Supplementary Software

Contents

1 Software packages

In this paper, we developed the *Expedition* suite, consisting of software packages that addressed three key deficiencies in single-cell alternative splicing analysis:

1. **Detect and quantify alternative splicing quickly, with minimum false positives: `outrigger`, Section 1.1**
   In single-cell analysis, absolute quantitation of gene expression or "percent spliced-in" (Psi/Ψ) is important and enable us to learn the distribution of these quantitations. Previously, relative quantitation for splicing ($\Delta\Psi$) is more commonly used to calculate the difference between groups. Such relative quantitation tolerates false positive better, as false positives may not vary between groups, $\Delta\Psi \sim 0$ and are thus not noticeable in pairwise comparisons. However, when studying distribution of absolute quantitation, such false positives obsure the observation in unpredictable way and hinder biological interpretation. The second main problem of previous splicing algorithm is the inflexible definitions of alternative exons. The same alternative exons may utilize different flanking exons in different cells/samples, thus leading to different biological interpretation. To address these problems, we create `outrigger`, which uses junction reads to find de novo exons, creates a splice graph to define junction-based alternative events, filters for conserved splice sites, and strictly rejectes cases of alternative events incompatible with the data at hand. Finally, we discuss and compare to the popular MISO[8] algorithm.

2. **Classify modalities of alternative splicing events, including bimodal: `anchor`, Section 1.2**
   The power of single-cell analyses rises from the ability to study the distribution of a parameter-of-interest. There are a few statistical methods for finding bimodal distributions, but none are sufficient because they are either not sensitive enough, or not robust enough to noise. Additionally, these methods only deal with bimodal distribution and do not classify other distributions, such as unimodal or multimodal. To create a sensitive distribution classifier for all modalities, we used Bayesian methods to create `anchor`, and compare our method to a simple binning method, the bimodality index[17], and the bimodal dip test[7].

3. **Quantify and visualize dynamics in distributions: `bonvoyage`, Section 1.3**
   While there are many statistical tests to compare changes in distributions, few of them is coupled with visual tools

to present changes in distribution with both magnitude and direction. For the specific question of alternative splicing changes, we are interested in observing a event becomes more included or more excluded. Thus we have employed machine learning methods to create a visualizable, interpretable 2d space with "included" and "excluded" axes. This method is compared to the quantification offered by the Jensen-Shannon Divergence (JSD)[2].

1.1 `outrigger`: Splicing estimation with *de novo* annotation and graph traversal

Currently available tools for AS detection and quanitification have two major problems: (1) inflexible definitions that cannot handle different configurations of flanking exons for the same alternative junctions, and (2) lack of rejection of an alternative event even if its definition is incompatible with the data-at-hand. The first problem is solved with `outrigger index`, which defines all potential alternative events based on the junctions and alternative exons from the aggregate of entire sample sets in a given project, and enumerates all biologically possible flanking exon combinations. This step maximize the likelihood to identify all possible alternative events. To ensure only valid alternative events were generated, we added `outrigger validate` to remove alternative events with introns lacking conserved splice sites. The second prolbem is solved with `outrigger psi`, which applies strict rules to only permit junctions with sufficient coverage for an event in a given sample. All the parameters in the rules can be user-defined. Thus, outrigger addresses key issues with current alternative splicing software.

1.1.1 Algorithm overview

Broadly, the goal of `outrigger` is to create a custom, *de novo* alternative splicing annotation by using junction reads and exon definitions to create a exon-junction graph, traversing the graph to find alternative events, and calculate percent spliced-in (Psi/Ψ) of the alternative exons.

1.1.1.1 `outrigger index`: Create custom alternative splicing annotation. The following is a narrative describing **Supplementary Software Figure 2A**.

1.1.1.1.1 Inputs. Two inputs are required for `outrigger index`: junction counts and gene annotations. The junction counts can be provided in many forms: either `.bam`[6] genome alignment files, splice junction count `.SJ.out.tab` files created by the STAR aligner[4], or a pre-compiled table of samples' junction reads in a `.csv` format. The gene annotations can be provided in `.gtf` or `.gff` format.

1.1.1.1.2 Step 1: Retain junctions from each cell with sufficient read depth. Junctions with reads in an individual sample less than the minimum number of reads, $r_{\min}$ are removed. By default, $r_{\min} = 10$, and can be adjusted by the user, for example to a minimum of 88 reads, with `--min-reads 88` on the command line. To illustrate, if one junction is observed with two (2) reads in 100 samples, although there were a total of 200 reads observed on the junction, it will be discarded at this step. Because, there is not sufficient evidence to suggest that this junction is well-covered in any sample.

