

Analysis of Gene Essentiality from TnSeq Data Using Transit

Thomas R. Ioerger
Department of Computer Science
Texas A&M University
ioerger@cs.tamu.edu

December 22, 2020

Abstract

TnSeq, or sequencing of transposon-insertion libraries, has proven to be a valuable method for probing the functions of genes in a wide range of bacteria. TnSeq has found many applications for studying genes involved in core functions (such as cell division or metabolism), stress response, virulence, etc., as well as to identify potential drug targets. Two of the most commonly used transposons in practice are Himar1, which inserts randomly at TA dinucleotides, and Tn5, which can insert more broadly throughout the genome. These insertions cause putative gene-function disruption, and clones with insertions in genes that cannot tolerate disruption (in a given condition) are eliminated from the population. Deep-sequencing can be used to efficiently profile the surviving members, with insertions in genes that can be inferred to be non-essential. Data from TnSeq experiments (i.e. transposon insertion counts at specific genomic locations) is inherently noisy, making rigorous statistical analysis (e.g. quantifying significance) challenging. In this paper, we describe Transit, a Python-based software package for analyzing TnSeq data that combines a variety of data processing tools, quality assessment methods, and analytical algorithms for identifying essential (or conditionally essential) genes.

Keywords: essentiality, transposon mutagenesis, TnSeq, statistical analysis, software

1 Introduction

TnSeq, or sequencing of transposon-insertion libraries, has proven to be a valuable method for probing the functions of genes in bacteria [1–3]. Libraries are generated by random insertion of a transposon throughout a genome, creating a pool of mutants. DNA from bacteria grown in a particular condition is extracted, the junctions between transposon and adjacent genomic region are amplified by PCR, the samples are sequenced by high-throughput sequencing, and insertions at individual sites are counted. The pattern of insertion counts can be used to infer the difference between essential (ES) and non-essential (NE) regions; regions that tolerate insertions are generally considered non-essential, while regions lacking insertions are taken as evidence of essentiality. In addition to the binary distinction of essential vs non-essential, some genes can exhibit a partial reduction of insertion counts, which can be interpreted as a growth defect (GD) caused by disruption of the gene, and indeed quantitative changes in insertion counts in has been shown to reflect changes in fitness *E. coli* [1]. There are several variations of sample preparation protocols currently in use, going by names such as TraDIS [4], HITS [5], and InSeq [6], which differ based on the transposon used, method of DNA fragmentation, and transposon junction amplification. More recently, Barseq [7] has been introduced to exploit barcoding to afford a very high level of multiplexing on next-gen sequencers, allowing the efficient analysis of hundreds of samples in parallel.

37 TnSeq has found many applications in many different bacteria, including *E. coli*, *Mycobacteria*,
38 *Pseudomonas*, *Vibrio*, *Haemophilus*, *Salmonella*, etc., to study genes involved in metabolism, stress
39 response, virulence, etc. For example, TnSeq has been used to study genes required for metabo-
40 lizing/utilizing specific nutrients such as cholesterol [24] or iron [33], tolerance of stress conditions
41 [37, 38], genetic interactions with knocked-out genes in libraries made from null mutants [39–41],
42 and virulence in animal models [4, 5, 9, 42, 43]. TnSeq gives a very different read-out on gene func-
43 tion than RNA-seq (transcriptomics), as essentiality of a gene is orthogonal to (i.e. not necessarily
44 correlated with) the level of expression [8]; one can change without the other. Gene essentiality is
45 especially useful for drug discovery efforts, such as interpreting mechanisms of resistance through
46 mutations found in resistant mutants and identifying vulnerable targets and pathways that are
47 critical for survival in vivo (during infection) [9]. Databases of essential genes, such as DEG [10],
48 have been assembled from curated collections of published results of TnSeq analyses, along with
49 inferences of essential genes extended to other species by homology.

50 Two of the most commonly used transposons in practice are Himar1 [11, 12] and Tn5 [4, 13].
51 While Himar1, a member of the mariner family, is restricted to inserting at TA dinucleotides [14],
52 Tn5 can insert effectively at any site in the genome, which has consequences for statistical analysis.
53 The magnitude of counts at an individual site is highly dependent on the library, as insertion into
54 the chromosome is a stochastic process, which is a major factor contributing to variability. Although
55 the magnitude of insertion counts can vary significantly between adjacent TA sites in non-essential
56 regions, no strong sequence-dependent bias has yet been found for insertion of various transposons.
57 For Himar1, there is an apparent non-permissiveness of TA sites with a G at flanking positions
58 +/-2 bp, with G or C at +/-3 [44], and also a general preference for insertions in more bendable
59 regions of the DNA [45]). Similarly, only a generalized sequence preference has been identified for
60 Tn5, but it is insufficient to make insertion locations predictable [46]. Thus insertion counts are
61 usually treated as a random variable, and analysis methods rely on averaging over multiple TA
62 sites in a gene to make a statistical assessment of essentiality. Even a GC-rich organism like *M.*
63 *tuberculosis* has 3 or more TA sites for most genes (median is 13 TA sites; only 2.7% of genes have
64 less than 3 TA sites); analysis of genes with only 1-2 TA sites is typically not reliable (analysis
65 of short genes becomes highly dependent on the degree of saturation), and of course genes with
66 0 TA sites are unanalyzable by TnSeq. However, intergenic regions and ncRNAs are often short
67 enough to have only 0-2 TA sites, and are difficult to analyze using TnSeq. Many of the analytical
68 methods for Himar1 can be extended to Tn5 data, provided that the saturation is high enough and
69 the assuming the locations and magnitudes of insertions can effectively be treated as random [2].
70 It is important to note that essential genes, which typically lack insertions throughout the body
71 of the ORF, have often been observed to tolerate insertions at the N- and C-termini, as well as in
72 linkers between domains, and can even contain non-essential domains, which also pose challenges
73 for rigorous identification of essentials. Some studies have observed large-scale chromosomal
74 biases, where mean insertion counts exhibit a trend based on position on the chromosome, such as
75 a gradation based on distance between the origin and termination of replication. Adjustments for
76 such biases can be made through methods such as the LOESS correction [47], which equalizes the
77 smoothed mean across the whole genome.

78 Data from TnSeq experiments (i.e. insertion counts) is inherently noisy, making rigorous statis-
79 tical analysis challenging. Variability can come from a variety of sources, including representation
80 (abundance) in the library, stochastic differences between identically-treated samples (plates, cul-
81 tures, animals), and the sequencing process. Reducing raw counts to template counts using bar-
82 codes can help ameliorate PCR jackpotting [15]. Loss of diversity (especially in animal infections)
83 and amplification of fitness differences due to differences in growth time (generations) can also lead
84 to significant artifacts such as skewing of read-count distributions [16]. Larger genes with more

85 TA sites ease this because the impact of isolated deviations (e.g. missing or outlier counts) can be
86 mitigated by averaging over multiple observations. Conversely, higher noise and lower saturation
87 can make statistical calls for smaller genes uncertain. Hence, normalization and rigorous statisti-
88 cal analysis are critical to identifying significant genes (e.g. essential or conditionally essential)
89 Also, collection of multiple replicates (2-3 replicates per sample is recommended) is important for
90 increasing statistical certainty through increasing the number of observations per gene. (Biological
91 replicates, i.e. samples from different plates or animals, are more important than technical repli-
92 cates, i.e. resequencing, for giving a fair picture of stochastic variability of insertion counts, which
93 is critical assessing statistical confidence in observed differences.) Several software packages are
94 available for analysis of TnSeq data, each based on different theoretical framework: ESSENTIALS
95 [17], TnSeq-Explorer [18], TraDIS-Toolkit [19], TnSeqDiff [20] fitness ratios [21], ARTIST [22], and
96 others.

97 In this paper, we describe Transit [23], a Python-based software package for analyzing TnSeq
98 data. Transit combines implementations of a range of previously published statistical analysis
99 methods. Transit was originally designed for statistical analysis of Himar1 TnSeq datasets (in
100 which insertions are assumed to be restricted to TA sites), though some of the methods have
101 been adapted for Tn5 data (for which the typical lower saturation causes challenges). Transit
102 has a pre-processor (TPP) for extracting insertion counts from raw sequence (.fastq) files that
103 encodes best practices accumulated from experience in multiple labs over the years. Transit also
104 incorporates tools for quality control (QC analysis) for assessing the quality of datasets. Transit
105 has a graphical user interface (GUI) to make it easy for users to to perform most tasks, though
106 this review will focus on running tasks from the command line. The methods are also accessible as
107 a Python library for programmers to call in their own scripts (the source code can be downloaded
108 from Github, <https://github.com/mad-lab/transit>). Further details can be found in the online
109 documentation, <https://transit.readthedocs.io/en/latest/>.

110 The analytical tools in Transit are oriented around addressing three types of questions:

- 111 1. identifying essential genes in a single (e.g. reference) condition
- 112 2. comparative evaluation of conditional essentiality between two conditions
- 113 3. analysis of genes showing variability across multiple conditions

114 Each of these has different use cases. A common case of single-condition analysis might be assessing
115 essential genes and pathways in a new bacterial species. A common case of pairwise analysis might
116 be comparing essentiality in a stress condition, such as starvation, iron limitation, low pH, hypoxia,
117 antibiotic exposure, or growth in an animal model, compared to a reference condition, such as
118 growth on rich medium. However, recently, more complex TnSeq experiments are being conducted
119 involving multiple experimental variables/treatments, such as varying antibiotic concentrations,
120 varying durations (number of days or weeks in culture or in vivo), comparison of survival in different
121 animal breeds/species/genotypes, or supplementation with various nutrients, making tools for task
122 multi-condition analysis necessary to explore/characterize patterns of response over larger sets of
123 experimental conditions. This paper describes how to do these analyses using Transit, and includes
124 comments about the impact of (and guidelines on) data quality, file formats, and other practical
125 information.

