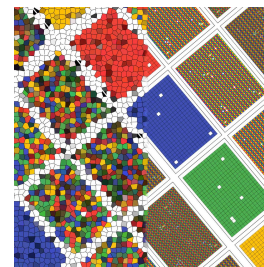


Feb 24, 2023

LI Detector Analytical Pipeline

DOI

<https://dx.doi.org/10.17504/protocols.io.3byl4kjd2vo5/v1>



Saurin B Parikh¹

¹University of Pittsburgh



Saurin B Parikh

University of Pittsburgh

Create & collaborate more with a free account

Edit and publish protocols, collaborate in communities, share insights through comments, and track progress with run records.

Create free account

OPEN  ACCESS



DOI: <https://dx.doi.org/10.17504/protocols.io.3byl4kjd2vo5/v1>

External link: <https://doi.org/10.1093/g3journal/jkaa068>

Protocol Citation: Saurin B Parikh 2023. LI Detector Analytical Pipeline. **protocols.io**

<https://dx.doi.org/10.17504/protocols.io.3byl4kjd2vo5/v1>



Manuscript citation:

Parikh, S. B., Castilho Coelho, N., & Carvunis, A.-R. (2021). LI Detector: a framework for sensitive colony-based screens regardless of the distribution of fitness effects. *G3 Genes|Genomes|Genetics*, 11(2). <https://doi.org/10.1093/g3journal/jkaa068>

License: This is an open access protocol distributed under the terms of the **Creative Commons Attribution License**, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited

Protocol status: Working

We use this protocol and it's working

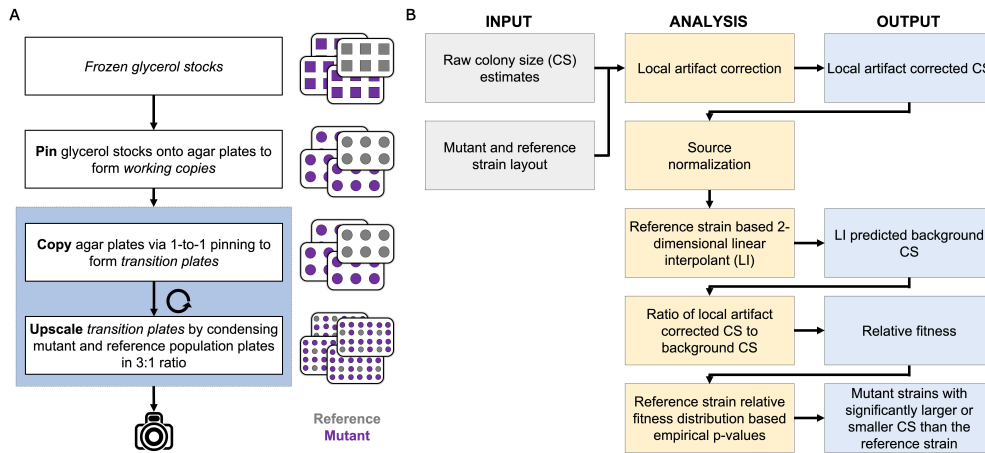
Created: November 19, 2020

Last Modified: February 24, 2023

Protocol Integer ID: 44689

Keywords: beneficial, normalization method, genetic screen, phenomics, microbiology, yeast, genomics, high-throughput, screening, li detector analytical pipeline, li detector framework, analytical pipeline, experimental pipeline, analysis pipeline, relative fitness estimations for the mutant strain, reference strain relative fitness distribution, local artifact correction, background colony size estimation, based background colony size estimation, upscale experimental pipeline from frozen glycerol stock, population for spatial bias correction, spatial bias correction, relative fitness estimation, strain layout, background colony size, source normalization, mutant strain, relative fitness distribution, user inputs consist of raw colony size estimate, local artifact, analytical step, reference strain, corrected relative fitness

