genomicInstabilityScore()`. This function generates an updated inferCNV-class object including a vector of computed GIS, with length equal to the number of cells—500 for this example—, as the slot `gis`.

```
> cnv <- genomicInstabilityScore(cnv)
> names(cnv)

[1] "nes"   "null"  "param" "gis"

> length(cnv$gis)

[1] 500
```

For a dataset consisting on a mixture of normal and tumor cells, we would expect GIS to be distributed with at least two modes (fig. 1).

```
> par(mai=c(.8, .8, .2, .2))
> plot(density(cnv$gis), lwd=2, xlab="GIS", main="")
```

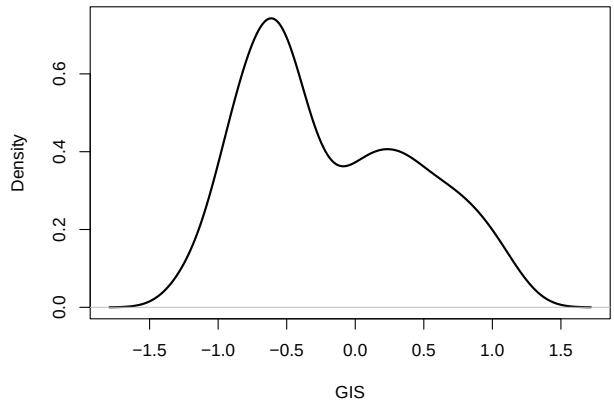


Figure 1: Density distribution for the GIS.

### 3.4 Estimating the relative likelihood of a cell to be neoplastic

The relative likelihood for each of the cells of being genomically unstable (neoplastic) can be estimated from a mixture of Gaussian distributions fitted to the GIS estimates. Parameter estimates for the mixture of Gaussian distributions are computed by the `giLikelihood()` function. The output is an updated inferCNV-class object with two new slots: (1) `cnv_fit`, which is a list of fitted parameters and fitting metrics, and (2) `gi_likelihood`, which contains a vector of estimated likelihood of genomic instability for the analyzed cells.

```
> cnv <- giLikelihood(cnv)

> names(cnv)

[1] "nes"           "null"          "param"         "gis"
[5] "gi_fit"        "gi_likelihood"

> names(cnv$gi_fit)

[1] "mu"     "sigma"  "lambda" "loglik" "mse"

> length(cnv$gi_likelihood)

[1] 500
```

The fitted Gaussian models can be visualized with the function `giDensityPlot()` (Fig. 2).

```
> giDensityPlot(cnv)
```

Inspection of the Gaussian distributions fitted to the GIS (see Fig. 2) suggests that the Gaussian model with smallest mean might represent normal, genomically stable cells, while the 2 Gaussian models with largest mean might represent the genomically unstable, tumor cells. Because the `giLikelihood()` function default is to consider the gaussian model with lowest mean as representing genomically stable (normal) cells, while all other fitted models as representing genomically unstable (tumor) cells, we could directly interpret the estimates in the `gi_likelihood` as the probability of each of the cells to be genomically unstable. This behavior can also be forced, in particular for cases not being properly accounted by the `giLikelihood()` function defaults, indicating the fitted models most likely representing genomically stable (normal) and unstable (tumor) cells:

```
> cnv <- giLikelihood(cnv, recompute=FALSE, normal=1, tumor=2:3)
```

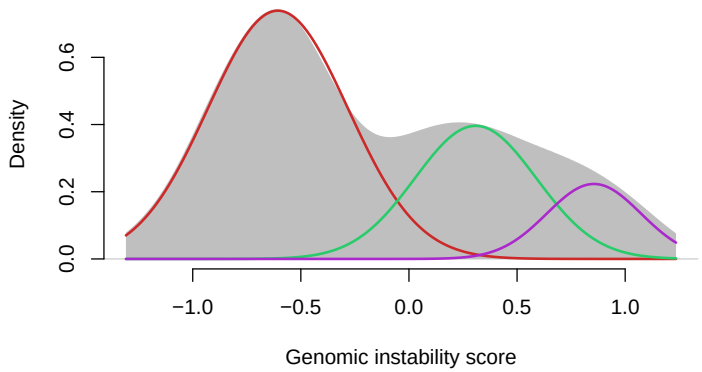


Figure 2: Distribution density for the GIS (grey filled density curve) and 3 gaussian distributions fitted to it.