2 Materials

Transit is designed to run on Linux, Macs, and Windows machines. Transit is written in python3, and requires wxPython4, R, and many other packages as dependencies. Transit can be installed in two ways: 1) via ‘pip3 install’, which will download and install the compiled code from the repository on PyPi along with all dependencies (so that it can be run directly from the command line as ‘transit’), or 2) by cloning the source code from GitHub, and then manually installing the required packages (see Installation Instructions in the online documentation, <https://transit.readthedocs.io/en/latest/>). The current version is 3.1.0 as of this writing; for future changes, see the online documentation. Note that BWA must be installed for TPP, and some of the more advanced functions (like ZINB) require R to be installed, along with some specific packages.

In the sections below, we will give examples based on a study of gene requirements for growth of *M. tuberculosis* H37Rv on cholesterol compared to glycerol [24]. In this study, there are 5 TnSeq datasets collection, 2 for glycerol and 3 for cholesterol. For brevity, we will refer to these wig files as G1.wig, G2.wig, C1.wig, C2.wig, and C3.wig.

If installed from GitHub, running Transit would require command sequences like ‘python3 \$TRANSITDIR/src/transit.py ...’, where \$TRANSITDIR is path where Transit is installed. However, for simplicity, we will simply use the command ‘transit’ in the examples below.

Users can get help on most commands (e.g. reminders of arguments and flags) by running Transit without any arguments, or with ‘--help’.

Most of the analyses in Transit generate tab-separated output files as a convention, which can be opened as spreadsheets in Excel. Lines prefixed with ‘#’ in the output files are comments.

All the input and output files for the examples in this chapter can be accessed online at: <http://orca1.tamu.edu/essentiality/transit/examples/index.html>

3 Methods

3.1 Pre-Processing (TPP)

Transit has a pre-processing step called TPP (Transit Pre-Processor) that maps reads from sequencing a Tn library (files in .fastq or .fasta format) into a genome (.fasta format) and tabulates insertion counts at TA sites. The counts are output in a file format called ‘.wig’ files, which simply have two values on each line - coordinate and insertion counts for each TA site. Wig files have two header lines, the second of which indicates the name of the genome sequence that was used as a reference (‘variableStep chrom=H37Rv’ indicating H37Rv.fna as the reference sequence, for example). In subsequent analyses, it is critical to use the annotation file corresponding to the genome sequence used in TPP, to ensure consistency of the coordinate system.

TPP uses BWA [25] (which must be installed separately) to map reads into the genome. Since the reads represent junctions between the transposon and chromosome, the first step is identifying reads (in read 1) with a prefix matching the terminus of the transposon, and the stripping this prefix off to map the genomic suffix. Many protocols introduce random shifts in the location of the prefix sequence within the reads, to reduce sequencing problems when all reads begin with the same nucleotides. By default, TPP searches for a prefix corresponding to Himar1, but other transposons can be accommodated by specifying the search sequence using the ‘-prefix’ flag. Illumina sequencers often provide pairs of reads, and while read 2 is not absolutely necessary for mapping reads (i.e. optional), Sasseti et al. have shown how to embed random nucleotide barcodes in read 2 that can be used by TPP to reduce raw read counts at each TA site to counts of unique DNA templates,

170 which helps reduce noise due to PCR jackpotting effects [15]. Because of the lengths of the prefix
171 in read 1 and barcodes (and surrounding constant regions and genomic regions in read 2), it is
172 recommended to use a read length of at least 75 bp (i.e. 75x75 bp paired-end sequencing). Various
173 flags can be used to adjust the methods and parameters for mapping reads, such as number of
174 mismatches allowed or constraints on location of prefix.

175 An example of running TPP is as follow:

```
176 > tpp -bwa /home/bwa/bwa-0.7.12/bwa -ref H37Rv.fna -reads1 G1_R1.fastq  
177 -reads2 G1_R2.fastq -output G1
```

178 The inputs are the sequencing data (fastq files for read 1 and 2) and the reference genome
179 sequence (H37Rv.fna, nucleotide fasta). Multiple intermediate files will be generated for the base
180 filename given by -output. The primary output file would be the wig file, **G1.wig**, which contains
181 the insertion counts at TA sites.

182 TPP can also map reads to genome sequences containing multiple contigs (or replicons), such as
183 when an organism contains multiple chromosomes and/or plasmids. Separate reference sequences
184 can be provides to TPP as a comma-separated list.

185 Another important output file from TPP is the .tn_stats file that gets generated. It contains
186 important parameters and diagnostics from the run. As a quick check, it is useful to examine the
187 saturation (fraction of TA sites with insertions, ideally > 30%) and NZmean (mean over non-zero
188 sites, ideally > 10). The .tn_stats files also reports the total reads, number of reads mapped, and
189 related statistics which can give insight into the degree of attrition and possible reasons for it
190 (for example, whether reads lacked the expected prefix, mapped predominantly to a single site, or
191 matched known sequences related to the transposon vector or primers).

192 3.1.1 Making Combined Wig Files

193 When working with a large collection of datasets, it can be cumbersome to have to type multiple
194 wig filenames as inputs to subsequent commands. In such cases, it is convenient to make a ‘com-
195 bined_wig’ file, which combines the counts at the same TA sites from multiple wig files on the same
196 line (and appends on the ORF id and gene name from the annotation). To aid in comparing inser-
197 tion counts between datasets, the counts are normalized using TTR normalization (by default, see
198 below), though other normalization methods (including ‘nonorm’ which preserves raw counts) can
199 be applied using the -n flag. The “#File:” header lines cocument the datasets that were combined.

```
200 > transit export combined_wig G1.wig,G2.wig,C1.wig,C2.wig,C3.wig H37Rv.prot_table  
201 glyc_chol_combined_wig.txt # TTR norm is implicit  
202  
203 > transit export combined_wig G1.wig,G2.wig,C1.wig,C2.wig,C3.wig H37Rv.prot_table  
204 glyc_chol_combined_wig_raw.txt -n nonorm
```

205 A recommended practice is to use the combined_wig file for examining and comparing normalized
206 insertion counts in a gene or locus that is indicated to be conditionally essential, to confirm whether
207 the effect is genuine or perhaps influenced by a few outlier insertion counts.

208 The ‘.prot_table’ file contains the coordinates and annotations of genes in the genome (see
209 Section 3.2.2 below).

210 It is often useful to reduce the data in each sample to the mean insertion counts for each gene,
211 which can be done with the ‘export mean_counts’ command:

```
212 > transit export mean_counts G1.wig,G2.wig,C1.wig,C2.wig,C3.wig H37Rv.prot_table
213                               glyc_chol_gene_means.txt
```

214 The output file, glyc_chol_gene_means.txt can be opened as a spreadsheet.
215 An individual wig file may be normalized itself using the following command,

```
216 > transit normalize <input.wig> <output.wig> -n [method]
```

217 where ‘method’ is TTR, betageom, or several others (see documentation). Normalization can also
218 be applied simultaneous to multiple datasets in a combined_wig file by adding the ‘-c’ flag before
219 the filename.

220 3.1.2 Evaluating Quality of TnSeq Data

221 To summarize the statistics of multiple samples in a combined_wig file, one may use the ‘tnseq_stats’
222 command (given either a list of individual wig files, or a combined_wig file):

```
223 > transit tnseq_stats G1.wig G2.wig C1.wig C2.wig C3.wig -o glyc_chol_tnseq_stats.txt
224 or
225 > transit tnseq_stats -c glyc_chol_combined_wig_raw.txt -o glyc_chol_tnseq_stats.txt
```

226 The output file is a tab-separated file that can be opened as a spreadsheet. It contains impor-
227 tant information, including the saturation, total counts, mean, NZmean, and max count, for each
228 dataset, along with several statistics on the read-count distribution (skewness, etc.). While the
229 saturation will not change, it is best to run *tnseq_stats* on a combined_wig file to which no normal-
230 ization has been applied (‘-n nonorm’), since otherwise, the mean counts will be reflect the results
231 of TTR normalization. If datasets with especially low saturation (e.g. < 15%) or low NZmean (e.g.
232 < 1) are observed, the researcher might consider excluding them from further analyses.

233 Users can assess the quality of the TnSeq datasets by generating and examining the **tnseq_stats**
234 table mentioned above. The primary metrics are saturation and mean insertion count (specifically,
235 NZmean). While there are not rigorous criteria for defining “bad” datasets, rules of thumb I use for
236 “good” datasets are: density > 30% (ideally > 50%) and NZmean > 10 (ideally > 50). In addition,
237 I look at MaxReadCount and Skewness as indicators. Typically, MaxReadCount will be in the
238 range of a few thousand to tens-of-thousands. If you see individual sites with counts in the range
239 of $10^5 - 10^6$, it might mean you have some positive selection at a site (e.g. biological (fitness
240 advantage), or an artifact due to things like PCR jackpotting), and this can have the effect of
241 reducing counts and influencing the distribution at all the other sites. If MaxReadCount < 100, that
242 is also probably problematic (either not enough reads, or possibly skewing). Also, skewness > 30
243 often (but not always) signals a problem. The reason it is not easy to boil all these down to a
244 simple set of criteria is that some some of the metrics interact with each other.

