Longitudinal Analysis of Cancer Evolution with LACE

Daniele Ramazzotti[1]***, Fabrizio Angaroni***[2]***, Davide Maspero***[2,3]***, Gianluca Ascolani***[2]***, Isabella Castiglioni***[4,5]***, Rocco Piazza***[1]***, Marco Antoniotti***[2]***, and Alex Graudenzi***[5]

[1]School of Medicine and Surgery, Univ. of Milan-Bicocca, Monza, Italy.
[2]Dept. of Informatics, Systems and Communication, Univ. of Milan-Bicocca, Milan, Italy.
[3]Fondazione IRCCS Istituto Nazionale dei Tumori, Milan, Italy.
[4]Department of Physics "Giuseppe Occhialini", Univ. of Milan-Bicocca, Milan, Italy.
[5]Inst. of Molecular Bioimaging and Physiology, Consiglio Nazionale delle Ricerche (IBFM-CNR), Segrate, Milan, Italy.

November 1, 2022

Overview. LACE (Longitudinal Analysis of Cancer Evolution) is an algorithmic framework that processes single-cell somatic mutation profiles from cancer samples collected at different time points and in distinct experimental settings, in order to describe the evolutionary history of the tumor. The tool can return an high resolution picture of clones' prevalence and their varietions, e.g., because of therapies.

LACE can be employed to process single-cell mutational profiles as generated by calling variants from the increasingly available scRNA-seq data, such as the ones obtained by SMARTseq2 protocol.

The output of the method is a longitudinal tree that best fits the input data, modelling both phylogenetic constraints and sc-RNAseq specific noise. Moreover, the package provides a suite of functions to visualize and explore the results.

In this vignette, we give an overview of the package by presenting its main functions.

Contents

1 Using the LACE R package

We now present an example of longitudinal analysis of cancer evolution with LACE using single-cell data obtained from Rambow, Florian, et al. "Toward minimal residual disease-directed therapy in melanoma." Cell 174.4 (2018): 843-855. The data comprises point mutations for four time points: (1) before treatment, (2) 4 days treatment, (3) 28 days treatment and finally (4) 57 days treatment.

We first load the data.

```
library("LACE")
data(longitudinal_sc_variants)
names(longitudinal_sc_variants)

## [1] "T1_before_treatment"  "T2_4_days_treatment"  "T3_28_days_treatment"
## [4] "T4_57_days_treatment"
```

The input data (D) can be either a list as shown above or a SummarizedExperiment object. In the latter case, the object needs to include genomic data (0: mutation absent, 1: mutation detected or NA: missing information) in assay field. Following guidelines, the rows of such matrix are variants, while colums are single cells concatenated for all the time points. Furthermore, a rowData field needs to be specified and must be a data.frame with a column named "TimePoint" reporting the label of the experiment for each single cell; we highlight that the ordering of the labels in rowData must match the one of the columns in the assay field. We next show an example.

```
library("SummarizedExperiment",attach.required=FALSE)
T1 = t(longitudinal_sc_variants[[1]])
T2 = t(longitudinal_sc_variants[[2]])
T3 = t(longitudinal_sc_variants[[3]])
T4 = t(longitudinal_sc_variants[[4]])
concat_time_point = cbind(T1,T2,T3,T4)
TimePointLabes = c(rep("T1", ncol(T1)),
                   rep("T2", ncol(T2)),
                   rep("T3", ncol(T3)),
                   rep("T4", ncol(T4)))
longitudinal_SE = SummarizedExperiment(assays = concat_time_point,
                                       colData = data.frame(TimePoint = TimePointLabes))
print(longitudinal_SE)

## class: SummarizedExperiment
## dim: 6 475
## metadata(0):
## assays(1): ''
## rownames(6): ARPC2_2_218249894_C_T CCT8_21_29063389_G_A ...
##   PRAME_22_22551005_T_A RPL5_1_92837514_C_G
## rowData names(0):
## colnames(475): SRR7424153 SRR7424154 ... SRR7424823 SRR7424824
## colData names(1): TimePoint
```