We can see the values for the estimated likelihood displayed over the GIS distribution (Fig. 3) with the following code:

```
> # Plotting the density distribution for GIS and fitted models
> par(mai=c(.8, .8, .2, .8))
> giDensityPlot(cnv, ylim=c(0, .8))
> # Adding the likelihood data and second-axis
> pos <- order(cnv$gis)
> lines(cnv$gis[pos], cnv$gi_likelihood[pos]*.8, lwd=2)
> axis(4, seq(0, .8, length=6), seq(0, 1, .2))
> axis(4, .4, "Relative likelihood", tick=FALSE, line=1.5)
> pos5 <- which.min((.5-cnv$gi_likelihood)^2)
> lines(c(rep(cnv$gis[pos5], 2), max(cnv$gis*1.05)), c(0,
+   rep(cnv$gi_likelihood[pos5]*.8, 2)), lty=3)
```

### 3.5 Using the genomically stable cells as reference for the analysis

We can use the cells showing the highest likelihood of being genomically stable (Genomic Instability (GI) likelihood $< 0.25$) as reference when estimating the enrichment of the loci-blocks and computing the GIS.

```
> cnv_norm <- inferCNV(tpm_matrix, nullmat=tpm_matrix[, cnv$gi_likelihood<.25,
+                                                     drop=FALSE])
> names(cnv_norm)

[1] "nes"   "null"  "param"
```

Using a set of normal cells as reference allows for a better estimation of the loci-block enrichment (NES), GIS and GI likelihood, which can be estimated by the function `genomicInstabilityScore()` by setting its argument `likelihood=TRUE`:

```
> cnv_norm <- genomicInstabilityScore(cnv_norm, likelihood=TRUE)
```

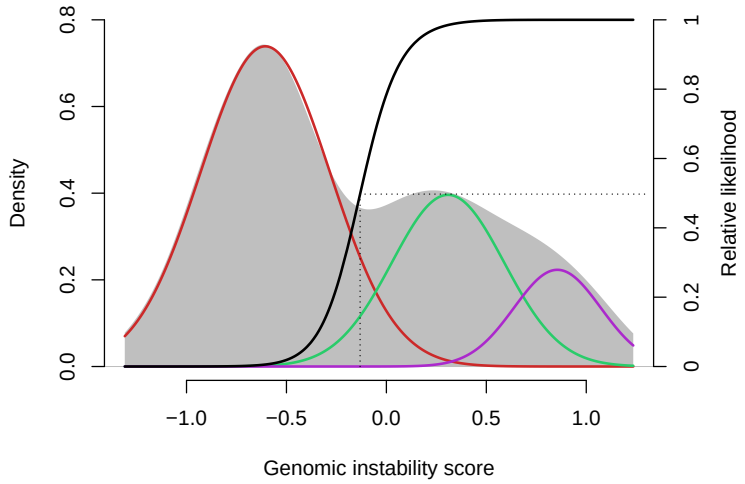


Figure 3: Distribution density for the GIS (grey filled density curve) and 3 Gaussian distributions fitted to it (shown in red, green and purple; corresponding to the left *y-axis*), and estimated relative likelihood for each cell to be genomically unstable (shown by the black line; corresponding to the right *y-axis*). The dotted line indicates the GIS corresponding to equal probability of a cell of being genomically stable or unstable.