245 A useful tool when evaluating the quality of a collection of TnSeq datasets is to make a cor-
246 relation plot of the mean insertion counts (averaged at the gene-level). While, it is difficult to
247 state how much correlation there should be between conditions (or even between replicates of the
248 same condition), the corrplot can often reveal individual samples which stand out as being far less
249 correlated with all the others (which subsequently might be excluded from analyses).

```
250 > transit corrplot glyc_chol_combined.wig.txt glyc_chol_corrplot.png
```

251 Although saturation and NZmean are the easiest to assess, a more complicated aspect of data
252 quality is the insertion-count distribution. Generally speaking, the number of TA sites with
253 low counts are most abundant, and sites with high counts are far less frequent. While it is not
254 theoretically guaranteed, well-behaved datasets typically conform closer to a geometric distribution
255 (though possibly with some excess dispersion, which could be modeled as a negative binomial). In
256 contrast, lower-quality datasets often exhibit a significant skew away from this distribution. Skewed
257 datasets are often more dominated by excessively high counts at a few sites, while the counts
258 at the vast majority of remaining sites are quite low by comparison. This can cause problems
259 with normalization; even though TTR is designed to be robust to a few outliers, it is only linear
260 transformation and cannot correct for skew in the data. This can lead to an inflation of apparently
261 differentially essential genes (artifacts) when skewed datasets are compared to other conditions.

262 The Transit GUI provides plots of the insertion-count distribution, including a QQ-plot (quantile-
263 quantile plot) against an ideal geometric, to visually assess the degree of skew. The closer to a
264 diagonal, the better. Aside from ‘skewness’ as a metric itself, the `tseq_stats` table also report the
265 Pickand’s tail index [48]. Anecdotal experience suggests that dataset with $PTI > 1.0$ (and maybe
266 even $PTI > 0.5$) are potentially problematic.

267 The causes of skew are (currently) not well understood, though, anecdotally, it often seems to be
268 associated with loss of diversity (e.g. in animal-passaged samples) over ‘over-selection’ of libraries.
269 For example, culturing too long under a stress condition can amplify the effects of even small fitness
270 differences.

271 What can be done with skewed datasets? One solution is to apply the Beta-Geometric cor-
272 rection (BGC) [16]. This is a non-linear normalization procedure that adjusts the insertion-count
273 distribution to look more like a conventional geometric distribution. This can be accomplished us-
274 ing the ‘-n betageom’ flag to the ‘normalize’ command. BGC normalization dramatically squashes
275 down the sites with the highest counts, but if it is applied uniformly to all the samples in a com-
276 parison, it often greatly reduces the number of (apparently) significant hits, hopefully getting rid of
277 false positives and retaining only a few true positives where the trend of insertions truly supports
278 a change in essentiality (fitness difference).

```
279 > transit normalize G1.wig G1_BGC.wig -n betageom
```

280 3.2 Analyses for Single Conditions

281 3.2.1 Gumbel Analysis

282 Analyses of individual conditions might occur when a new strain is being evaluated in a reference
283 condition (rich growth medium, for example), and the goal is to make a preliminary catalog of
284 essential genes. There are two principle methods in Transit: Gumbal analysis, and a Hidden
285 Markov Model (HMM). Gumbel analysis focuses on genes with statistically significant gaps, or
286 consecutive sequences of TA sites lacking insertions (empty sites, with counts of 0) [26] Since most
287 libraries are sub-saturated, empty sites will occur in non-essential regions at random, but long
288 sequences of empty sites are statistically unlikely, and genes containing such gaps are taken as
289 essential. The Gumbel method uses the Extreme Value Distribution to quantify the certainty, and
290 a posterior probability is reported in the output file (with separate thresholds for Essential and
291 Non-essential genes, and Uncertain genes in between). An important advantage of this gap-based
292 approach is that it is tolerant of insertions which sometimes occur at the N- and C-termini of ORFs.
293 A disadvantage is that it works less well with less-saturated libraries ($< 40\%$), and can yield many
294 more Uncertain calls, especially for shorter genes (with < 10 TA sites). Note that the magnitude
295 of insertion counts (and hence normalization) do not matter for Gumbel analysis.

```
296 > transit gumbel G1.wig,G2.wig ref.prot_table gumbel_H37Rv_glycerol.txt
```

297 Gumbel can be run on multiple replicates. If multiple replicates are available, they can be
298 provided to Gumbel via a comma-separated list, as shown above, and the data will be merged.
299 The output file will contains values for each gene, such as number of TA sites, number of sites with
300 insertion, longest run of sites without insertion, and finally a posterior probability ('zbar') and call
301 (ES=essential, NE=non-ess, U=uncertain, S=too short to analyze). This command also has flags
302 for explicitly trimming (ignoring) TA sites in termini of genes, etc.

303 A typical expectation is that around 10-15% of genes will be essential for growth in vitro in most
304 bacterial genomes, as has been observed across many organisms [10], though of course it depends
305 on size of genome and growth condition.

306 3.2.2 Prot Tables

307 An important input file to Gumbel, like many other methods in Transit, is a "prot_table," which con-
308 tains the annotation of the genome (including start and stop coordinates of genes). The prot_table
309 format is derived from an old format that could be downloaded from GenBank years ago. It is
310 a tab-separated file with the following fields for each gene: function/description, start coord, end
311 coord, strain ('+' or '-'), gene length (in amino acids; unused), 3 unused fields, gene name, and
312 ORF id. More recently, the gff (or gff3) format is being used for genome annotations. Transit has a
313 method to convert a gff file into a prot_table. However, there is greater flexibility in the gff format
314 (especially, varying use of keywords), which is difficult to anticipate, so if the Transit command
315 does not work, users might have to write their own script to convert their gff file to a prot_table.

```
316 > transit convert gff_to_prot_table <ref.gff> <ref.prot_table>
```

317 See Notes on including other types of genes in prot_tables. Note that one can add other types
318 of genes to a prot_table, such as tRNAs, rRNAs, and ncRNAs. All that is required it to include
319 lines with the expected format, including start and end coordinates and strand. Furthermore, the
320 same approach can be used to represent other types of loci, such as operons (spanning multiple
321 genes) or intergenic regions, to be analyzed for essentiality. However, note that most intergenic
322 regions and ncRNAs are relatively small compared to typical protein-coding regions, and hence
323 they often have only a few (if any) TA sites, making essentiality analysis highly uncertain (except
324 for the most highly saturated libraries [44]).

325 3.2.3 Hidden Markov Model

326 The HMM can also be used to assess TnSeq data in a single condition. It works a different principle
327 - using a probabilistic model to estimate the state at each TA based on the counts and consistency
328 with adjacent sites [27]. This allows the HMM to smooth over individual outlier values (such as an
329 isolated insertion in any otherwise empty region, or empty sites scattered among insertion in a non-
330 essential region) and make a call for a region/gene that integrates information over multiple sites.
331 The important difference from Gumbel analysis is that the HMM takes into account the magnitudes
332 of insertion counts, which can also carry information about the growth requirement (or fitness
333 effect) of a gene. This allows the HMM to make finer distinctions, utilizing 4 states for individual
334 sites: ES (essential), GD (growth-defect), NE (non-essential), and GA (growth-advantaged). One
335 might see a GD call for a gene whose disruption (by the transposon) impairs growth, so counts
336 are suppressed (compared to the global average, but not all the way to 0), while a gene might be
337 called GA if transposon insertions actually confer a growth advantage, resulting in inflated counts

338 [21]. The gene-level calls are made based on the majority call among the TA sites within each
339 gene. The HMM automatically tunes its internal parameters (e.g. transition probabilities) to the
340 characteristics of the input datasets (saturation and mean insertion counts), and can work over a
341 broad range of saturation levels (as low as 20%, [27]).

```
342 > transit hmm G1.wig,G2.wig H37Rv.prot_table hmm_H37Rv_glycerol.txt
```

343 The HMM command has flags for normalization (-n) and also how to handle merging of repli-
344 cates (-r), though defaults are usually fine. There are actually two output files generated by the
345 HMM, one named on the command line, which will contain the analysis (e.g. state probabilities
346 and call) at each individual TA site, and another, `hmm_H37Rv_glycerol_genes.txt` ('genes' added
347 as suffix to filename), which contains the call for each gene. There is no "probability" associated
348 with the call for each gene. It is important to keep in mind that the call for a given gene can be
349 influenced by insertions in the adjacent region, which is part of the design of HMMs.

350 The '-l' flag may be used to apply a LOESS correction to adjust for large-scale variations
351 in insertion counts across the genome. Because the HMM takes the magnitudes of insertions
352 into account, it can be affected by large-scale variations in insertion counts across the genome. In
353 the GUI, a plot of the smoothed mean insertion count can be generated. If there are noticeable
354 differences, the local deviations from the global mean can be subtracted out using the LOESS
355 correction [17] ('-l' flag for the `hmm` command). While this is typically not necessary for Tn
356 libraries in *M. tuberculosis*, chromosomal biases can be more severe in faster-replicating species,
357 like *Vibrio cholera* [3] and *E. coli* [49].

358 Because essential and growth-defect regions are close (both have suppressed counts), the dif-
359 ference in call by the HMM (ES vs GD) can be affected by noise, so we often combine the two
360 categories. Typically, 15-20% of genes in a bacterial genome are called by the HMM as ES or GD,
361 suggesting they are either absolutely required (ES) or contribute to fitness (GD) in the growth
362 condition evaluated.

363 3.3 Pairwise Comparisons: Resampling

364 Comparisons between two conditions can be used to identify conditionally-essential genes. In
365 addition to binary cases where an essential becomes non-essential or vice versa, we also include
366 genes with *quantitative changes in insertion counts*, reflecting apparent fitness changes for mutants
367 (for example, genes in which the mean insertion count decreases by 2-fold, though not going all
368 the way to 0). The primary tool in Transit used for pairwise comparisons is "resampling." It is
369 equivalent to a permutation test on the difference of the mean counts between the two conditions
370 for each gene. Of course, because of stochasticity, almost every gene will exhibit some difference in
371 mean insertion counts between any pair of conditions. What matters is whether the difference is
372 statistically significant.

