D3AIST: Data-Driven Detection of Atypical Interactions in Spatial Transcriptomics
Supplementary material I

This document describes the first part of the software implemented for processing the data of experiments in spatial transcriptomics.

It includes the explanation and documentation of three programs written in C++ and usable from the command line that allow the preprocessing of all the data from the original .csv files and the generation of data to feed the R part of the software and to generate images to illustrate the results.

The main steps are:

1. Transform the data in .csv to a more compact binary format that can be read faster. This is done by program `csv2bin`. This step is not compulsory: the next step can read the data directly from the .csv files, but since it is likely that the user wants to run the second step several times to assess the behaviour of different genes or groups of them, having the initial data in binary format makes the load much faster. This is particularly important for the transcription events whose number is typically of several million. Program `csv2bin` runs as:

```
$ csv2bin <object_type> <file> [<cell_bound_file>] [<nucleus_bound_file>] <bin_file>
 where object_type is:
  'p' to convert a gene panel,
  'c' to convert a cell set,
  'b' to convert a cell set plus a cell boundary,
  'n' to convert a cell set plus a nucleus boundary,
  'a' to convert a cell set, a cell boundary and a nucleus boundary,
  't' to convert a transcript set,
  'w' to convert a transcript set without codewords (needs a gene panel file as second argument
 Arguments between [ ] are required for the appropriate types (b,n or a) according to type.
 The resulting binary file must be always the last argument.
```

   All the files except the binary output are assumed to be .csv. Therefore, typical calls for one of the cases, as an example for patient 1 whose data could be in folder P1_CRC_Add_on_FFPE_outs, would be:

```
csv2bin p GSE280314_gene_panel.csv GSE280314_gene_panel.bin
csv2bin a P1_CRC_Add_on_FFPE_outs/cell.csv P1_CRC_Add_on_FFPE_outs/cell_boundaries.csv \
          P1_CRC_Add_on_FFPE_outs/nucleus_boundaries.csv P1_CRC_Add_on_FFPE_outs/cell.bin
csv2bin t P1_CRC_Add_on_FFPE_outs/transcripts.csv P1_CRC_Add_on_FFPE_outs/transcripts.bin
```

2. Generate the information from a given case, a requested set of genes and a desired area of the image. The results are:

   - A collection of .csv files to be loaded as tables in R with the location ((x, y) coordinates) of all transcription events for any requested gene as long as additional marks like distance of event to cell nucleus, ratio of nucleus to cell area and cluster to which the cell belongs to. This is the information used to build and analyze a marked point process using the R package `spatstat`, as it will be explained in Supplementary material-II.
   - A set of binary images, one for each interesting gene, with the same size of the focus area, that will be colored and superimposed to visualize the spatial distribution

- A groovy script to extract the background image from the original, big histopathological image.
- A .csv file with a table of the requested genes and the number of transcription events for each one in the focus area.

This is done by program `buildim`. Since the number of parameters needed is high and complex, they are not passed in the command line but collected in a configuration file in the form
`'Key: value'`
The name of this configuration file is the only argument to the program.
`buildim` works simply as

```
$ buildim
Usage:
   buildim <task_desc_file>
```

being a typical call
`buildim P1Ac.desc`
The keys and valid values for the configuration file, together with an example, will be explained later.

3. Construct images where each transcription event is represented as a point with a color different for each gene, superimposed to the background image of the tissue preparation. This is done by program `imfuse`. Again, it need the names of all binary images generated in the former step and the colors to assign to each one, which is again collected in a configuration file, automatically generated from the former step. It also asks for the name of the final generated image and optionally an option to apply a convex or concave curve of gamma correction to the background image, to improve quality of visualization. `imfuse` works as

   ```
   Usage:
      imfuse <imagelist_fusion_file> <fusion_image> [-adjcup|-adjcdown]
   ```

   being a typical call

   ```
   imfuse P1Ac.fuse P1Ac_genes_set_1.tif -adjcdown
   ```

   File `P1Ac.fuse` was generated by `buildim` in step 2, with a typical content like