1.1.1.1.3 Step 2: Collapse reads on shared exon-exon junctions, across all samples. The aggregate of all junctions from all samples in a given project are create to maximize the likelihood of identifing all potential alternative events.

1.1.1.1.4 Step 3: Detect exons *de novo*. If the gap between two junctions is under X nucleotides, an exon will be inserted at the gap. This maximum X is necessary, because otherwise we could insert "exons" that are many kilobases long, but aren't true exons – they are the intergeneic space between genes. By default, $X = 100$, and this can be adjusted by the user, for example to 157 nucleotides, with the command line flag, `--max-de-novo-exon-length 157`.

1.1.1.1.5 Step 4: Integrate exon annotation to obtain pairwise exon-junction relationships. Annotated exons are integrated with the *de novo* exons and create a table of the pairwise relationships of each exon to each junction. We do this by creating a database of genes, transcripts, and exons from a GTF gene annotation file using `gffutils`[3], and observing which junctions are adjacent to each exon. This outputs an *"exon-direction-junction"* table which is used in Step 5.

1.1.1.1.6 Step 5: Combine pairwise relationships to obtain global structure. We then use the adjacencies to build a directional graph which connects exons to each other via junctions. This graph database was built using `graphlite`[1], a Python program that provides a lightweight graph wrapper over SQLite.

1.1.1.1.7 Step 6: Search for alternative exons. To find alternative events, all exons in the graph database were transversed to test, if starting from that exon, it could be a first exon of an skipped exon (SE) or mutually exclusive exon (MXE) event.

1.1.1.1.8 Outputs. The output of `outrigger index` is a folder containing the following. The `events.csv` file contains the event definitions will be used by `outrigger psi`. The `exonN.bed` files, where `N` is an exon number, will be used by `outrigger validate` to check for canonical or non-canonical splice sites.

The splicing event definitions in the `events.csv` files are specified by the junctions and the alternative exon. As there may be multiple potential flanking exons with the same junctions, rather than choosing a single version (as is done by MISO, **Supplementary Software Figure 2B**), we output all possible flanking exon configurations. Thus, while the critical alternative exons are exon 2 for SE events and exons 2 and 3 for MXE events, we show all possible exon flanking exon 1s and exon 3s for SE, and all possible flanking exon 1s and exon 4s for MXE events (**Supplementary Software Figure 2A**, lower right).

Below is an example command using `outrigger index`:

```
outrigger index --bam *sorted.bam \
    --gtf /projects/ps-yeolab/genomes/mm10/gencode/m10/gencode.vM10.annotation.gtf
```

This creates a folder called `outrigger_output` with the following contents:

```
outrigger_output/
├── index
│   ├── gtf ..... Added by Step 3
│   │   ├── gencode.vM10.annotation.gtf ..... Added by Step 4
│   │   ├── gencode.vM10.annotation.gtf.db ..... Added by Step 4
│   │   └── novel_exons.gtf ..... Added by Step 3
│   ├── exon_direction_junction.csv ..... Added by Step 4
│   ├── mxe ..... Added by Step 6
│   │   ├── event.bed ..... Added by Step 6
│   │   ├── events.csv ..... Added by Step 6
│   │   ├── exon1.bed ..... Added by Step 6
│   │   ├── exon2.bed ..... Added by Step 6
│   │   ├── exon3.bed ..... Added by Step 6
│   │   ├── exon4.bed ..... Added by Step 6
│   │   └── intron.bed ..... Added by Step 6
│   └── se ..... Added by Step 6
│       ├── event.bed ..... Added by Step 6
│       ├── events.csv ..... Added by Step 6
│       ├── exon1.bed ..... Added by Step 6
│       ├── exon2.bed ..... Added by Step 6
│       ├── exon3.bed ..... Added by Step 6
│       └── intron.bed ..... Added by Step 6
└── junctions ..... Added by Step 1
    ├── metadata.csv ..... Added by Step 2
    └── reads.csv ..... Added by Step 1
```

Supplementary Figure 1: Example output of `outrigger index` command.

Besides outputting the relevant `events.csv` which is used in `outrigger psi` to define events, we also output `.bed` files for the entire event, the alternative intron, and each exon, facilitating downstream sequence analysis.