373 Resampling is a frequentist approach that compares the observed differences of mean insertion
374 counts to a null distribution to determine whether the difference is larger than would be expected
375 by chance (given the same counts but without knowledge of the condition). First, the normalized
376 insertion counts are pooled over all TA sites and replicates for both conditions. Then a null
377 distribution is created by repeatedly drawing random two samples (of the same size as the original
378 number of observations) from the pooled counts and calculating the difference of means 10,000
379 times. From this a two-tailed p-value is derived for the observed difference, and the p-values
380 are adjusted post-hoc for multiple testing via the Benjamini-Hochberg correction [50] to limit the
381 overall false discovery rate (FDR) to 5%.

382 This comparative approach to identifying conditionally essential genes is preferred over simply
 383 combining the results of individual analyses of each condition (e.g. Gumbel) and selecting genes
 384 that are called Essential in one condition but not the other, because the results of that approach
 385 could be influenced by which genes fall just above or below the significance cutoff. Resampling
 386 is a direct comparison that looks at differences in insertion counts. The resampling procedure is
 387 designed to be (somewhat) robust to noise (e.g. outliers, high variance among counts), and is
 388 appropriately less sensitive for smaller genes and more sensitive with more replicates.

389 Normalization is critically important for resampling. If the individual datasets are not properly
 390 normalized, it can lead to the appearance of an artificially inflated number of conditional essentials
 391 (i.e. false positives, artifacts). The default normalization used in Transit is called TTR (Trimmed
 392 Total Read-count). For each dataset, the total insertions over all TA sites is tabulated, after
 393 removing the top and bottom 1% of sites (to reduce the influence of outliers). The sites are
 394 divided by this total, and then scaled-up so that the mean insertion count over the whole genome
 395 is 100. While the calculation is simple, it puts counts from different datasets on a comparable
 396 basis. Furthermore, it maintains a balance so that counts from less-saturated datasets are inflated
 397 proportionally, so that the mean insertion for most genes (averaged over multiple TA sites) stays
 398 about the same on average. This objective is important for the assumptions of resampling (see
 399 explanation in [23]). A way to verify that TTR normalization is doing the right thing is to make a
 400 scatter plot of the mean insertion counts for each gene between any two datasets; the data should
 401 scatter along the X=Y diagonal. Individual points can deviate from this line due to: experimental
 402 noise, sampling error for small genes with few TA sites, or biological differences (if the datasets
 403 represent distinct experimental conditions). But if there is a systematic deviation away from the
 404 X=Y line in this scatter plot (due to improper normalization), it would likely produce an excess of
 405 conditional essentials detected by resampling.

406 Resampling can be run on 2 sets of wig files (comma-separated) as follows:

```
407 > transit resampling G1.wig,G2.wig C1.wig,C2.wig,C3.wig H37Rv.prot_table
408                 resampling_glyc_chol.txt -a
```

409 The full list of options for the resampling command is:

```
410 > transit resampling
411 usage:
412 transit resampling <comma-separated .wig control files> <comma-separated .wig experimental files>
413                 <annotation .prot_table or GFF3> <output file> [Optional Arguments]
414 or
415 transit resampling -c <combined_wig> <samples_metadata> <ctrl condition name> <exp condition name>
416                 <annotation .prot_table> <output file> [Optional Arguments]
417 NB: The ctrl and exp condition names should match Condition names in samples_metadata file.
418
419 Optional Arguments:
420 -s <int>      := Number of samples. Default: -s 10000
421 -n <string>   := Normalization method. (Default: -n TTR)
422 -h           := Output histogram of the permutations for each gene. (Default: Turned Off)
423 -a           := Perform adaptive resampling. (Default: Turned Off)
424 -ez         := Exclude rows with zero across conditions. (Default: Turned Off)
425 -PC <float>  := Pseudocounts used in calculating LFC. (default: 1)
426 -l           := Perform LOESS Correction (Default: Turned Off)
427 -iN <int>    := Ignore TAs occurring within given percent (as int) of the N terminus. (Def: 0)
428 -iC <int>    := Ignore TAs occurring within given percent (as int) of the C terminus. (Def: 0)
429 --ctrl_lib   := String of letters representing library of control files in order
```

430 e.g. 'AABB'. Default empty. Letters used must also be used in --exp_lib
431 If non-empty, resampling will limit permutations to within-libraries.
432 --exp_lib := String of letters representing library of experimental files in order

433 where typically the experimental condition represents a treatment and the control condition rep-
434 resentation the untreated or reference condition. The samples_metadata used with combined_wig
435 inputs will be explained in the next section.

436 The input files are implicitly normalized by TTR (though this can be changed using '-n').

437 TA sites near the termini of ORFs can be trimmed using -iC and -iN. For example, to ignore
438 insertion in the first or last 5% of an ORF, use '-iC 5 -iN 5.'

439 I recommend using the '-a' flag to apply the adaptive version of resampling, which is much faster
440 and generally outputs p-values close to those obtained by performing all 10,000 samples (within a
441 factor of 2-3). It truncates the resampling early for genes when it is clear that they are not going
442 to be significant. Almost all significant genes will be still be significant, except for possibly a few
443 marginal differences very close to the 0.05 threshold on adjusted p-value.

444 If the -h flag is given, histograms of the null distribution from resampling for each gene, along
445 with a demarcation for the observed difference in mean insertion count between the conditions,
446 will be generated in a sub-directory. These images can be examined to determine whether the null
447 distribution for a gene of interest looks appropriately bell-shaped (versus bimodal, for example).

448 The output file contains the various statistics on the resampling for each gene and a *Padj*
449 column at the end. Users can open the tab-separated file as a spreadsheet and sort by *Padj*. The
450 conditionally-essential genes are those with *Padj* < 0.05.

451 It is difficult to say what the expectations would be for number of conditional-essentials, since
452 it depends on the impact of the biological effects of the selection conditions. In some cases,
453 the treatment (e.g. a high-stress condition) might affect the essentiality (and fitness) of hundreds
454 of genes. In other cases, biologically weak selection criteria might affect only a few (or perhaps
455 no) genes. If the experimenter feels there are too few hits (e.g. compared to their expectations,
456 possibly lacking genes known to be in affected pathways), then it might help to collect additional
457 replicates, which can increase the sensitivity of detection. If, subjectively, too many significant
458 genes are output, it might be a sign of problems with skewing of the data. The user can follow
459 the QC guidelines above to possibly identify lower-quality datasets to try excluding, or they might
460 consider applying BGC normalization ('-n betageom') to the datasets.

461 The output also reports the log-fold-change (LFC, base 2) of the mean counts. It can be used
462 to sort the genes to identify those showing the greatest increase or decrease in counts (relatively
463 less or more essential). However, only genes with *Padj* < 0.05 should be considered significant.
464 But these could be uncertain, and should be ignored if they have a *Padj* above 0.05. Frequently,
465 genes with the largest-magnitude LFC are often smaller genes (with fewer TA sites) or genes with
466 very low insertion counts, which are often not statistically significant. Of course, as with other
467 frequentist statistical tests, genes with *Padj* above 0.05 should not be over-interpreted as evidence
468 that they are *unaffected* by the condition.

469 Pseudo-counts are used in the calculation of LFCs to help reduce noise. To dampen the ap-
470 pearance of high-magnitude LFCs from genes with low insertion counts (which are more susceptible
471 to noise), one can increase the pseudo-counts using '-PC'. The LFC is calculated using the following
472 formula, which uses PC=1 by default, to avoid the result being undefined for genes with means of
473 0 in either condition:

$$LFC = \log_2((\text{mean in expt condition} + PC)/(\text{mean in ctrl condition} + PC))$$

474 For example, a gene with a mean count of 1 in the first condition and 4 in the second would
475 appear to have a fold-change of 4 and LFC of 2.0 (with PC=0). However, this is typically in

476 the range of noise for TnSeq experiments. Considering that TTR normalization adjusts the mean
477 insertion count of each sample to around 100, raising the pseudo-counts to a level of PC=5 would
478 reduce this to $\log_2(9/6) = 0.58$, masking out differences below this level.

479 By loading the resampling output file into the GUI (bottom panel), a *volcano plot* can be
480 generated (via 'Choose Action'), showing a scatter plot of the P-values versus the LFC for each
481 gene. Genes that become more essential (in the experimental condition over control) are on the left
482 (negative LFC), and significant genes are above the dashed red line (p-value threshold adjusted for
483 5% FDR).

484 See Notes for analyzing datasets from different libraries using resampling. If the replicates in
485 each conditions represent samples from multiple libraries, the sensitivity of resampling can some-
486 times be increased by performing separate random draws from the counts for each library and
487 computing the difference of the means between conditions summed separately for each library.
488 This can help reduce the variance in the difference in the mean insertion count between conditions
489 by effectively subtracting out variability due to differences in abundance at each TA site between
490 libraries. The transposon library each sample in the two conditions can be specified symbolically
491 using a string of characters. For example '--exp_lib AABB --ctrl_lib AB' would indicate that
492 the first 2 experimental samples came from library A and the second two samples from library B;
493 there can be different numbers of samples in the control condition, but they should use the same
494 codes (in this case, one sample each from library A and B).