The fitted Gaussian models, as well as the NULL model based on the cells previously considered as "genomically stable", can be displayed over the GIS density plot with the function `giDensityPlot()`. The following code shows how to add the genomic instability likelihood estimations to the plot (Fig. 4):

```
> # Plotting the density distribution for GIS and fitted models
> par(mai=c(.8, .8, .2, .8))
> giDensityPlot(cnv_norm, ylim=c(0, 1.1))
> # Adding the likelihood data and second-axis
> pos <- order(cnv_norm$gis)
> lines(cnv_norm$gis[pos], cnv_norm$gi_likelihood[pos], lwd=2, col="blue")
> axis(4, seq(0, 1, length=6), seq(0, 1, .2), col="blue", col.axis="blue")
> axis(4, .5, "Relative likelihood", tick=FALSE, line=1.5, col.axis="blue")
> pos5 <- which.min((.5-cnv_norm$gi_likelihood)^2)
> lines(c(rep(cnv_norm$gis[pos5], 2), max(cnv_norm$gis*1.05)),
+       c(0, rep(cnv_norm$gi_likelihood[pos5], 2)), lty=3, col="blue")
```

The distribution for the GIS based on the original classification of the cells by Puram et.al [3] can be displayed over this last plot with the following code (Fig. 5):

```
> metadata_tumor <- as.vector(metadata$classified..as.cancer.cell)[match(colnames(tpm_matrix),
+     rownames(metadata))]
> metadata_tumor <- as.logical(as.numeric(metadata_tumor))
> # Plotting the density distribution for GIS and fitted models
> par(mai=c(.8, .8, .2, .8))
> giDensityPlot(cnv_norm, ylim=c(0, 1.1))
> # Estimating GIS density distributions for normal and tumor cells
> gis_normal <- cnv_norm$gis[!metadata_tumor]
> gis_tumor <- cnv_norm$gis[metadata_tumor]
> den_normal <- density(gis_normal, from=min(gis_normal), to=max(gis_normal))
> den_tumor <- density(gis_tumor, from=min(gis_tumor), to=max(gis_tumor))
```

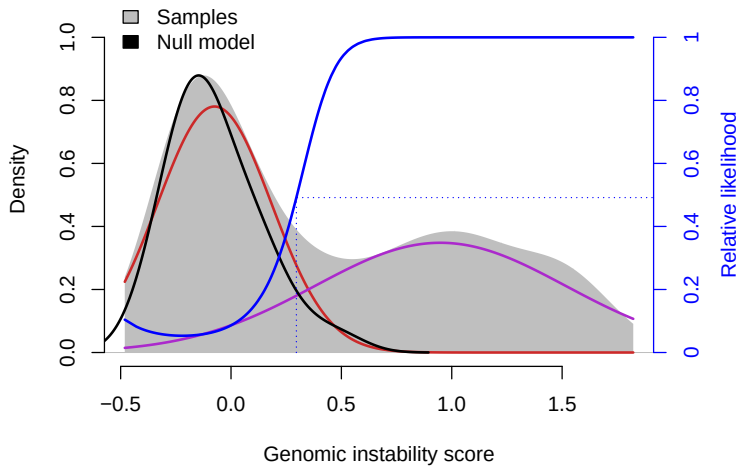


Figure 4: Distribution density for the GIS (grey filled density curve) and 2 Gaussian distributions fitted to it (shown in red and purple). The GIS distribution for the NULL model (cells considered genomically stable) is shown by the black line. The estimated relative likelihood for each cell to be genomically unstable is shown by the blue line. The dotted line indicates the GIS corresponding to equal probability of a cell of being genomically stable or unstable.

```
> # Scaling the densities based on the inferred proportions for each group
> den_normal$y <- den_normal$y * cnv_norm$gi_fit$lambda[1]
> den_tumor$y <- den_tumor$y * sum(cnv_norm$gi_fit$lambda[2])
> # Function to add the density plot
> addDensity <- function(x, col="grey") {
+   polygon(c(x$x[1], x$x, x$x[length(x$x)], x$x[1]), c(0, x$y, 0, 0), col=col)
+ }
> # Adding the density plots
> addDensity(den_normal, col=hsv(.6, .8, .8, .4))
> addDensity(den_tumor, col=hsv(.05, .8, .8, .4))
> # Adding the likelihood data and second-axis
> pos <- order(cnv_norm$gis)
> lines(cnv_norm$gis[pos], cnv_norm$gi_likelihood[pos], lwd=2, col="darkgreen")
> axis(4, seq(0, 1, length=6), seq(0, 1, .2), col="darkgreen", col.axis="darkgreen")
> axis(4, .5, "Relative likelihood", tick=FALSE, line=1.5, col.axis="darkgreen")
> pos5 <- which.min((.5-cnv_norm$gi_likelihood)^2)
> lines(c(rep(cnv_norm$gis[pos5], 2), max(cnv_norm$gis*1.05)),
+       c(0, rep(cnv_norm$gi_likelihood[pos5], 2)), lty=3, col="blue")
> # Adding the legend
> legend(c(.9, 1), c("Normal", "Tumor"), fill=hsv(c(.6, .05), .8, .8, .4), bty="n")
```