1.1.1.2 `outrigger validate`: Remove alterantive splicing lacking conserved splice sites. The following describes the biological intuition behind **Supplementary Software Figure 3A**. Major (U2) splicesome recognize splice-sites as ($5'$ end of intron/$3'$ end of intron) `GT/AG` and `GC/AG` the Minor (U12) spliceosome recognizes splice-sites as `AT/AC`[5;10]. By default, these combinations of splice-sties are allowed. But the valid splice sites can be user-specified and changed for example to `AA/AA` and `GG/GG` with `--valid-splice-sites AA/AA,GG/GG`.

The output of `outrigger validate` is a `splice_sites.csv` folder containing the splice sites, and an additional folder in the splice type folder, called `validated`, containing filtered `events.csv` which only contain alternative events with valid splice sites. For example, as a follow up on our previous `outrigger index` command, we validate the alternative exons with the command,

```
outrigger validate --genome mm10 \
    --fasta /projects/ps-yeolab/genomes/mm10/GRCm38.primary_assembly.genome.fa
```

This creates the following additions to the `outrigger_output` folder:

```
outrigger_output/
├── index
│   ├── gtf
│   │   ├── gencode.vM10.annotation.gtf
│   │   ├── gencode.vM10.annotation.gtf.db
│   │   └── novel_exons.gtf
│   ├── exon_direction_junction.csv
│   ├── mxe
│   │   ├── event.bed
│   │   ├── events.csv
│   │   ├── exon1.bed
│   │   ├── exon2.bed
│   │   ├── exon3.bed
│   │   ├── exon4.bed
│   │   ├── intron.bed
│   │   ├── splice_sites.csv ..... Added by outrigger validate
│   │   ├── validated ..... Added by outrigger validate
│   │   │   └── events.csv ..... Added by outrigger validate
│   ├── se
│   │   ├── event.bed
│   │   ├── events.csv
│   │   ├── exon1.bed
│   │   ├── exon2.bed
│   │   ├── exon3.bed
│   │   ├── intron.bed
│   │   ├── splice_sites.csv ..... Added by outrigger validate
│   │   ├── validated ..... Added by outrigger validate
│   │   │   └── events.csv ..... Added by outrigger validate
└── junctions
    ├── metadata.csv
    └── reads.csv
```

Supplementary Figure 2: Example output of `outrigger validate` command.

1.1.1.3 Potential "Franken-events" created by combining junctions over multiple datasets.

As many junctions may occur spuriously in a single cell (sample), aggregating all junctions across all cells (sample) may create events that were not observed in any individual cell (sample) (**Supplementary Software Figure 3B**). We wanted to ensure we strictly defined when events were valid or not in these cases.

In the case of SE events, the exon will have $\Psi = \text{NA}$ for the cell with the observed inclusion junctions, since they don't have sufficient reads on both sides of the exon. For the cell with the exclusion junction, it will have $\Psi = 0$ since no inclusion reads were observed.

For MXE events, if each of the four junctions was observed independently in a different cell, then all of the cells will have $\Psi = \text{NA}$ for that splicing event since there are no cells which have sufficient reads on all junctions of either isoform.

1.1.1.4 `outrigger psi`: Calculate percent spliced-in of alternative exons

To calculate percent spliced-in (Psi/Ψ) of a potentially alternative exon identified in `outrigger index`, we use the equation for $\Psi = \frac{\text{inclusion reads}}{\text{total reads}}$ [16], with substantial checks for whether the event is valid (**Supplementary Software Figure 4**). For SE, there is only one exclusion junction and thus the the exclusion junction is weighted by two to compensate (Eq. 1). For MXE, the calcluation is simply the inclusion reads divided by the total reads (Eq. 2). The junction reads between exon i and exon j are presented as $r_{i,j}$, displaying inclusion reads in red and exclusion reads in blue.

$$\text{SE } \Psi \quad \Psi = \frac{r_{1,2} + r_{2,3}}{r_{1,2} + r_{2,3} + 2r_{1,3}} \tag{1}$$

$$\text{MXE } \Psi \quad \Psi = \frac{r_{1,2} + r_{2,4}}{r_{1,2} + r_{2,4} + r_{1,3} + r_{3,4}} \tag{2}$$

Multiple validation steps were incorporated to ensure that the junction reads observed in each sample are consistent with the type of splicing event annotated by `outrigger`. This process is described in **Supplementary Software . 4**.