495 3.3.1 Pathway Analysis

496 Typically, the significant genes do not all come from the same pathway, but often represent a
497 variety of pathways in which genes experience apparent increases or decreases in fitness. In order
498 to get a sense of which pathways might be enriched among the conditionally essential genes, the
499 'pathway_enrichment' command may be used:

```
500 > transit pathway_enrichment <resampling_file> <associations> <pathways>  
501                               <output_file> [-M <FET|GSEA|ONT>] [-PC <int>]
```

502 The <associations> file lists mappings of ORF ids to pathway ids in a two-column format. Three ex-
503 ample systems of functional categories are: Sanger roles [28] (123 roles), COG categories (Clusters of
504 Orthologous Genes, [29], 20 categories), and GO terms (Gene Ontology, <http://www.geneontology.org/>;
505 there are ~4000 GO terms with at least 1 gene in H37Rv). The corresponding pathway as-
506 sociations files for genes in Mtb H37Rv are provided in the Transit data directory (\$TRAN-
507 SIT/src/pytransit/data/) are: H37Rv_sanger_roles.dat, H37Rv_COG_roles.dat, and H37Rv_GO_terms.txt.

508 There may be multiple roles/pathway associations for each gene (listed on separate lines), or
509 none (e.g. for hypothetical genes not yet functionally annotated). Note that these files have been
510 expanded to include associations of each gene with all parents of each GO term or role (since
511 these are hierarchical systems). The <pathways> file gives the functional descriptions for each
512 pathway_id/role/GO term (e.g. sanger_roles.dat, COG_roles.dat, and GO_term_names.dat in the
513 data directory). These will be the same for all organisms.

514 There are three alternative methods used to evaluate significance of pathway enrichment:

- 515 • The first is to use Fisher's exact test, where the p-value based on hypergeometric distribution
516 of observed counts of category members among the hits compared the expected number based
517 on the whole genome [30]. This is the default method ('-M FET'). The output file contains a
518 list of significant pathways, sorted by adjusted p-value. In addition, the *enrichment* for each
519 gene is reported. Enrichment is defined as the ratio of *observed* pathway members among

520 the significant genes to the *expected* number based on the background proportion of category
 521 members in the overall genome. To mitigate the impact small pathways with only a few
 522 genes (which might spuriously appear as enriched), pseudocounts of 2 are incorporated in
 523 the numerator and denominator. This can be modified with the ‘-PC’ flag. However, this
 524 analysis can be of limited use if the total number of significant (conditionally essential) genes
 525 is small (< 10).

- 526 • An alternative approach is Gene Set Enrichment Analysis (GSEA, [31]) (using the ‘-M GSEA’
 527 flag). GSEA takes into account the ranking of all the genes, without regard to a significance
 528 cutoff. First, the genes are sorted by LFC. Alternatively, they can be ranked by the signed
 529 log-P-value ($SLPV = sign(LFC) * log_{10}(pval)$), which effectively ranks the genes by signifi-
 530 cance from largest increase in insertion counts to largest decrease (with insignificant genes
 531 falling in the middle of the ranking). Then the GSEA algorithm calculates a score reflecting
 532 the mean rank of a given set of genes and performs a simulation to determine the signifi-
 533 (p-value) by comparing to a null distribution of scores derived from random shuffling of the
 534 order. The closer the mean rank of a group of genes is to the top of the entire ranked list,
 535 or the closer to the bottom, the more significant. The potential advantage of GSEA over
 536 Fisher’s exact test is that all the genes in a pathway can contribute to its enrichment, even if
 537 only a few (or none) are above the significance cutoff, as long as there is a systematic trend
 538 of increased or decreased counts shared by many of them.
- 539 • Finally, the Ontologizer method [32] has been implemented and is available via the ‘-M ONT’
 540 flag. The Ontologizer method acknowledges the hierarchical nature of the Gene Ontology and
 541 is designed to take advantage of parent-child relationships among GO terms by computing a
 542 conditional version of Fisher’s exact test for a node conditioned on the genes in its parents.
 543 This can help focus the analysis on nodes in the GO hierarchy showing the most specific
 544 enrichment.

545 Here are some examples of using these various options.

```
546 # uses Fisher’s exact test by default (with PC=2 as pseudocounts)
547 > transit pathway_enrichment resampling_glyc_chol.txt $DATA/H37Rv_sanger_roles.dat
548 $DATA/sanger_roles.dat pathways_glyc_chol_Sanger.txt
549
550 # can do this with GO terms too
551 > transit pathway_enrichment resampling_glyc_chol.txt $DATA/H37Rv_GO_terms.txt
552 $DATA/GO_term_names.dat pathways_glyc_chol_GO.txt
553
554 # with COG categories
555 > transit pathway_enrichment resampling_glyc_chol.txt $DATA/H37Rv_COG_roles.dat
556 $DATA/COG_roles.dat pathways_glyc_chol_COG.txt
557
558 # can also do GSEA method (on any system of functional categories)
559 > transit pathway_enrichment resampling_glyc_chol.txt $DATA/H37Rv_sanger_roles.dat
560 $DATA/sanger_roles.dat pathways_Sanger_GSEA.txt -M GSEA
561
562 # Ontologizer is a specialized method for GO terms
563 > transit pathway_enrichment resampling_glyc_chol.txt $DATA/H37Rv_GO_terms.txt
564 $DATA/GO_term_names.dat pathways_Ontologizer.txt -M ONT
```

565 where \$DATA refers to the path to the Transit data directory noted above. In this dataset,
 566 2 COG and 6 Sanger categories related to secondary-metabolite/small-molecule metabolism and

567 lipid metabolism are identified as significant. The most significant GO term identified (among
568 26 significant terms, though many overlap) is GO:0008202, ‘steroid metabolic process’ ($P_{adj} =$
569 0.0000089; 15 out of 75 conditionally essential genes from resampling), which is consistent with the
570 experiment (growth on cholesterol versus glycerol).

571 3.4 Analyses of Multiple Conditions

572 Recently, researchers have begun conducting more complex experiments using Tn libraries where a
573 library is assessed across a large number of treatment conditions. For example, one might evaluate
574 responses (conditional essentiality) to treatment with a panel of different antibiotics, possibly at
575 different concentrations. One might assess the library in different stress conditions (heat, cold,
576 hypoxia, SDS, nutrient starvation, iron limitation, etc.), or with media supplemented in various
577 ways (e.g. multiple sources of carbon, nitrogen, sulfur, or phosphorous). Or one might compare
578 survival of a bacterial transposon mutants in animal models with different genetic backgrounds,
579 or for different durations of infection, etc. Analysis of such TnSeq datasets goes well beyond
580 simple pairwise comparisons between conditions and often requires customized statistical analysis
581 to evaluate the effects of the experimental variables. As a first step, Transit has some tools for
582 evaluating variability of insertion counts across conditions. The analysis below focus on identifying
583 genes that exhibit some statistically significant differences in insertion counts across the panel of
584 conditions. This is a useful starting point to begin to assess effects of the treatments (and similarities
585 among them) based on the subsets of genes that respond.

586 In this section, we use an example from a study of iron utilization in mycobacteria [33]. The
587 data includes 2-3 replicates each of an *M. tuberculosis* H37Rv transposon library grown in in-vitro
588 conditions involving several different vehicles for iron delivery, focusing on various forms of heme
589 and mycobactin (6 conditions). The 14 wig files for this example have already been consolidated
590 into a combined wig file (iron_combined_wig4.txt), and a samples metadata file is used to encode
591 which samples belong to which condition.

592 3.4.1 Genetic Interaction Analysis

593 A special case of multi-condition analysis that occurs frequently is when TnSeq libraries in a wild-
594 type strain and a mutant strain (e.g. gene-knockout) are compared between two conditions (e.g. a
595 stress and a control). Typically, one is looking for genes that are conditionally essentially in the
596 stress condition, but one wants to factor out those genes with a similar response in the wild-type
597 strain and focus on differences unique to the mutant. There are many variations of this scheme
598 which also require a comparison of 4 TnSeq datasets, arranged as 2×2 . Transit has a specialized
599 method based on Bayesian analysis [34] to identify significantly interacting genes (that interact
600 with the knockout in the context of the stress). An example of how to run this command is as
601 follows, which takes 4 groups of comma-separated wig files as input:

```
602 > python3 ../../transit/src/transit.py GI  
603     <wigs_for_strA_cond1> <wigs_for_strA_cond2> <wigs_for_strB_cond1> <wigs_for_strB_cond2>  
604     <annotation .prot_table or GFF3> <output file>
```

605 In the output file, significant genes are categorized as ‘aggravating’, ‘alleviating’, or ‘suppressive’
606 interactions, depending on whether they exhibit increased or decreased insertion counts in the
607 mutant compared to the wild-type in the stress condition.

608 3.4.2 ANOVA

609 ANOVA analysis in Transit can be used to identify genes that exhibit significant *variability* of inser-
610 tion counts across multiple conditions. ANOVA is a traditional statistical method for determining
611 whether there are differences in observations among several groups. This method can be run in
612 Transit using the following command:

```
613 > transit anova --help
614 Usage: python3 transit.py anova <combined wig file> <samples_metadata file>
615         <annotation .prot_table> <output file> [Optional Arguments]
616 Optional Arguments:
617 -n <string> := Normalization method. Default: -n TTR
618 --include-conditions <cond1,...> := Comma-sep list of conds to use for analysis
619 --ignore-conditions <cond1,...> := Comma-sep list of conds to ignore
620 -iN <int> := Ignore TAs within given percentage of N terminus. Def: -iN 0
621 -iC <int> := Ignore TAs within given percentage of C terminus. Def: -iC 0
622 -PC <int> := pseudocounts to use for calculating LFC. Default: -PC 5
```

623 For example,

```
624 > transit anova iron_combined_wig4.txt iron_samples_metadata.txt
625         H37Rv.prot_table anova_iron.txt
```

626 The use of a combined_wig file makes it easy to work with a large number of datasets in
627 this context. The samples metadata file is a spreadsheet (in tab-separated text format) prepared
628 by users that contains information about each sample/dataset/wig file. The headers must con-
629 tain ‘Id’, ‘Filename’, and ‘Condition’, where Id is a unique name for each sample, Filename is
630 the name of its wig file (incorporated in the combined_wig file) and Condition is a symbolic
631 name used to represent the treatment (typically represented by multiple replicates). The sam-
632 ples metadata file can contain additional information as well (e.g. time points, concentrations,
633 batches, etc.). During the ANOVA analysis, the (TTR-normalized) insertion counts for the TA
634 sites in each gene will be pooled into groups based on the condition labels, before computing
635 the F-statistic and p-value. P-values are adjusted post-hoc by the Benjamini-Hochberg method.
636 Flags `--ignore-conditions` and `--include-conditions` can be used to focus on the analy-
637 sis on just a subset of desired conditions (provided as a comma-separated list). For example
638 `--include-conditions HighFeMBT,Hemin,Hemoglobin` or `--ignore-conditions LowFeMBT`.