Abstract



The LI Detector framework consists of integrated experimental and analytical pipelines. A. The pin-copy-upscale experimental pipeline from frozen glycerol stocks (top) to imaging (bottom). Each box represents a pinning step, and the steps within the sky-blue highlighted portion can be repeated until the desired colony density is reached. Illustrations to the right of the flowchart is a simplified representation of four experimental plates. A reference population (grey) is introduced on every plate during the first upscale step. The analytical pipeline uses this population for spatial bias correction and relative fitness estimations for the mutant strains of interest (purple). **B.** Workflow of the analysis pipeline where columns from left to right represent user inputs, analytical steps, and outputs. User inputs consist of raw colony size estimates and the strain layout of the plates. The analytical pipeline performs: i) local artifact correction, ii) source normalization, iii) reference-based background colony size estimation using a 2-dimensional linear interpolation, iv) corrects for spatial bias by dividing the local artifact corrected colony sizes with the background colony sizes and provides a measure of relative fitness, and iv) assigns empirical p-values using the reference strain relative fitness distribution. The outputs include local artifact corrected colony sizes, background colony sizes, spatially corrected relative fitness, and mutant strains identified as having a mean colony size that is significantly larger or smaller than the reference strain.

Before start

LI Detector analytical pipeline can only be applied to experiments conducted in accordance to the LI Detector experimental pipeline. Please refer to the LI Detector manuscript for best practices on conducting the colony-based high-throughput experiment.

Files

- 1 Plate maps of the starting density plate
 - A .xlsx file with one plate per sheet
 - Cells contain strain-id
 - **Example**
- 2 Table specifying strain-id to orf-name relationship
 - A .xlsx file containing unique strain_id to each orf_name
 - First column is strain_id
 - Second column is orf_name
 - Each strain_id from Step 1 should have an associated orf_name
 - **Example**

*orf_name variable is used for names of the mutants in the experiment.

Download LID and dependencies

- 3 Dependencies:
 1. Install Database Toolbox from the APPS > Get More Apps option within MATLAB
 2. Download and unzip mysql connector JDBC driver from [here](#).
- 4 Download LID and associated scripts from Github in your MATLAB folder.

```
~$ cd MATLAB
```

```
~/MATLAB$ git clone https://github.com/sauriiin/Matlab-Colony-Analyzer-Toolkit.git
```

```
~/MATLAB$ git clone https://github.com/sauriiin/bean-matlab-toolkit.git
```

```
~/MATLAB$ git clone https://github.com/sauriiin/lidetector.git
```

```
~/MATLAB$ git clone https://github.com/sauriiin/sau-matlab-toolkit.git
```

- 5 Make LID bash scripts executable.



```
~/MATLAB$ cd lidedetector

~/MATLAB/lidedetector$ chmod +x initialize.sh
~/MATLAB/lidedetector$ chmod +x buildraw.sh
~/MATLAB/lidedetector$ chmod +x lid.sh
```

Initialize

- 6 Information to keep in hand before proceeding:
 1. MySQL credentials - username, password, database name
 2. Name of experiment - this will be used as a prefix for all the tables that will be generated
 3. Upscale patterns from the experiment - ie in what combinations were the lower density plates condensed to form the higher density plates
 4. Name (orf_name) of reference strain used
 5. File path to plate map .xlsx file from Step 1
 6. File path to the strain_id to orf_name .xlsx file from Step 2
- 7 Execute the initialize bash script from within the lidedetector folder.

```
~/MATLAB/lidedetector$ ./initialize.sh
```

Successful run will create the following tables

1. **_pos2coor** = position ids and their corresponding plate coordinate (density, plate number, column number and row number).
2. **_pos2orf_name** = position ids and the corresponding orf-name
3. **_pos2rep** = position ids of lowest density plates to their replicates at higher density plates based on the upscale pattern
4. **_pos2strain_id** = position ids and their corresponding strain ids
5. **_strainid2orf_name** = same as table from Step 2

Example files can be found in [Data.zip](#).

Colony Size Data

- 8 Organize colony size estimations from your favorite colony size estimator, like the **MATLAB Colony Analyzer Toolkit** (MCAT), in ascending order of hours, plate number, column number, row number.