Case 1: Incompatible junctions with sufficient reads. This step checks whether the junction reads are compatible with a MXE event, or rather a twin cassette event. Specifically, evidence of $r_{2,3} > r_{\min}$ or $r_{1,4} > r_{\min}$ suggests this junction is a twin cassette event but not an MXE event. In such cases, $\Psi = \text{NA}$. As described in `outrigger index`, the minimum number of reads is user-defined, for example to 37 with `--min-reads 37`.

Case 2: Zero observed reads. Given no reads is observed, this event is $\Psi = \text{NA}$, rather than $\Psi = 0$ since $\Psi = 0$ indicates exclusion.

Case 3: All compatible junctions with insufficient reads. No single junction has the minimum number of reads $r_{\min}$, by default $r_{\min}$ is 10, and can be modifiable by the `--min-reads` flag. If this is the case, we assign $\Psi = \text{NA}$.

Case 4: Only one junction with sufficient reads. This applies to a single junction of two junctions per isoform, e.g. Isoform2 of either SE or MXE events, and Isoform1 of an MXE event, has sufficient reads. Since only one junction has the minimum number of reads, $r_{\min}$, no sufficient evidence indicates inclusion of exon-of-interest, thus, we assign $\Psi = \text{NA}$.

Case 5: One junction with $> 10\times$ more reads than the other. When the alternative exon is covered on the two sides with junction reads of great disparity, there is insufficient evidence supporting the inclusion of alternative exon or suggests the exon may involved in a complex splicing, rather than a SE or MXE. Thus, $\Psi = \text{NA}$. The default multiplier is 10 and can be modified by the user, for example to 55 by `--uneven-coverage-multiplier 55`.

Case 6: Exclusion: Isoform2 with suffcient reads and Isoform1 with zero reads. All junctions on Isoform2 have greater than the minimum reads $r_{\min}$, and all junctions of Isoform1 have no observed reads, thus $\Psi = 0$.

Case 7: Inclusion: Isoform2 with zero reads and Isoform1 with sufficient reads. All junctions on Isoform2 have no observed reads and all junctions of Isoform1 have greater than the minimum reads $r_{\min}$, thus $\Psi = 1$.

Case 8: Sufficient reads on all junctions. Both Isoform1 and Isoform2 have greater than the minimum reads on all their junctions. This is the best possible case for alternative splicing.

Case 9: Isoform2 with sufficient reads but Isoform1 has one or more junctions with insufficient reads. If the exclusion isoform, Isoform2 has sufficient reads, but the inclusion isoform (Isoform1) does not, then we assess whether the total read coverage of the event, $\sum_{i,j} r_{i,j}$exceeds $r_{\text{threshold}}$. If so, a Ψ is calculated; if not, $\Psi = \text{NA}$. We define $r_{\text{threshold}}$ as the number of junctions n times the minimum number of reads $r_{\min}$. For example, with a minimum read count is 10 on an SE event, $r_{\text{threshold}} = 30$. For a minimum read count of 10 on an MXE event, $r_{\text{threshold}} = 40$.

Case 10: Isoform2 has one or more junctions with insufficient reads but Isoform1 has sufficient reads. Similar to Case 9, we again test if the total read coverage is sufficient to calculate Ψ, i.e. if $\sum_{i,j} r_{i,j} \geq r_{\text{threshold}}$. If so, we calculate Ψ, and if not, we assign $\Psi = \text{NA}$.