The results in figure 5 show a good agreement between GIA-based inferences of genomic instability and the classification performed by Puram et.al. In fact, we can estimate the statistical significance of the overlap between Puram et.al classification and the GIA-based inferences—i.e. GI likelihood $> 0.5$—using the Fisher's Exact Test (FET):

```
> fisher.test(metadata_tumor, cnv_norm$gi_likelihood>.5, alternative="greater")

        Fisher's Exact Test for Count Data
```

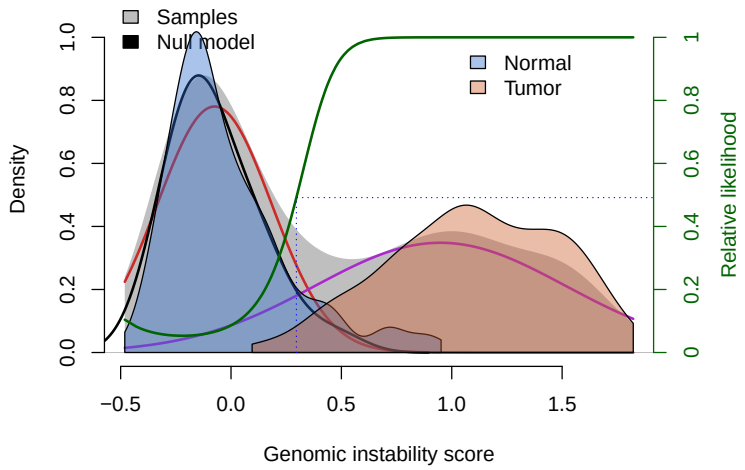


Figure 5: Distribution density for the GIS (grey filled density curve) and 2 Gaussian distributions fitted to it (shown in red and purple). The GIS distribution for the NULL model (cells considered genomically stable) is shown by the black line. The estimated relative likelihood for each cell to be genomically unstable is shown by the green line. The dotted line indicates the GIS corresponding to equal probability of a cell of being genomically stable or unstable. The GIS for the normal and tumor cells according to Puram et. al [3] is shown by the solid blue and red distributions, respectively.

```
data:  metadata_tumor and cnv_norm$gi_likelihood > 0.5
p-value < 2.2e-16
alternative hypothesis: true odds ratio is greater than 1
95 percent confidence interval:
 164.4681      Inf
sample estimates:
odds ratio
  456.8105
```

To explore further this agreement, we can do a Received Operating Characteristic (ROC) curve analysis (Fig. 6). This analysis requires the `pROC` package, which is available from CRAN (`https://cran.r-project.org/web/packages/pROC/`).

```
> # Loading the pROC package
> library(pROC)
> roc_res <- roc(response=as.numeric(metadata_tumor), predictor=cnv_norm$gi_likelihood, auc=TRUE, ci=TRU
> par(mai=c(.8, .8, .2, .8))
> plot(roc_res)
> legend("bottomright", c(paste0("AUC: ", round(roc_res$auc, 3)), paste0("CI 95%: ",
+   round(roc_res$ci[1], 3), "-", round(roc_res$ci[3], 3))), bty="n")
```

### 3.6 Plotting the loci-block enrichment results

The enrichment for each of the loci-blocks in each of the analyzed cells can be plotted using a heatmap. This can be done with the `plot()` function (Fig. 7).

```
> plot(cnv_norm, output="heatmap.png", resolution=120, gamma=2)
```