   ```
   /home/RAID5/SP_TRANS/P1AcImages/Case1Ac_backg.tif
   /home/RAID5/SP_TRANS/P1AcImages/Case1Ac_423_STAB1.tif 0 255 0
   /home/RAID5/SP_TRANS/P1AcImages/Case1Ac_458_REG1A.tif 139 79 12
   /home/RAID5/SP_TRANS/P1AcImages/Case1Ac_277_LCN2.tif 0 255 255
   ```

   where the three numbers after each image are the R,G,B components of the color assigned to each gene.

   Background image must be extracted by the user and stored with the indicated name in the correct folder in any way he/she decides. It MUST have the same width and height of the binary images; otherwise, superposition will not be possible.

   For QuPath users, the program `buildim` generates a groovy script to be applied to a one-layer histopathological image to extract the background. The sequence is:

- Open QuPath, load the big image with layers and export it to a one-layer image using the QuPath menu `File --> export Images --> OME TIFF`. Appropriate values for export options can be `Compression Type: JPEG-2000 (lossless)`, `Pyramidal downsample: 8`, `Tile Size: 4096`, tick the box Parallelize export and **untick** the box `All z-slices`. Save the result with any name.
- Open again QuPath, open the recently saved image and with menu `Automate --> User scripts` load the groovy script generated by `buildim` and make it run. The background image will be generated with the appropriate name in the correct folder

Finally, the keys and values for the configuration file of step 2 are as follows. Parameters marked with * are compulsory. Others may be omitted and will take its default value.

PrependPath: Path to be prepended to all file names from now on, except if they are absolute paths (i.e.: start with /). If given, this parameter must hold and absolute path, so it must start with /. Default value if using `none` or omitting the parameter is the empty string.

Genes*: The csv/bin file containing the panel of used genes.

Cells*: The csv/bin file containing the list of detected cells. Using here a .bin file generated with `csv2bin` using the 'a' mode and the files of cell and nucleus boundaries will also load implicitly such information.

Transcripts*: The csv/bin file containing the localized expressions of the transcripts.

Clusters: The csv/bin file containing the assigned classifications (clusters) of each cell. Default value: `none` (don't use this file).

CellBoundaries: The csv file with the cell boundaries. Omit or use `none` to ignore this information. Don't use it if it is contained inside the .bin file of cells (see parameter **Cells**).

NucBoundaries: The csv file with the nucleus boundaries. Omit or use `none` to ignore this information. Don't use it if it is contained inside the .bin file of cells (see parameter **Cells**).

Spacing*: Pixel spacing in micrometers along x (width) and y (height) directions, respectively. Two possitive real numbers.

Dimensions*: Dimensions in pixels of the complete image. Two possitive integer numbers as width height in that order.

PixPerBin*: How many pixels are grouped together as a single pixel (for image generation) or single location (for location-table generation). Integer number. It is taken as the same value for horizontal and vertical dimension.

AoIWin$^{(alt)}$**:** Window of interest as a rectangular region. It expects four real possitive numbers: (upper_left_x upper_left_y) (lower_right_x lower_rigth_y) all in micrometers

AoIID$^{(alt)}$**:** Window of interest as a region identifier like B9, AB12, etc.
$^{(alt)}$ **means that exactly one of the parameters AoIWin or AoIID must be used, but not both.**

QvThres*: Threshold on the quality value qv (which is in 0..40) to consider and select a transcription event

OnlyTargets: Should we restrict the analysis to genes marked as targets in the gene panel or any gene is allowed? (possible values: true or false, Default value: false)

TargetIIds$^{(alt)}$**:** a list of comma-sepparated integer numbers with the codewords of the genes the user is interested in. All numbers must be in the same line (no carriage-return allowed).