639 The output of ANOVA analysis is a tab-separated spreadsheet which can be sorted by adjusted
640 P-value. In this data, 181 genes were found to exhibit significant variability in insertion counts
641 among the 6 iron-supplementation conditions. If a gene is determined to be significant by ANOVA,
642 it only means that its counts vary in some condition (at least one) relative to the others, but does
643 not indicate which one. Like traditional ANOVA analysis, post-hoc analyses must be employed
644 to determine which conditions a gene is responding to, such as Tukey’s range-test (or honestly
645 significant difference), which is based on pairwise comparisons between conditions [35].

646 To facilitate this post-hoc analysis, the ANOVA method in Transit also prints out LFCs for each
647 gene in each condition. The LFCs can be used to look for genes that respond to specific conditions,
648 e.g. by sorting on these columns to look for genes with the most enrichment or depletion in a given
649 condition. Note that *LFCs are computed relative to the mean insertion count for a gene across all*
650 *conditions*. Thus, there will almost always be some condition(s) with higher counts (representing
651 more fitness for the mutant) and other condition(s) with lower counts (where disruption of gene

652 has less fitness, and hence is relatively more essential) than average. Pseudocounts of 5 are used in
 653 calculating the LFCs (which helps reduce high-magnitude LFCs for genes with low counts, which
 654 are susceptible to noise), though this can be changed with the -PC flag.

655 The LFC columns in the ANOVA output can be colored as a heatmap in the spreadsheet (using
 656 Excel) to make the patterns of variation among the genes more clear. In addition, the ANOVA
 657 output file can be used to generate a heatmap (see Figure 1, adapted from [33]) that simultaneously
 658 clusters the significant genes and the conditions, which is especially useful for shedding light on the
 659 relationships among the conditions apparent in the data:

```
660 > transit heatmap -anova anova_iron.txt heatmap_iron.png
```

661 Similarly, the anova file can be used as input the corrplot command (with the '-anova' flag suffixed)
 662 to show the similarities among the condition.

```
663 > transit corrplot anova_iron.txt heatmap_iron.png -anova
```

664 Importantly, the heatmap and corrplot analyses are based only on the significantly varying genes
 665 ($P_{adj} < 0.05$, typically only a few hundred) in order to enhance the patterns, since otherwise they
 666 would be washed out by the rest of the genes in the genome, the majority of which do not exhibit
 667 significant variation.

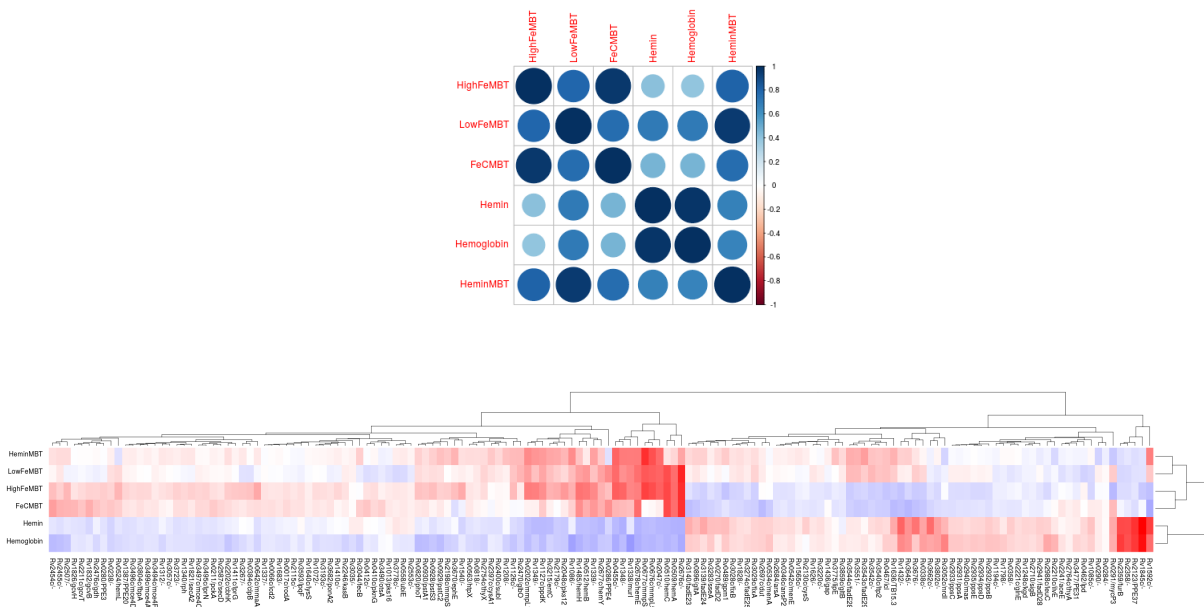


Figure 1: a) Correlation plot among iron-supplementation conditions based on significantly varying genes according to ANOVA. b) Heatmap of same data showing clustering of genes and conditions. Blue means more insertions than average (i.e. less essential), and red means less insertions than average (i.e. more essential).

668 3.4.3 Zero-Inflated Negative Binomial (ZINB)

669 An alternative method for identifying genes exhibiting variability of insertion counts across con-
 670 ditions is Zero-Inflated Negative Binomial (ZINB) regression, which has recently been added to

671 Transit [36]. One of the limitations of ANOVA analysis is that it assumes the data are Normally
672 distributed. However, transposon insertion counts clearly violate this assumption (for several rea-
673 sons). The ZINB method is designed to be a better model for insertion count data by: a) represent-
674 ing the non-zero counts using a Negative Binomial distribution, and b) representing the saturation
675 (TA sites with counts of 0) using a zero component in a mixture model. Thus, when analyzing the
676 counts in a gene across multiple conditions, ZINB can detect variations in either the magnitudes in
677 the insertion counts or the local level of saturation. This enables ZINB to be a little more sensitive
678 than ANOVA, and empirical studies suggest ZINB can detect about 30-50% more significant genes.
679 The usage of the ZINB command in Transit is as follows:

```
680 > transit zing --help
681 Usage: transit zinb <combined wig file> <samples_metadata file> <annotation .prot_table>
682           <output file> [Optional Arguments]
683 Optional Arguments:
684 -n <string>      := Normalization method. Default: -n TTR
685 --condition      := column name in samples_metadata to use. Default: "Condition"
686 --ignore-conditions <cond1,...> := Comma separated list of conditions to ignore
687 --include-conditions <cond1,...> := Comma separated list of conditions to include
688 -iN <float>      := Ignore TAs within given percentage of the N terminus. Def: 5
689 -iC <float>      := Ignore TAs within given percentage of the C terminus. Def: 5
690 -PC <N>          := Pseudocounts used in calculating LFCs in output file. Default: -PC 5
691 --covars ... := Comma-sep. list of variables (columns in metadata) to use as covariates
692 --interactions ... := Comma-sep. list of variables to test for interactions
693 --gene <ORF id or gene name> := Run method for just one gene and print model output.
```

694 For example,

```
695 > transit zinb iron_combined_wig4.txt iron_samples_metadata.txt
696           H37Rv.prot_table zinb_iron.txt
```

697 If there are just two conditions provided, then using ZINB analysis to identify genes with
698 variable insertion counts is equivalent to detecting conditional essentiality. Hence ZINB can be
699 viewed as an alternative to resampling in this limiting case, and anecdotal testing suggests that
700 there is a great deal of overlap. ZINB can sometimes identify additional genes as conditionally
701 essential where there is a difference in either local saturation or magnitudes of counts alone, even if
702 the overall mean count between conditions is not significantly different, and hence not detected by
703 resampling. (see [36] for a more thorough comparison of the differences in significant hits between
704 ZINB and resampling on example datasets).

705 In the output file for ZINB, multiple columns of information are provided on the mean counts in
706 each condition, LFCs (relative to the mean across all conditions), non-zero mean, and saturation.
707 For convenience, the `-gene` flag can be used to run the analysis just on a single gene, and the
708 Transit will print out the condition-dependent and condition-independent models, likelihoods, etc.,
709 as in this example:

```
710 > transit zinb iron_combined_wig4.txt iron_samples_metadata.txt
711           H37Rv.prot_table output.txt --gene glpK
```

712 The significance (p-value) of each gene is determined by a likelihood ratio test (LRT). First, a
713 condition-dependent model is generated by fitting independent parameters for each condition (such
714 as mean and dispersion of insertion counts in the Negative Binomial, and saturation as the mixing

715 coefficient with the zero component). Then a condition-independent ZINB model is generated by
716 fitting a common set of parameters based on the counts pooled across all conditions. Finally, a
717 likelihood ratio test is performed to determine whether the increase in likelihood of the condition-
718 dependent model is justified, given the increased number of parameters. A p-value is derived
719 for each gene using a chi-square distribution, and then the p-values are adjusted post-hoc by the
720 Benjamini-Hochberg procedure to control the overall FDR.