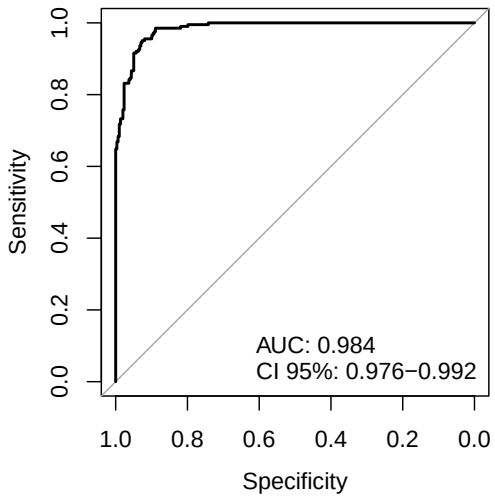


Figure 6: ROC curve, Area Under the Cure (AUC) and its 95% confidence interval (CI) for the GIS, as predictor of genomically unstable (tumor) cells.

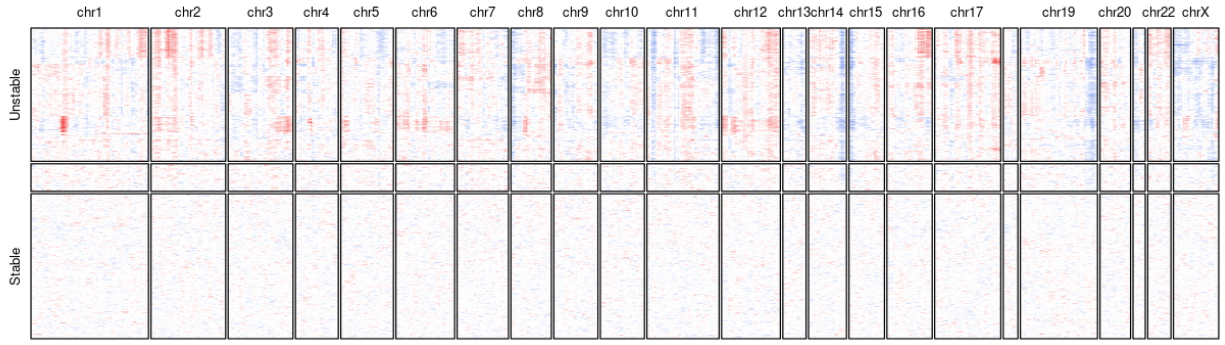


Figure 7: Heatmap for the enrichment of loci-blocks on the expression profile of each cell. Loci-blocks (columns) have been organized by chromosome. Cells (rows) have been sorted by GIS and divided in 3 groups: (1) Unstable (GI likelihood $> 0.8$), (2) Intermediate ($0.2 \geq$ GI likelihood $\geq 0.8$), and (3) Stable (GI likelihood $< 0.2$). Red color in the heatmap indicates potential genomic amplifications, while blue color shows potential deletions.