We setup the main parameter in oder to perform the inference. First of all, as the three data proint may potentially provide sequencing for an unbalanced number of cells, we weight each time point as follow $w_s = (1 - \frac{n_s}{n_T})/(y-1)$ in order to account for this. In the formula, e.g., the weigth for time point s (w_s) is calculated based on the number of cell observed in the time point (n_s) and the total number of cells in the threee time points (n_T). The denominator $(y-1$, with y being the numnber of time points, i.e., 3 in our case) aims at normalizing the weights to sum to one.

```
lik_weights = c(0.2308772,0.2554386,0.2701754,0.2435088)
```

The second main parameter to be defined as input is represented by the false positive and false negative error rates, i.e., alpha and beta. We can specify a different rate per time point as a list of rates. When multiple set of rates are provided, LACE performs a grid search in order to estimate the best set of error rates.

```
alpha = list()
alpha[[1]] = c(0.02,0.01,0.01,0.01)
alpha[[2]] = c(0.10,0.05,0.05,0.05)
beta = list()
beta[[1]] = c(0.10,0.05,0.05,0.05)
beta[[2]] = c(0.10,0.05,0.05,0.05)
head(alpha)

## [[1]]
## [1] 0.02 0.01 0.01 0.01
##
## [[2]]
## [1] 0.10 0.05 0.05 0.05

head(beta)

## [[1]]
## [1] 0.10 0.05 0.05 0.05
##
## [[2]]
## [1] 0.10 0.05 0.05 0.05
```

We can now perform the inference as follow. Notice that D can be either the list longitudinal sc variants or the SummarizedExperiment longitudinal SE.

```
inference = LACE(D = longitudinal_sc_variants,
                 lik_w = lik_weights,
                 alpha = alpha,
                 beta = beta,
                 keep_equivalent = TRUE,
                 num_rs = 5,
                 num_iter = 10,
                 n_try_bs = 5,
                 num_processes = NA,
                 seed = 12345,
                 verbose = FALSE)
```

We notice that the inference resulting on the command above should be considered only as an example; the parameters num rs, num iter and n try bs representing the number of steps perfomed during the inference are downscaled to reduce execution time. We refer to the Manual for discussion on default values. We provide within the package results of inferences performed with correct parameters as RData.

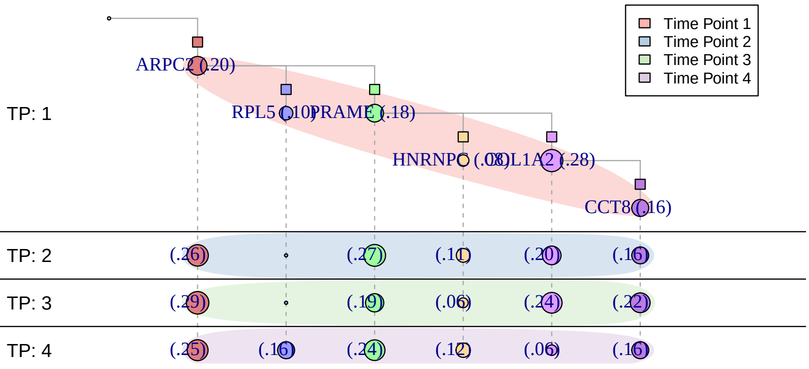
```
data(inference)
print(names(inference))

## [1] "B"                     "C"                     "corrected_genotypes"
## [4] "clones_prevalence"     "relative_likelihoods" "joint_likelihood"
## [7] "clones_summary"        "equivalent_solutions" "error_rates"
```

LACE returns a list of nine elements as results. Namely, B and C provide respectively the maximum likelihood longitudinal tree and cells attachments; corrected genotypes the corrected genotypes, clones prevalence, the estimated prevalence of any observed clone; relative likelihoods and joint likelihood the estimated likelihoods for each time point and the weighted likelihood; clones summary provide a summary of association of mutations to clones. In equivalent solutions, solutions (B and C) with likelihood equivalent to the best solution are returned; notice that in the example we disabled this feature by setting equivalent solutions parameter to FALSE. Finally, error rates provide the best error rates (alpha and beta) as estimated by the grid search.