721 One of the advantages of the ZINB model is that it is implemented in a Generalized Linear
722 Model (GLM) framework, which allows incorporation of experimental variables (attributes of the
723 conditions) as covariates. One application of this idea is to factor out the effect of an attribute that
724 is known to affect the main condition in a way that is not of interest. For example, suppose we
725 are interested in identifying genes that exhibit variable responses to different antibiotics. Further,
726 suppose the samples were each cultured for varying amounts of time (e.g. 0, 1, or 2 weeks). It is
727 natural to expect that there will be variation in insertion counts between samples at different time-
728 points, even if they are treated with the same drug. To evaluate the effect of the drug, it is desirable
729 to subtract out any systematic effects on the insertion counts due to time (independent of drug). If
730 a “Time” column is included in the samples metadata file, and a column with the header “Drug”
731 encodes the drug treatment for each sample, then this may be achieved by using the `--covars` flag,
732 as in this hypothetical example:

```
733 > transit zinb combined_wig.txt samples_metadata.txt H37Rv.prot_table  
734 ZINB_cond_drug_covar_time.txt --condition Drug --covars Time
```

735 If there are multiple covariates, they can be specified using a comma-separated list (i.e. corre-
736 sponding to separate columns in the metadata file). The `--covars` flag could also potentially be
737 used to correct for batch effects (where the variation in insertion counts in some samples appear to
738 be determined by the batch of the experiment or data collection).

739 A similar approach can be used to test for interactions of variables with the main condition.
740 For example, suppose we are interested in genes that respond differentially to a panel of drugs.
741 Furthermore, suppose TnSeq data was collected for cultures grown on media containing one of
742 several carbon sources, e.g. glycerol, glucose, or cholesterol. In order to test whether carbon source
743 interacts with drug, ZINB will fit a model based on the cross-product of all combinations of the
744 two variables and compare it (using an LRT) to a condition-independent model (where the counts
745 are pooled for the main condition). Assuming there is a column with the header “CarbonSource”
746 in the samples metadata, the interaction may be tested as in the following hypothetical example:

```
747 > transit zinb combined_wig.txt samples_metadata.txt H37Rv.prot_table  
748 ZINB_cond_drug_interac_carbon.txt  
749 --condition Drug --interactions CarbonSource
```

750 In the resulting output file, significant genes are those that exhibit some variability among the
751 drug treatments that is dependent on carbon source.

752 As with ANOVA, the ZINB output file can be used to make a heatmap showing the clustering
753 of the significant genes and the conditions:

```
754 > transit heatmap -zinb zinb_iron.txt zinb_iron_heatmap.png
```

755 4 Summary

756 Transit is designed to be a platform for statistical analysis of TnSeq data, with a focus on analysis
757 of Himar1 transposon libraries (where insertions are restricted to TA dinucleotides). Some of the

758 analytical methods can be applied to other transposons, like Tn5, though currently, they don't
759 work as robustly. Although Transit has a graphical interface (GUI), some of the more recent tools
760 that have been added can only be invoked at the command line, which has been the focus of this
761 paper. The Transit Pre-Processor (TPP) provides a way of processing raw sequencing data files
762 and reducing the raw data to TA-site insertion counts in the form of .wig files. The 'tseq_stats'
763 command provides important summary statistics on datasets which is useful for diagnostics (i.e.
764 identifying poor-quality datasets that might need to be excluded or re-collected). The analytical
765 tools can be divided into 3 major tasks. First, individual datasets from an organism in a single
766 (e.g. reference) condition can be used to identify essential genes using methods such as Gumbel
767 analysis (gaps) or a Hidden Markov Model. Second, pairs of conditions can be compared to identify
768 conditionally essential genes using resampling (a permutation test on mean counts). The recently
769 added ability to apply resampling on datasets mapped to different genome sequences has proven
770 useful for studying differences in gene essentiality between libraries made from different strains,
771 such as clinical isolates. Several methods for pathway enrichment analysis have also been added to
772 gain additional insight from functional similarities among conditionally essential genes.

773 However, more recent developments in Transit have focused on supplying tools for analyzing
774 larger collections of datasets from experiments involving multiple conditions. To keep things man-
775 ageable when working with large collections of datasets, many of the tools in Transit have been
776 extended to use 'combined_wig' files and accompanying metadata files that encode relevant infor-
777 mation about the different conditions. A starting point for analyzing such complex experiments is
778 to identify genes exhibiting statistically significant variability across the conditions, using ANOVA
779 or ZINB analysis, and then to begin to cluster and assess genes based on the similarity of their
780 patterns of count variations, utilizing correlation plots and heatmaps. ZINB can be used to perform
781 more sophisticated analyses through the exploitation of variables relating the different conditions
782 as covariates and/or interactions (including capturing the trend or dependence of insertion counts
783 on quantitative variables such as time or concentration). Genetic interaction (GI) analysis can be
784 used to evaluate experiments where two different experimental variables are evaluated, producing
785 a 2x2=4-way comparison of conditions, and test for significant interactions (e.g. suppressive, alle-
786 viating, or aggravating), which is especially useful for identifying genes associated with phenotypic
787 changes in a knock-out strain compared to a wild-type strain.

788 Transit continues to evolve and improve, especially through feedback and suggestions from
789 users (send email to ioerger@cs.tamu.edu). In the future, we hope to add new statistical methods
790 to support analysis of more complex experiments, improve integration with the GUI, and also
791 extend and improve the analyses to TnSeq libraries made with other transposons, especially Tn5.

792 Acknowledgements

793 This work has been supported by NIH grants U19 AI107774 and P01 AI143575.

794 References

- 795 1. van Opijnen T, Camilli A (2013) Transposon insertion sequencing: a new tool for systems-level
796 analysis of microorganisms. *Nat Rev Microbiol* 11(7):435–442
- 797 2. Barquist L, Boinett CJ, Cain AK (2013) Approaches to querying bacterial genomes with
798 transposon-insertion sequencing. *RNA Biol* 10(7):1161–1169

- 799 3. Chao M, S A, Davis B, Waldor M (2016) The design and analysis of transposon insertion
800 sequencing experiments. *Nat Rev Microbiol* 14(2):119128
- 801 4. Langridge GC, Phan M, Turner D, Perkins T, Parts L, Haase J, Charles I, Maskell
802 D, Peters S, Dougan G, et al (2009) Simultaneous assay of every salmonella typhi
803 gene using one million transposon mutants. *Genome Research* 19(12):2308–2316, URL
804 <http://www.ncbi.nlm.nih.gov/pubmed/19826075>
- 805 5. Gawronski JD, Wong SM, Giannoukos G, Ward DV, Akerley BJ (2009) Tracking insertion
806 mutants within libraries by deep sequencing and a genome-wide screen for *Haemophilus* genes
807 required in the lung. *Proc Natl Acad Sci USA* 106(38):16422–16427
- 808 6. Goodman AL, McNulty NP, Zhao Y, Leip D, Mitra RD, Lozupone CA, Knight R, I GJ (2009)
809 Identifying genetic determinants needed to establish a human gut symbiont in its habitat. *Cell*
810 *Host Microbe* 6(3):279–89
- 811 7. Wetmore K, Price M, Waters R, Lamson J, He J, Hoover C, Blow M, Bristow J, Butland G,
812 Arkin A, A D (2015) Rapid quantification of mutant fitness in diverse bacteria by sequencing
813 randomly bar-coded transposons. *mBio* 6(3):e00306–15
- 814 8. Jensen P, Zhu Z, van Opijnen T (2017) Antibiotics disrupt coordination between transcriptional
815 and phenotypic stress responses in pathogenic bacteria. *Cell Reports* 20(7):1705–1716
- 816 9. Zhang YJ, Reddy MC, Ioerger TR, Rothchild AC, Dartois V, Schuster BM, Trauner A, Wallis
817 D, Galaviz S, Huttenhower C, Sacchettini JC, Behar SM, J RE (2013) Tryptophan biosynthesis
818 protects mycobacteria from CD4 T-cell-mediated killing. *Cell* 155(6):1296–308
- 819 10. Luo H, Lin Y, Gao F, Zhang CT, Zhang R (2014) DEG 10, an update of the Database of
820 Essential Genes that includes both protein-coding genes and non-coding genomic elements.
821 *Nucleic Acids Research* 42:D574–D580
- 822 11. Rubin EJ, Akerley BJ, Novik VN, Lampe DJ, Husson RN, Mekalanos JJ (1999) In vivo transpo-
823 sition of mariner-based elements in enteric bacteria and mycobacteria. *PNAS* 96(4):1645–1650
- 824 12. Sassetti CM, Boyd DH, Rubin EJ (2001) Comprehensive identification of con-
825 ditionally essential genes in mycobacteria. *PNAS* 98(22):12712–12717, DOI
826 10.1073/pnas.231275498, URL <http://www.pnas.org/content/98/22/12712.abstract>,
827 <http://www.pnas.org/content/98/22/12712.full.pdf+html>
- 828 13. Reznikoff WS (2003) Tn5 as a model for understanding DNA transposition. *Molecular Micro-*
829 *biology* 43(5):1199–1206
- 830 14. Lampe DJ, Churchill ME, Robertson HM (1996) A purified mariner transposase is sufficient
831 to mediate transposition in vitro. *The European Molecular Biology Organization Journal*
832 15(19):5470–5479
- 833 15. Long J, DeJesus M, Ward D, Baker R, Ioerger T, Sassetti C (2015) Identifying essential genes in
834 *Mycobacterium tuberculosis* by global phenotypic profiling. In: Lu LJ (ed) *Methods in Molecular*
835 *Biology: Gene Essentiality*, Springer, vol 1279, pp 79–95
- 836 16. DeJesus MA, Ioerger TR (2016) Normalization of transposon-mutant library sequencing
837 datasets to improve identification of conditionally essential genes. *J Bioinform Comput Biol*
838 p 1642004