TargetNames$^{(alt)}$**:** a list of comma-sepparated strings with the gene names. All strings must be in the same line (no carriage-return allowed)
$^{(alt)}$ **means that exactly one of the parameters TargetIds or TargetNames must be used, but not both.**

GenMarks: Should associated marks be generated? (exclusively in the R files). Marks are: Nucleus distance (0 for transcript inside nucleus, the distance to nucleus for transcripts inside cell and NA for transcripts outside any cell), ratio Cell_nucleus_area/cell_area of the cell with the transcript or NA for transcripts outside the cell. Valid values for this parameter are only true or false. Default value: false.

OutRfile: Base name of the .csv location files to be generated. If you don't want to generate them use `none` as its name or omit the parameter. Names of files will be this base name with 000, 001, etc. appended and with extension .csv

OutImfile: Base name of the image file to be generated. If you don't want to generate them use value `none` or omit it. Names of image files will be this name plus GeneName_GeneCode appended and with extension .tif

ColorFile: Name of the file with the color assignment to each gene. It must be a .csv file with Genecode,GeneName,R,G,B in each line, where R,G and B are integer numbers in 0..255. Use `none` or omit it if you have not asked for the generation of image files

OutHistFile: Name of a .csv file with the gene name and number of event transcription counts of each gene inside the area (only for events with quality bigger than the quality threshold, QvThres). Use `none` or omit it if you don't need to generate this file

CheckThisFile: Boolean value (true/false) to indicate that we want to check the syntax and parameters of this file before execution. Default value is false. Use true just to check before launch the program and set it to false or omit it afterwards.

A typical example of configuration file is as follows:

```
# Parameters of this file are those whose names are specified. You must use
# all of them with exactly that name. Order of lines/parameters does not matter.
# Lines whose first character is # are comments.
# In that case, all line is considered a comment.
#
PrependPath: /home/RAID5/SP_TRANS/
Genes: GSE280314_gene_panel.bin
Cells: P1_CRC_Add_on_FFPE_outs/cells.bin
Transcripts: P1_CRC_Add_on_FFPE_outs/transcripts.bin
Clusters: P1_CRC_Add_on_FFPE_outs/analysis/clustering/gene_expression_graphclust/clusters.csv
# The csv file containing the cell boundaries. It is commented since we have used a cells.bin file
# CellBoundaries: COL_CAN_DATA/P1_CRC_Add_on_FFPE_outs/cell_boundaries.csv
# The csv file containing the nucleus. It is commented since we have used a cells.bin file
# NucBoundaries: COL_CAN_DATA/P1_CRC_Add_on_FFPE_outs/nucleus_boundaries.csv
Spacing: 0.2125 0.2125
Dimensions: 31345 34111
PixPerBin: 1
AoIWin: 4733.2250 3113.5500 4989.9250 3387.4625
# If the area is one of the predefined areas, a line like
# AoIId: B0
# would be correct.
QvThres: 30
# It is equivalent to use OnlyTargets: false or omit it
TargetNames: STAB1,REG1A,LCN2
# Alternatively, codewords can be used:
# TargetIds: 423,458,277
GenMarks: true
OutRfile: P1AcLocations/Case1Ac_.csv
OutImfile: P1AcImages/Case1Ac_.tif
ColorFile: coltable.csv
OutHistfile: P1AcImages/P1Ac.hist
```

A typical use of the software involves generally these phases:

- Execution of step 1 only once to generate the binary files.
- Execution of step 2 for all or most of the target genes. Then, the R part of the software (see Supplementary material-II) is applied to obtain the genes with atypical behaviour and their graphs representing their mutual spatial interactions in terms of attraction/repulsion. This leads to a selection of interesting genes.
- Execution of step 2 for the interesting genes
- Execution of step 3 to obtain the graphical results. These are the images that together with the gene interaction graphs, must be biologically interpreted. These last two phases can be repeated with different genes to refine the analysis.

Computational resources required for the programs in C++ are quite limited. Execution time is never higher than a few minutes in a typical Intel/AMD modern processor at clock frequency of 4GHz.

The programs in R, specifically the evaluation of the K-function using the `spatstat` package, are relatively slow for very big areas or even the whole image.