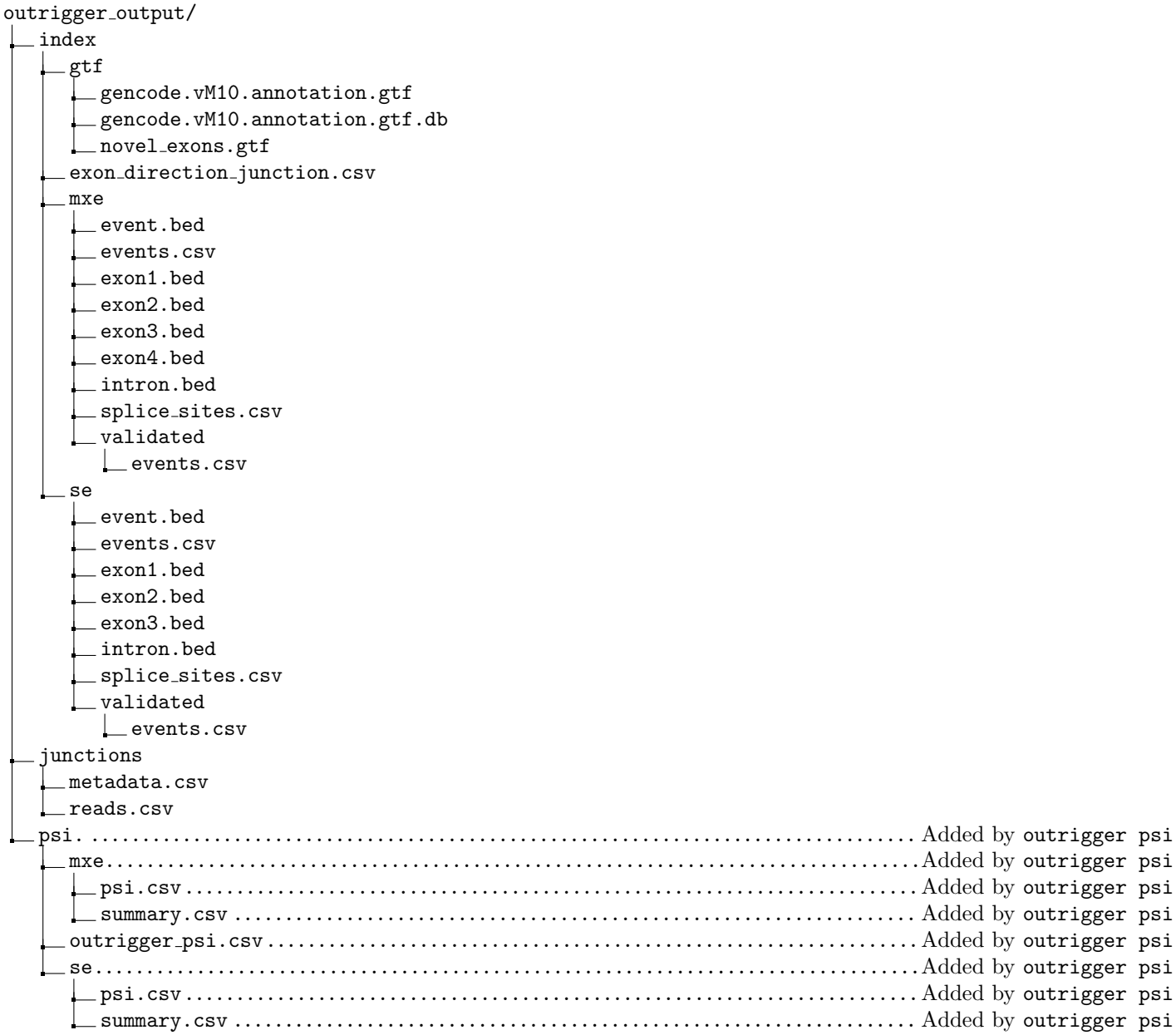
Case 11: Isoform1 and Isoform2 each have both sufficient and insufficient junctions. This case only applies to MXE events as SE events have as single Isoform2 junction, and cannot have both sufficient and insufficient junctions. If by the per-junction coverage, it is unclear whether the event has sufficient coverage, then we test if the total coverage of the event is sufficient. If so, we calculate Ψ, and if not, we assign $\Psi = \text{NA}$.

1.1.1.4.1 Outputs The output of `outrigger psi` is added into the `outrigger_output` folder by creating a `psi` folder for each splice type. `psi.csv` contains Ψ in a matrix, and the `summary.csv` produces a summary of all the events observed in all samples with their junction reads.

To follow up with our `outrigger index` and `outrigger validate` commands, we can run the below example command in the same directory:

```
outrigger psi
```

This command adds to the existing output folder `outrigger_output`. Therefore, we don't need to specify a genome location or reads or index location if this command is run from the same folder as the `outrigger index` command was run, and there exists in the directory a folder called `outrigger_output`.



Supplementary Figure 3: Example output of `outrigger psi` command.

1.1.1.5 Advantages and limitations of `outrigger`. The main advantages of `outrigger` are speed and conserved memory footprint. As `outrigger` operates only on junction reads, rather than resampling reads from a `.bam` alignment file, which can range in size from 500MB to 20GB and results in a high memory footprint, `outrigger` summarizes each `.bam` file to only its junction reads and uses that to estimate Psi/Ψ values. Additionally, employing three steps of `outrigger` `outrigger` is able to maximize the number of potential alternative events and subsequently apply strict validation rules in the step of outrigger psi calculation to eliminate false positive events from each sample. However, currently, `outrigger` can only deal with SE and MXE events. We are in the process of incorporating other alternative splice types.

