nature research

https://doi.org/10.1038/s41596-020-00465-2

Supplementary information

Scalable characterization of the PAM requirements of CRISPR–Cas enzymes using HT-PAMDA

In the format provided by the authors and unedited

Supplementary information

Supplementary Note 1. Instructions for HT-PAMDA data analysis.

Instructions for running the HT-PAMDA analysis are provided below, however the most up-to-date instructions, please refer to the GitHub page: https://github.com/kleinstiverlab/HT-PAMDA.

There are two main sections of the analysis:

1. Randomized PAM library quality control: Assess the distribution of PAM sequences in an untreated randomized library control.
2. HT-PAMDA data analysis: Analyze HT-PAMDA data to define PAM preferences of CRISPR-Cas nucleases.

Example data has been provided to run both sections of the analysis.

Randomized PAM library quality control

1. Create a barcode_csv file for the sample(s) based on the examples provided in the `barcode_csv` directory.
2. Navigate to the `code` directory and open the `library_QC_inputs.py` file in a text editor.
3. Enter the required parameters in the `library_QC_inputs.py` file, including a run name, the file path to the barcode_csv file, the directory containing all fastq files, the abbreviated name of the control fastq file, and the PAM orientation, length, and start position. There are additional parameters for custom implementations of the assay. Save the file.
4. From the virtual environment, enter the following command to run the analysis: `python3 library_QC.py`
5. The results will populate the `output` and `figures` directories.

HT-PAMDA data analysis

1. Create a barcode_csv file for the sample(s) based on the examples provided in the `barcode_csv` directory. This will require indicating the sample barcode sequences for each sample.
2. Navigate to the `code` directory and open the file `inputs.py` file in a text editor.
3. Enter the required parameters in the `inputs.py` file, including a run name, the file path to the barcode_csv file, the directory containing all fastq files, a python dictionary mapping abbreviated fastq names to timepoints, indexed from 0, and the PAM orientation, length, and start position. If the randomized PAM library quality control analysis was performed on the library used in the HT-PAMDA experiment, the file path to the PAMDA_1_raw_counts_csv.gz output file from the randomized PAM library quality control analysis should be provided as input. If the randomized PAM library quality control analysis was not performed and the control sample is included in the HT-PAMDA library, indicate the fastq containing the control sample. Enter the sample name of the control sample. There are additional parameters for custom implementations of the assay. Save the file.
4. From the virtual environment, enter the following command to run the complete HT-PAMDA analysis: `python3 PAMDA_complete.py`
5. For troubleshooting, individual steps of the analysis can also be run with the following commands. The command `python3 fastq2count.py` will generate raw read counts from fastq files. The command `python3 rawcount2normcount.py` will generate normalized read counts from raw read counts. The command `python3 normcount2rate.py` will generate rate constants from normalized read counts. The command `python3 rate2heatmap.py` will generate heatmap representations of PAM preference from rate constants.

Supplementary Note 2. Visualization of PAM preferences in Figure 4.

Each representation of PAM preference in **Fig. 4** was generated from the same HT-PAMDA dataset. For each nuclease, two replicates on each of two spacer sequences were used for a total of four replicates per nuclease. Sequence logos are representations of position weight matrices generated from sequence enrichment data. To generate an equivalent representation from HT-PAMDA data, normalized read counts (where a value of 1 indicates no targeting and a value of 0 indicates complete targeting) were converted to enrichment scores. Briefly, a position frequency matrix was generated from the normalized read counts of the

32-minute timepoint. To convert depletion to enrichment, frequencies at each base and position were subtracted from the maximum frequency observed at any base and position. Finally, frequency was normalized to sum to one at each position to generate the final position-weight matrix from which sequence logos were created using Logomaker[30]. To represent HT-PAMDA data as PAM wheels, we performed the following alterations to generate a representation of rate compatible with Krona plots: PAM wheel rate = MAX($\log_{10}(k)$, -5) + 5. PAM wheels were generated from the PAM wheel rates using Krona plots[26].

References:

26. Ondov, B. D., Bergman, N. H. & Phillippy, A. M. Interactive metagenomic visualization in a Web browser. *BMC Bioinformatics* **12**, 385 (2011).

30. Tareen, A. & Kinney, J. B. Logomaker: beautiful sequence logos in Python. *Bioinformatics* **36**, 2272–2274 (2020).