IGVF CRISPR Jamboree 2024: Perturb-seq Inference

Gene Katsevich

February 23, 2024

1 Overview

1.1 Perturb-seq inference

The goal of perturb-seq inference is to quantify the extent to which the perturbation of a given genomic element impacts the expression of a given gene. We allow a range of statistical interpretations of this task. In a frequentist framework, this task can be viewed as testing the null hypothesis that the perturbation of the genomic element has no effect on the gene's expression, or as estimating the effect size of the perturbation on the gene's expression. In a Bayesian framework, this task can be viewed as estimating the posterior probability of the presence of a non-zero effect, or as a posterior mean of the effect size.

1.2 Jamboree goals

The goal of the perturb-seq inference portion of the Jamboree is to implement a number of perturb-seq inference methodologies using common input and output formats. Following the Jamboree, these implementations will be added as modules to a Nextflow pipeline, which will then be used to benchmark their statistical and computational performance. This benchmarking effort will suggest best practices for perturb-seq inference, and will be used to inform the development of the IGVF perturb-seq analysis pipeline.

1.3 Data format overview

The primary input to a perturb-seq inference module is a `MuData` object, which contains both the perturb-seq data and a set of element-gene pairs for which the inference is to be performed. The output of each method should be the same `MuData` object, except with an additional table containing one or more measures of association for each element-gene pair. The `MuData` format is an HDF5-based language-agnostic format compatible with import into both R and Python. Each `MuData` object will contain a minimal set of fields required for inference, and potentially one or more optional fields that provide additional information. For the purposes of this Jamboree, we have provided `MuData` objects for subsets of the Gasperini et al (2019) and Papalexi et al (2021) datasets. For each dataset, we have provided a minimal `MuData` object that contains just the required fields, as well as a more fleshed out object that contains additional optional fields. In-depth descriptions of the `MuData` structures are provided from the perspectives of R and Python.

2 Requested code submission

2.1 Inference function

Please write a function in your language of choice with the following arguments:

- The first argument should be `mudata_input_fp`, a filepath to a `MuData` object.
- The second argument should be `mudata_output_fp`, a filepath to an output `MuData` object.
- There may be one or more additional arguments specific to your method.

The function should read the `MuData` object from `mudata_input_fp`, perform the inference, and write the resulting `MuData` object to `mudata_output_fp` (in R, via `MuData::readH5MU()`). The function should include documentation of any additional arguments used. For example, see wilcoxon-test.R.

2.2 Demonstration

Please create an iPython notebook (unfortunately R Markdown is not supported on our computer cluster) to briefly describe what your function does and demonstrate its use on at least one of the sample datasets provided. For example, see wilcoxon-test-demo.ipynb.