We can plot the inferred model using the function longitudinal.tree.plot.

```
clone_labels = c("ARPC2","PRAME","HNRNPC","COL1A2","RPL5","CCT8")
longitudinal.tree = longitudinal.tree.plot(inference = inference,
                                           labels_show = "clones",
                                           clone_labels = clone_labels,
                                           legend_position = "topright")
```



2 `sessionInfo()`

- R version 4.2.1 (2022-06-23), `x86_64-pc-linux-gnu`

- Locale: LC_CTYPE=en_US.UTF-8, LC_NUMERIC=C, LC_TIME=en_GB, 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 20.04.5 LTS
- Matrix products: default
- BLAS: /home/biocbuild/bbs-3.16-bioc/R/lib/libRblas.so
- LAPACK: /home/biocbuild/bbs-3.16-bioc/R/lib/libRlapack.so
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: LACE 2.2.0, SummarizedExperiment 1.28.0, knitr 1.40
- Loaded via a namespace (and not attached): AnnotationDbi 1.60.0, Biobase 2.58.0, BiocFileCache 2.6.0, BiocGenerics 0.44.0, BiocManager 1.30.19, BiocStyle 2.26.0, Biostrings 2.66.0, DBI 1.1.3, DT 0.26, DelayedArray 0.24.0, GenomeInfoDb 1.34.0, GenomeInfoDbData 1.2.9, GenomicRanges 1.50.0, IRanges 2.32.0, KEGGREST 1.38.0, Matrix 1.5-1, MatrixGenerics 1.10.0, R6 2.5.1, RColorBrewer 1.1-3, RCurl 1.98-1.9, RSQLite 2.2.18, Rcpp 1.0.9, RcppTOML 0.1.7, RcppZiggurat 0.1.6, Rfast 2.0.6, S4Vectors 0.36.0, XML 3.99-0.12, XVector 0.38.0, assertthat 0.2.1, biomaRt 2.54.0, bit 4.0.4, bit64 4.0.5, bitops 1.0-7, blob 1.2.3, bsplus 0.1.3, cachem 1.0.6, callr 3.7.2, cli 3.4.1, codetools 0.2-18, compiler 4.2.1, configr 0.3.5, crayon 1.5.2, curl 4.3.3, data.table 1.14.4, data.tree 1.0.0, dbplyr 2.2.1, digest 0.6.30, doParallel 1.0.17, dplyr 1.0.10, ellipsis 0.3.2, evaluate 0.17, fansi 1.0.3, fastmap 1.1.0, filelock 1.0.2, foreach 1.5.2, fs 1.5.2, generics 0.1.3, glue 1.6.2, grid 4.2.1, highr 0.9, hms 1.1.2, htmltools 0.5.3, htmlwidgets 1.5.4, httpuv 1.6.6, httr 1.4.4, igraph 1.3.5, ini 0.3.1, iterators 1.0.14, jsonlite 1.8.3, later 1.3.0, lattice 0.20-45, learnr 0.11.1, lifecycle 1.0.3, logr 1.3.3, lubridate 1.8.0, magrittr 2.0.3, markdown 1.3, matrixStats 0.62.0, memoise 2.0.1, mime 0.12, parallel 4.2.1, pillar 1.8.1, pkgconfig 2.0.3, png 0.1-7, prettyunits 1.1.1, processx 3.8.0, progress 1.2.2, promises 1.2.0.1, ps 1.7.2, purrr 0.3.5, rappdirs 0.3.3, readr 2.1.3, rlang 1.0.6, rmarkdown 2.17, rprojroot 2.0.3, shiny 1.7.3, shinyBS 0.61.1, shinyFiles 0.9.3, shinydashboard 0.7.2, shinyjs 2.1.0, shinythemes 1.2.0, shinyvalidate 0.1.2, sortable 0.4.6, stats4 4.2.1, stringi 1.7.8, stringr 1.4.1, this.path 1.0.1, tibble 3.1.8, tidyr 1.2.1, tidyselect 1.2.0, tools 4.2.1, tzdb 0.3.0, utf8 1.2.2, vctrs 0.5.0, withr 2.5.0, xfun 0.34, xml2 1.3.3, xtable 1.8-4, yaml 2.3.6, zlibbioc 1.44.0