Longitudinal Analysis of Midgut

LAM v1.0

User Manual

Arto I. Viitanen

Hietakangas Laboratory

University of Helsinki

Table of Contents

[1 Description 3](#_Toc25059606)

[2 Installation of Dependencies 3](#_Toc25059607)

[3 Usage 3](#_Toc25059608)

[3.1. Limitations in vector creation 3](#_Toc25059609)

[3.2. Input 3](#_Toc25059610)

[3.2.1. File organization and naming 3](#_Toc25059611)

[3.3. Step-by-Step 3](#_Toc25059612)

[3.3.1. Primary settings 4](#_Toc25059613)

[3.3.2. Data gathering 4](#_Toc25059614)

[3.3.3. Vector creation 4](#_Toc25059615)

[3.3.4. Distance calculations 4](#_Toc25059616)

[3.3.5. Statistics 4](#_Toc25059617)

[3.3.6. Plotting 4](#_Toc25059618)

[4 Output Files 4](#_Toc25059619)

[4.1. Plots 4](#_Toc25059620)

[4.2. Data files 4](#_Toc25059621)

[5 Troubleshoot 4](#_Toc25059622)

# Description

**Notification:**

While the positional input data that LAM accepts can indicate any sort of feature that is to be counted and analyzed, **for the purpose of the user manual the data will be referred to as cells.** LAM also accepts multiple data folders for each sample, and while each of these folders does not necessarily contain data from one microscopy channel, **these separate data will be referred to as channels for simplicity.** Microscopy channels will be referred to as such.

# Installation of Dependencies

1. Install Anaconda3 distribution (<https://www.anaconda.com/distribution>/)

2. Add Shapely-package:

Windows:

Get Shapely .whl from [https://www.lfd.uci.edu/~gohlke/pythonlibs/#shapely](https://www.lfd.uci.edu/~gohlke/pythonlibs/%23shapely)

Then write following command(s) in Anaconda prompt:

(0.) pip install wheel (should be included in Anaconda)

1. pip install <path-to-the-downloaded-whl-file>

OS X & Linux:

Open Anaconda prompt and write following command:

pip install shapely

3. Add pycg3d-package:

Open Anaconda Prompt and write command:

pip install pycg3d

# 

# Usage

1. Organizing files for input (user)
2. Creation of sample-specific vectors (automated or user)
3. Gathering of data and projection onto vector
4. Calculation of cell numbers and additional data
5. Finding nearest cells
6. Finding clusters
7. Calculation of statistics
8. Plotting

## Input

The main functionality of LAM, i.e. the locational quantification of cells, requires only X- and Y-coordinates of cells, found within a csv-file with data on columns and each cell on separate row. For finding nearest cells and clusters, the Z-coordinate of the cells is also required. Any additional data that is to be analyzed has to be defined in the settings, and can either be located within the same csv-file with the positional data or separately. All samples are not expected to have data on all channels; LAM only adds the data to analysis if it is found.

The vectors are created based on the positional data of the ‘vector channel’, defined by the given value in Vector/Channel –option (vectChannel in settings.py). The creation begins from cells with the lowest X-coordinates, and consequently all samples are expected to be oriented the same in the coordinate system. In case of wrongly oriented samples, a coordinate system-rotation script (rotator.py) can be found in the ‘Companions’-folder of the LAM -master folder.

On some experiments the size proportions of different regions may alter, e.g. when comparing starved and fully-fed midguts. In these cases more accurate results can be obtained by dividing the image/data into multiple analyses. A typical way to do this is to run separate analyses for R1-2, R3, and R4-5. Alternatively, a user-defined coordinate (MP = measurement point) at a distinguishable point can be used to anchor the individual samples for comparison, e.g. points at R2-3-border are lined, with each sample having variable numbers of bins on either side. The variation however likely leads to a compounding error as distance from the MP grows. When MP is not used, the samples are lined at bin 0, and compared bin-by-bin. The MP-input is done similarly to channel data, i.e. as a separate directory that contains position.csv for a single coordinate, the MP.

## File organization and naming

**Fig. 1. Analysis directory hierarchy.**

LAM expects to find all input data in a certain hierarchy of directories and with specific filenames that contain necessary information (Fig. 1). As LAM scans the directory for files and folders without user-defined samples, sample groups, or channels, the recommendation is not to have any unrelated files or folders within the input data-directory, as this can break the analysis. LAM automatically gathers the names of the samples, groups, and channel-names from the paths of the files.

The full naming convention for the paths in the analysis is as follows:

<group>\_<descriptor>\_<sample>\_<channel>\_xyz,

where group denotes the sample group, descriptor and sample are used to identify individual samples (e.g. date of imaging and sample name), and channel is an identification for a data folder. The “xyz” at the end can be any string of text and is not used by LAM; it is present for convenience, as e.g. Imaris-exported data includes “\_Statistics” at the end.

Within the analysis directory, the files and folders for the analysis must be organized in the manner shown by Fig. 1. All samples should be located at the root of the directory in separate folders named as <group>\_<descriptor>\_<sample>, e.g. Control\_09-06-2019\_sample1. Within these sample folders, each sample should have a separate channel folder for each channel it has data for, named with the before mentioned name extended with <channel>\_xyz, e.g. Control\_09-06-2019\_sample1\_DAPI\_Stats. Each data file that relates to the specific channel and sample must be found within these channel folders.

**Underscore ‘\_’ is used as a delimiter for information, and is consequently reserved for LAM.** Any use of underscore by the user will likely interfere with the analysis. Naming should be restricted to letters and numbers.

## Data files

## Primary settings

**Fig. 2. LAM graphical user interface.**

## Default values

The default values are stored in settings.py.

## Data gathering

## Vector creation

The vectors for the samples can be created in two ways: by a running median, or by skeletonization. Both methods have their own benefits and drawbacks discussed in chapter 3.4.1. The whole analysis can be performed in one run, however the best and most accurate results can typically be obtained by subjecting the samples to multiple rounds of vector creation and then passing only the well-fitting vectors to further analysis.

A typical workflow starts by only selecting the ‘Process’-setting, and optionally ‘Use MP’, to create vectors for all of the samples. The vectorization can be run in multiple steps: the quality of each sample’s vector can be verified from the vector plots located at ‘./Analysis Data/Samples’, and when deemed fit, the respective LAM-created sample folders containing the vector.csv-files can be collected to be used later. Next, vector creation can be performed with other settings, again collecting acceptable vectors. Any unfit vectors can also be directly modified by changing any offending coordinates or by adding new ones to the vector.csv. Alternatively, the vector.csv could be entirely generated by the user.

Once all of the vectors are created, the collected sample folders can be transferred back to ‘./Analysis Data/Samples’, overwriting any non-usable data.

## Vector types

## User-created vectors

When opting to use self-created vectors, it should be noted that the first coordinate in the file is considered to be the beginning of the first bin (bin zero) of the vector. Consequently, all vectors must be given in the same orientation, e.g. from anterior to posterior end of the sample.

## Distance calculations

## Statistics

## Plotting

# Output Files

## Plots

## Data files

# Troubleshoot