# 4 Session Information

```
R Under development (unstable) (2021-10-19 r81077)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Ubuntu 20.04.3 LTS

Matrix products: default
BLAS:   /home/biocbuild/bbs-3.15-bioc/R/lib/libRblas.so
LAPACK: /home/biocbuild/bbs-3.15-bioc/R/lib/libRlapack.so

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_GB              LC_COLLATE=C
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] stats4    stats     graphics  grDevices utils     datasets  methods
[8] base

other attached packages:
 [1] pROC_1.18.0                 GSE103322_0.99.1
 [3] GEOquery_2.63.0             ExperimentHub_2.3.0
 [5] AnnotationHub_3.3.0         BiocFileCache_2.3.0
 [7] dbplyr_2.1.1                SingleCellExperiment_1.17.0
 [9] SummarizedExperiment_1.25.0 Biobase_2.55.0
[11] GenomicRanges_1.47.1        GenomeInfoDb_1.31.0
[13] IRanges_2.29.0              S4Vectors_0.33.0
[15] BiocGenerics_0.41.0         MatrixGenerics_1.7.0
[17] matrixStats_0.61.0          genomicInstability_1.1.0
[19] checkmate_2.0.0

loaded via a namespace (and not attached):
 [1] mixtools_1.2.0               httr_1.4.2
 [3] tidyr_1.1.4                  bit64_4.0.5
 [5] splines_4.2.0                shiny_1.7.1
 [7] assertthat_0.2.1             interactiveDisplayBase_1.33.0
 [9] BiocManager_1.30.16          blob_1.2.2
[11] GenomeInfoDbData_1.2.7       yaml_2.2.1
[13] BiocVersion_3.15.0           pillar_1.6.4
[15] RSQLite_2.2.8                backports_1.3.0
[17] lattice_0.20-45              limma_3.51.0
[19] glue_1.4.2                   digest_0.6.28
[21] promises_1.2.0.1             XVector_0.35.0
[23] plyr_1.8.6                   htmltools_0.5.2
[25] httpuv_1.6.3                 Matrix_1.3-4
[27] pkgconfig_2.0.3              zlibbioc_1.41.0
[29] purrr_0.3.4                  xtable_1.8-4
[31] later_1.3.0                  tzdb_0.2.0
[33] tibble_3.1.5                 KEGGREST_1.35.0
```

```
[35] generics_0.1.1               ellipsis_0.3.2
[37] cachem_1.0.6                 withr_2.4.2
[39] survival_3.2-13              magrittr_2.0.1
[41] crayon_1.4.1                 mime_0.12
[43] memoise_2.0.0                fansi_0.5.0
[45] MASS_7.3-54                  segmented_1.3-4
[47] xml2_1.3.2                   data.table_1.14.2
[49] tools_4.2.0                  hms_1.1.1
[51] lifecycle_1.0.1              kernlab_0.9-29
[53] DelayedArray_0.21.0          AnnotationDbi_1.57.0
[55] Biostrings_2.63.0            compiler_4.2.0
[57] rlang_0.4.12                 grid_4.2.0
[59] RCurl_1.98-1.5               rstudioapi_0.13
[61] rappdirs_0.3.3               bitops_1.0-7
[63] DBI_1.1.1                    curl_4.3.2
[65] R6_2.5.1                     dplyr_1.0.7
[67] fastmap_1.1.0                bit_4.0.4
[69] utf8_1.2.2                   filelock_1.0.2
[71] readr_2.0.2                  Rcpp_1.0.7
[73] vctrs_0.3.8                  png_0.1-7
[75] tidyselect_1.1.1
```

# References


[1] Hanahan, D. & Weinberg, R. A. Hallmarks of cancer: the next generation. Cell 144, 646–674 (2011).

[2] Patel, A. P. et al. Single-cell RNA-seq highlights intratumoral heterogeneity in primary glioblastoma. Science 344, 1396–1401 (2014).

[3] Puram, S. V. et al. Single-Cell Transcriptomic Analysis of Primary and Metastatic Tumor Ecosystems in Head and Neck Cancer. Cell 171, 1611–1624 (2017).

[4] Muller, S. & Diaz, A. Single-Cell mRNA Sequencing in Cancer Research: Integrating the Genomic Fingerprint. Front. Genet. 8, 73 (2017).

[5] Fan, J., Slowikowski, K. & Zhang, F. Single-cell transcriptomics in cancer: computational challenges and opportunities. Exp. Mol. Med. 52, 1452–1465 (2020).

[6] Fan, J. et al. Linking transcriptional and genetic tumor heterogeneity through allele analysis of single-cell RNA-seq data. Genome Res. 28, 1217–1227 (2018).

[7] Serin Harmanci, A., Harmanci, A. O. & Zhou, X. CaSpER identifies and visualizes CNV events by integrative analysis of single-cell or bulk RNA-sequencing data. Nat. Commun. 11, 89, doi:10.1038/s41467-019-13779-x (2020)

[8] Alvarez, M. J. et al. Functional characterization of somatic mutations in cancer using network-based inference of protein activity. Nat. Genet. 48, 838–47 (2016).