Below is the structure of such a file. Here image1,2,3 are pixel counts from 3 different images of the same plate. Average column consists of the average pixel count of image1,2,3.

	A	B	C	D	E
	hours	image1	image2	image3	average

Example

- 9 Combine the above table with positions ids from **_pos2coor** table using the below command.

```
~/MATLAB/lidetector$ ./builddraw.sh
```

Successful completion of this command will generate:

1. **_RAW** = raw colony size estimations per hour per position id of all the images
2. **_smudgebox** = position ids to be excluded from analysis that correspond to the user defined coordinates
3. **_JPEG** = clean version of the raw table with border colonies, colonies corresponding to the smudge box and those colonies with pixel count of less than 10 NULL'd

Example files can be found in **Data.zip**.

- 10 Users can skip step 8 & 9 to use LI Detector's imageanalyzer function if they choose to utilize MCAT as their desired tool for colony size estimation.

Protocol



NAME

LID: imageanalyzer

CREATED BY

Saurin B Parikh

Preview

Skip this step if you have successfully executed step 8 & 9.

10.1 Photos should be organized as Experiment > Arm > Stage > Hours

- Within the Hours folder the photos should be arranged in the same order as the plate names/number
- Example
 - If you were conducting an experiment using the mutant collection
 - The experiment had two parallel arms going from 384 density plates (Starter Plates) to 1536 density plates (Pre-screen) to 6144 density plates (Final Screen)
 - Photos for the Starter Plates and Pre-screen were taken at saturation and those for Final Screen were taken at 0, 4 and 12 hours
 - Then the folder heirarchy would be as follows:
 - Experiment
 - Arm #1
 - Starter Plates
 - 36h
 - Plate 1
 - Plate 2
 - Pre-screen
 - 20h
 - Plate A
 - Plate B
 - Final Screen
 - 00h
 - 04h
 - 12h
 - Arm #2
 - Starter Plates
 - 36h
 - Pre-screen
 - 20h
 - Final Screen
 - 00h
 - 04h
 - 12h
 - If the bifurcation of the arms occur later in the experiment then the folder heirarchy could be as follows:
 - Experiment
 - Starter Plates
 - 36h
 - Plate 1
 - Plate 2

- Pre-screen
 - 20h
 - Plate A
 - Plate B
- Arm #1
 - Final Screen
 - 00h
 - 04h
 - 12h
- Arm #2
 - Final Screen
 - 00h
 - 04h
 - 12h

- Make sure the terminal folders containing the photos are names is a 'tth' manner like shown above

10.2 Image Quality Check

1. Are all the images in the right orientation? - top left corner of the plate should be at the top left corner of the image.
2. Are there the expected number of images in your folders?
3. Do all the images have a uniform black background? Any 'white' in the background will cause the MATLAB Colony Analyzer Toolkit (MCAT) to falsely select it as a colony and fail.
4. Make sure that all the images are ordered correctly.
5. Take note of any 'smudges' or problem colonies on the plates. One might consider adding these to the smudgebox when prompted.

10.3 Execute Step 1 to 7 of the LI Detector Analytical Pipeline

Note: The order of the plates in the upscale pattern from Step 6 of the pipeline should be the same as the order of images within the folders.

