Methods used for PhIP-Seq Data Analysis

Caitlin et. al

September 14, 2020

Alignment Pipeline The enrichment matrix was created by aligning all samples to the reference library using a **Nextflow** data processing pipeline. The pipeline starts with the metadata for all samples (including a path to the fastq reads) as well as the metadata for all the peptides in the library. The processing steps are as follows: (1) Build a **Bowtie** CITE[] index from the peptide metadata by converting the metadata to fasta format and feeding into the **bowtie-build** command. (2) Align each of the samples to the library using **Bowtie** end-to-end alignment allowing for up to 2 mismatches. Each of the reads is 125bp long, so we trim the low quality end of the read to match the reference length, 117bp, before alignment. (3) The pipeline extracts the counts of each peptide in each sample alignment by using **samtools-idxstats** CITE[]. (4) Finally, all the individual counts information for each sample are merged into a single raw counts *enrichment* matrix. The final result of the alignment pipeline is a dataset containing the enrichment matrix, sample metadata, and peptide metadata organized into a coherent dataset using the **xarray** CITE[] package.

Sample curation and model fit groups Each sample in our dataset is defined by a single Immuno-Precipitation (IP) experiment with a serum antibody sample and library batch of phage-displayed peptides. To be sure our results are reproducible, we curate results from samples using two classes of replicates for each serum sample. *technical replicates* represent a set of samples which are from the same serum sample. The IP experiments for each technical replicate use the same library batch, and are performed by the same person on the same day. *biological replicates* are defined by IP experiments done on a different day with a split serum draw; may be the same or different libraries. The first step of sample curation is a 0.5 threshold of correlation between the counts of all technical replicates. The counts of the samples above this threshold are then averaged to make a single reproducible biological replicate. For data from a serum to be included in the final analysis, it must have a reproducible biological replicate using both library batches. Then for each sample, we take the two biological replicates with highest technical replicate reproducibility, such that the two were not generated using the same library After sample curation we are left with two sample sets divided by the library batch used for respective sample IP experiments. Concretely, each serum sample's results in the analysis is represented by a biological replicate IP using each of the two library batches.

Assessing significance using a Gamma-Poisson model We observe noise introduced by library peptide abundance and "sticky" peptide bias. To parse this, each of the two curated sample sets are fit to a Gamma-Poisson mixture model first introduced by **phip-stat** CITE[]. We fit each model with a number of mock-IP controls to give us information about abundance and false binding affinity for each peptide in the library. To control for the variance in sequencing coverage between samples, we first normalize all samples using counts factor method as seen in CITE[]. This leaves us with a normalized raw counts matrix, $M_{i,j}$, for i peptides and j samples. The model assumes that each entry in the count matrix, for any given peptide i, is sampled from a Poisson distribution with rate, λ_i. Next, we assume the prior distribution for each possible λ_i is pulled from a Gamma distribution defined by α and β (inverse scale) parameters. We then infer these parameters by looking at the set of mean normalize counts for each peptide using the **scipy.optimize** CITE[] package. Given the posterior of the rate is also Gamma distributed, the posterior hyperparameters are thus defined $\alpha^{'} = \alpha + \sum_i^n x_i$, $\beta^{'} = \beta + n$. Given these, $\lambda_i = \alpha_i^{'}/\beta_i^{'}$. Finally, we get $-log10(pval)$ (MLXP) values by computing the size of the tail of the Poisson distribution for each normalized sample count at peptide, i.