- 839 17. Zomer A, Burghout P, Bootsma HJ, Hermans PW, van Hijum SA (2012) ESSENTIALS: soft-
840 ware for rapid analysis of high throughput transposon insertion sequencing data. PLoS ONE
841 7(8):e43012
- 842 18. Solaimanpour S, Sarmiento F, Mrazek J (2015) Tn-seq explorer: a tool for analysis of high-
843 throughput sequencing data of transposon mutant libraries. PLoS ONE 10(5):e0126070
- 844 19. Barquist L, Mayho M, Cummins C, Cain AK, Boinett CJ, Page AJ, Langridge GC, Quail MA,
845 Keane JA, Parkhill J (2016) The tradis toolkit: sequencing and analysis for dense transposon
846 mutant libraries. *Bioinformatics* 32(7):1109
- 847 20. Zhao L, Anderson MT, Wu W, T Mobley HL, Bachman MA (2017) TnseqDiff: identification of
848 conditionally essential genes in transposon sequencing studies. *BMC Bioinformatics* 18(1):326
- 849 21. van Opijnen T, Bodi KL, Camilli A (2009) Tn-seq: high-throughput parallel sequencing for
850 fitness and genetic interaction studies in microorganisms. *Nat Methods* 6(10):767–772
- 851 22. Pritchard JR, Chao MC, Abel S, Davis BM, Baranowski C, Zhang YJ, Rubin EJ, Waldor MK
852 (2014) ARTIST: high-resolution genome-wide assessment of fitness using transposon-insertion
853 sequencing. *PLoS Genet* 10(11):e1004782
- 854 23. DeJesus MA, Ambadipudi C, Baker R, Sasseti C, Ioerger TR (2015) TRANSIT—A Software
855 Tool for Himar1 TnSeq Analysis. *PLoS Comput Biol* 11(10):e1004401
- 856 24. Griffin JE, Gawronski JD, DeJesus MA, Ioerger TR, Akerley BJ, Sasseti CM (2011) High-
857 resolution phenotypic profiling defines genes essential for mycobacterial growth and cholesterol
858 catabolism. *PLoS Pathog* 7(9):e1002251, DOI 10.1371/journal.ppat.1002251
- 859 25. Li H, Durbin R (2009) Fast and accurate short read alignment with Burrows-Wheeler transform.
860 *Bioinformatics* 25(14):1754–1760
- 861 26. DeJesus MA, Zhang YJ, Sasseti CM, Rubin EJ, Sacchettini JC, Ioerger TR (2013) Bayesian
862 analysis of gene essentiality based on sequencing of transposon insertion libraries. *Bioinformat-
863 ics* 29(6):695–703
- 864 27. DeJesus MA, Ioerger TR (2013) A Hidden Markov Model for identifying essential and growth-
865 defect regions in bacterial genomes from transposon insertion sequencing data. *BMC Bioinform-
866 atics* 14:303
- 867 28. Cole ST, Brosch R, Parkhill J (1998) Deciphering the biology of mycobacterium
868 tuberculosis from the complete genome sequence. *Nature* 393(6685):537–544, URL
869 <http://dx.doi.org/10.1038/31159>
- 870 29. Galperin MY, Makarova KS, Wolf YI, Koonin EV (2015) Expanded microbial genome coverage
871 and improved protein family annotation in the COG database. *Nucleic Acids Research* 43:D261–
872 9
- 873 30. Irizarry RA, Wang C, Zhou Y, Speed TP (2009) Gene set enrichment analysis made simple.
874 *Stat Methods Med Res* 18(6):565–575
- 875 31. Subramanian A, Tamayo P, Mootha VK, Mukherjee S, Ebert BL, Gillette MA, Paulovich
876 A, Pomeroy SL, Golub TR, Lander ES, Mesirov JP (2005) Gene set enrichment analy-
877 sis: a knowledge-based approach for interpreting genome-wide expression profiles. *PNAS*
878 102(43):15545–50

- 879 32. Grossmann S, Bauer S, Robinson PN, Vingron M (2007) Improved detection of overrepresenta-
880 tion of Gene-Ontology annotations with parent-child analysis. *Bioinformatics* 23(22):3024-3031
- 881 33. Zhang L, Hendrickson RC, Meikle V, Lefkowitz EJ, Ioerger TR, Niederweis M (2020)
882 Comprehensive analysis of iron utilization by *Mycobacterium tuberculosis*. *PLoS Pathogens*
883 16(3):e1008337
- 884 34. DeJesus MA, Nambi S, Smith CM, Baker RE, Sassetti CM, Ioerger TR (2017) Statistical
885 analysis of genetic interactions in Tn-Seq data. *Nucleic Acids Res* 45(11):e93
- 886 35. Tukey J (1949) Comparing individual means in the analysis of variance. *Biometrics* 5(2):99-114
- 887 36. Subramaniam S, DeJesus MA, Zaveri A, Smith CM, Baker RE, Ehrt S, Schnappinger D,
888 Sassetti CM, Ioerger TR (2019) Statistical analysis of variability in tnsq data across conditions
889 using zero-inflated negative binomial regression. *BMC Bioinformatics* 20(1):603
- 890 37. Xu W, DeJesus MA, Rcker N, Engelhart CA, Wright MG, Healy C, Lin K, Wang R, Park
891 SW, Ioerger TR, Schnappinger D, S E (2017) Chemical genetic interaction profiling reveals
892 determinants of intrinsic antibiotic resistance in *Mycobacterium tuberculosis*. *Antimicrobial*
893 *Agents Chemotherapy* 61(22):e01334-17
- 894 38. Matern WM, Rifat D, Bader JS, Karakousis PC (2018) Gene enrichment analysis reveals major
895 regulators of *Mycobacterium tuberculosis* gene expression in two models of antibiotic tolerance.
896 *Front Microbiol* 9:610
- 897 39. Kieser KJ, Boutte CC, Kester JC, Baer CE, Barczak AK, Meniche X, Chao MC, Rego EH,
898 Sassetti CM, Fortune SM, Rubin EJ (2015) Phosphorylation of the Peptidoglycan Synthase
899 PonA1 Governs the Rate of Polar Elongation in *Mycobacteria*. *PLoS Pathog* 11(6):e1005010
- 900 40. Nambi S, Long JE, Mishra BB, Baker R, Murphy KC, Olive AJ, Nguyen HP, Shaffer SA,
901 Sassetti CM (2015) The Oxidative Stress Network of *Mycobacterium tuberculosis* Reveals Co-
902 ordination between Radical Detoxification Systems. *Cell Host Microbe* 17(6):829-837
- 903 41. Baranowski C, Welsh MA, Sham LT, Eskandarian HA, Lim HC, Kieser KJ, Wagner JC, McK-
904 inney JD, Fantner GE, Ioerger TR, Walker S, Bernhardt TG, Rubin EJ, H RE (2018) Maturing
905 *Mycobacterium smegmatis* peptidoglycan requires non-canonical crosslinks to maintain shape.
906 *Elife* p e37516
- 907 42. Fu Y, Waldor M, Mekalanos J (2013) Tn-seq analysis of *Vibrio cholerae* intestinal colonization
908 reveals a role for t6ss-mediated antibacterial activity in the host. *Cell Host Microbe* 14(6):652-
909 63
- 910 43. Dragset MS, Ioerger TR, Loevenich M, Haug M, Sivakumar N, Marstad A, Cardona PJ,
911 Klinkenberg G, Rubin EJ, Steigedal M, Flo TH (2019) Global assessment of *Mycobacterium*
912 *avium* subsp. *hominissuis* genetic requirement for growth and virulence. *mSystems* pp e00402-
913 19
- 914 44. DeJesus MA, Gerrick ER, Xu W, Park SW, Long JE, Boutte CC, Rubin EJ, Schnappinger D,
915 Ehrt S, Fortune SM, Sassetti CM, Ioerger TR (2017) Comprehensive Essentiality Analysis of
916 the *Mycobacterium tuberculosis* Genome via Saturating Transposon Mutagenesis. *MBio* 8(1)
- 917 45. Lampe DJ, Grant TE, M RH (1998) Factors affecting transposition of the Himar1 mariner
918 transposon in vitro. *Genetics* 149(1):179-87

- 919 46. Ason B, Reznikoff WS (2004) DNA sequence bias during Tn5 transposition. *Journal of Molec-*
920 *ular Biology* 335:1213–1225
- 921 47. Chao MC, Pritchard JR, Zhang YJ, Rubin EJ, Livny J, Davis BM, Waldor MK (2013) High-
922 resolution definition of the *Vibrio cholerae* essential gene set with hidden Markov model-based
923 analyses of transposon-insertion sequencing data. *Nucleic Acids Res* 41(19):9033–9048
- 924 48. Pickands J (1975) Statistical inference using extreme order statistics. *Annals of Statistics* 3:119–
925 131
- 926 49. Warr AR, Hubbard TP, Munera D, Blondel CJ, zur Wiesch PA, Abel S, Wang X, Davis BM,
927 Waldor MK (2019) Transposon-insertion sequencing screens unveil requirements for ehec growth
928 and intestinal colonization. *PLoS Pathogens* 15(8):e1007652
- 929 50. Benjamini Y, Yekutieli D (2005) False discovery rate controlling confidence intervals for selected
930 parameters. *Journal of the American Statistical Association* 100(469):71–81