10.4 Information to keep in hand before proceeding:

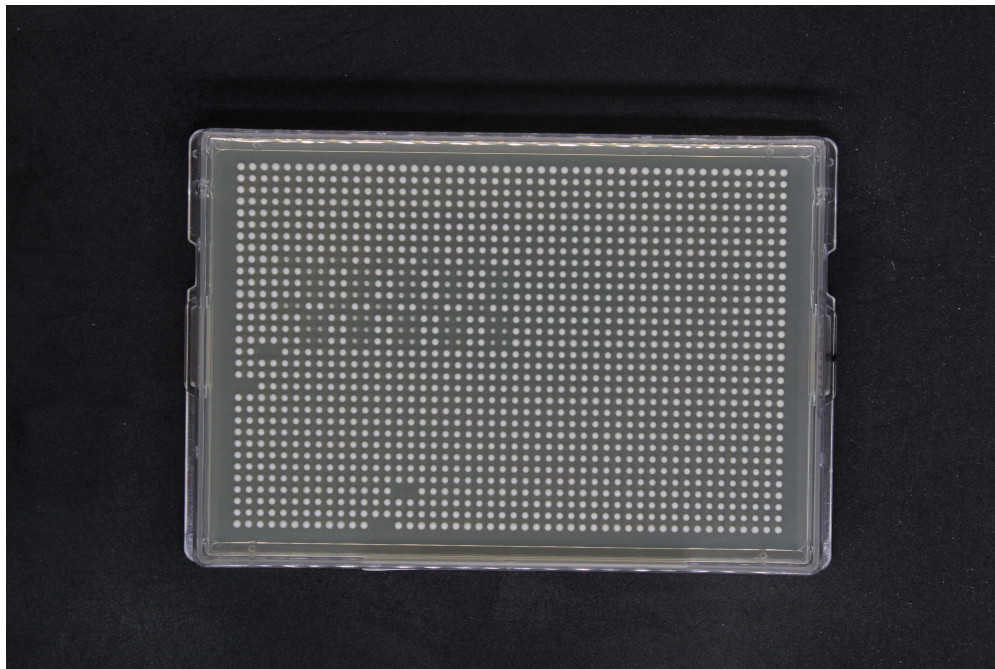
1. Path to MALTAB directory
2. Path to lidetector directory
3. Path to where the JDBC driver was unzipped from Step 3 of LI Detector Analytical Pipeline
4. Path to the 'Stage' level photos from Step 1
5. Location of any smudges on the plates ie the colonies you want to remove from the analysis because of any technical issues - plate number, row number, column number

10.5 Execute the Image Analyzer

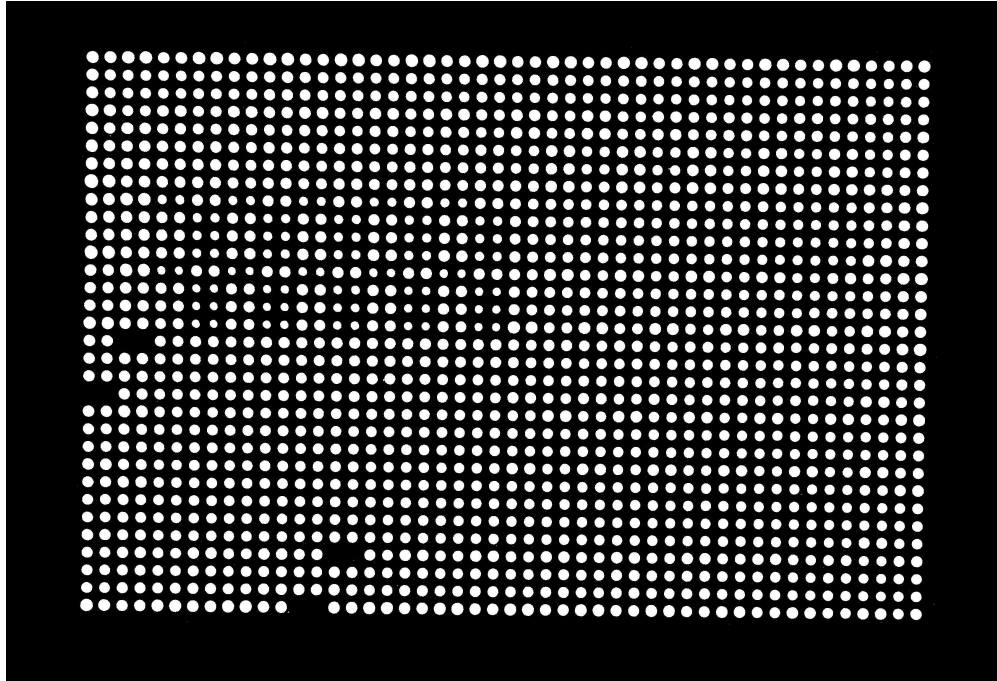
```
~/MATLAB/lidetector$ ./imageanalyzer.sh
```

10.6 User will be asked to verify binary files before uploading raw pixel count data

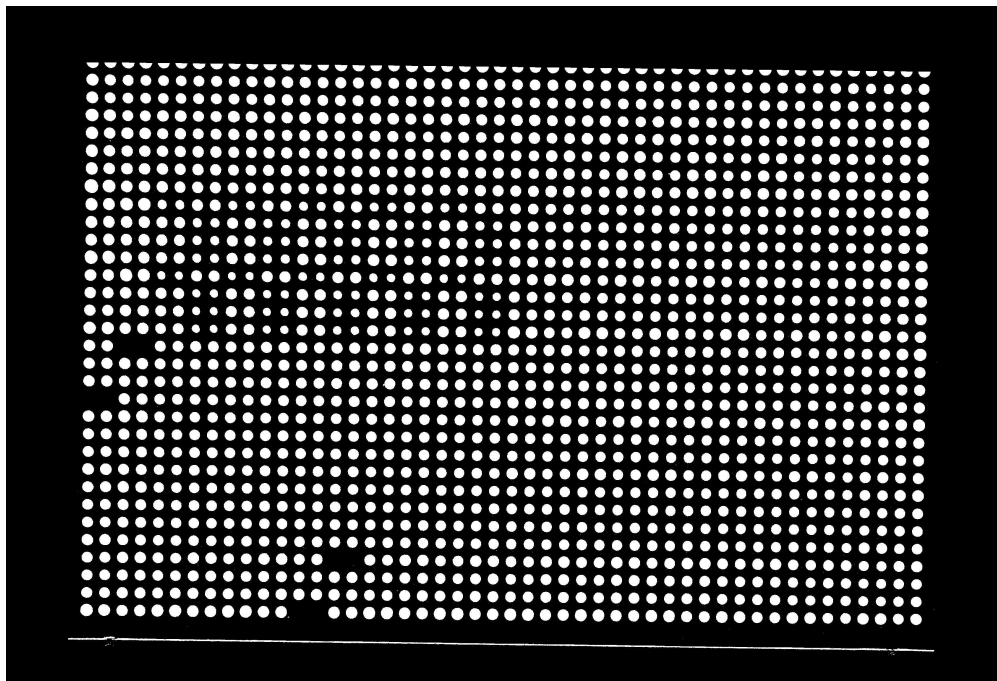
- Each image will now have 3 additional files - .binary, .cs.txt and .info.mat
- View the .binary file (using Preview in Mac) to verify if the colonies have been correctly identified
- Original image:



- Good binary image:



- Bad binary image:



- Proceed to upload only if all binary files are 'good.'

10.7 Successful completion of this command will generate:



1. **_RAW** = raw colony size estimations per hour per position id of all the images
2. **_smudgebox** = position ids to be excluded from analysis that correspond to the user defined coordinates
3. **_JPEG** = clean version of the raw table with border colonies, colonies corresponding to the smudge box and those colonies with pixel count of less than 10 NULL'd

Example files can be found in [Data.zip](#).

Spatial Bias Correction

- 11 Information to keep in handy before proceeding:
 1. Path to MALTAB directory
 2. Path to lidetector directory
 3. Path to where the JDBC driver was unzipped from Step 3
- 12 Execute the LI Detector

```
~/MATLAB/lidetector$ ./lid.sh
```

Successful run will create the following tables:

1. **_NORM**= position ids and their corresponding relative fitness measurements along with the background pixel count measurement based on references
2. **_FITNESS** = similar to **_NORM** but with strain ids and orf-names included
3. **_FITNESS_STAT**= strain-id-wise mean, median and standard deviation of relative fitness
4. **_PVALUE** = strain-id-wise empirical p-values where
$$\text{stat} = (\text{strain mean fitness} - \text{reference mean fitness}) / \text{reference fitness standard deviation}$$
$$\text{es} = (\text{strain mean fitness} - \text{reference mean fitness}) / \text{reference mean fitness}$$

Example files can be found in [Data.zip](#).