1.1.2 Comparison to other methods

In comparison to the popular splicing program MISO[8], `outrigger` has three major advantages:

1. Ability to build de novo exon indexes (`outrigger index`)
2. Flexiblity of junction-based definitions of alternative exons, enumerating all possible flanking exons (`outrigger index`)
3. Ability to eliminate incompatible alternative events (`outrigger psi`)
4. Speed of evaluation. Instead of using the huge `.bam` alignment files directly, `outrigger` summarizes the files as junction reads, leading to much faster calculation of percent spliced-in. Once an index is built with `outrigger index` (24-48

hours), then calculation of Ψ/Psi takes 2-4 hours, even on hundreds of samples. With MISO, the calculation can take 8 hours per sample.

1.1.2.1 Ability to build de novo exon indexes. MISO provides pre-built alternative splicing indexes, which may not be incompatible with the data at hand. There is a program, GESS [18] to detect alternative exons from `.bam` files, which can only handle a handful files at a time and freeze when given hundreds of single-cell `.bam` files. In contrast, in the outrigger indexing step, `outrigger` builds indexes based on provided data, which will be integrated with provided exon annotation allowing identification of novel exons.

1.1.2.2 Flexiblity of junction-based definitions of alternative exons, enumerating all possible flanking exons. Multiple possible flanking exons can be associated with an alternative exon, most algorithms, including MISO and rMATS [15], choose a single set (often the shortest one), rather than being flexible and allowing the user to choose the relevant ones. The resulting "best guess" of the alternative event may not be biologically relavent and may be misleading to interprete. In such case, computational translation of alternative events, as demonstrated in Figure 4, will not be possible.

1.1.2.3 Ability to eliminate incompatible alternative events Comparing MISO Ψ values side-by-side with a corresponding `outrigger psi` calculation, we find that 46% of MISO Ψ values are rejected and assigned Ψ = NA by `outrigger` (**Supplementary Software Figure 1**).

A large group of false positives that are correctly rejected by `outrigger` are Case 1, where only incompatible junctions present sufficient reads. For example, when twin cassette events are annotated as MXE events and the data indicates inclusion of both alternative exons, MISO will calculate Ψ as 0.5. Because MISO uses a prior of Ψ = 0.5 and resamples the data to calculate Ψ. In such a case, MISO is never convinced that Ψ should be towards 1 or 0 and remains at Ψ 0.5 (**Supplementary Software Figure 1A**).

The majority of the false positives are Case 4, where only one junction has sufficient reads. As MISO counts both junctions to calculate Ψ, shown in **Supplementary Software Figure 1B-C**, many of the events are not covered on both sides of the alternative exons, which may suggest the events are not true SE events, but rather alternative first exon events, for instance.

We used MISO's event definitions and found that as many as 50% of MISO events did not pass the stringent rules of `outrigger`, primarily due to the incompatibility with the annotation of SE and MXE and insufficient coverage (**Supplementary Software Figure 1J-L**).

1.2 `anchor`: Modality estimation

1.2.1 Algorithm overview

1.2.1.1 Model modalities as beta distributions We define *modality* as a distinct type of distributions. Since Ψs are continuous value between $(0, 1)$, distribution of Ψ can be modeled as Beta distribution. The probability density function for the Beta distribution, $\Pr(\alpha, \beta)$ is defined between $(0, 1)$, with parameters $\alpha > 0$ and $\beta > 0$,

$$\Pr(\alpha, \beta) \sim \frac{1}{\mathrm{B}(\alpha, \beta)} x^{(\alpha-1)} (1-x)^{(\beta-1)}, \tag{3}$$

where $\mathrm{B}(\alpha, \beta)$ is the Beta function, defined by $\alpha > 0$ and $\beta > 0$. It may be easier to think about how the α and β parameters affect distribution by observing the mean and variance **Supplementary Software Figure 5A**. The beta distributions can be described by four parameterizations: $1 \leq \alpha < \beta$, $\alpha = \beta > 1$, $\alpha > \beta \geq 1$, $\alpha = \beta < 1$ (**Supplementary Software Figure 5B**). Conveniently, these four configurations correspond to the four modalities we are interested in: $1 \leq \alpha < \beta$ corresponds to *excluded*, $\alpha = \beta > 1$ to *middle*, $\alpha > \beta \geq 1$ to *included*, and $\alpha = \beta < 1$ to *bimodal* (**Supplementary Software Figure 5C**). The final *multimodal* modality corresponds to $\alpha = \beta = 1$, which is equivalent to the uniform distribution used as null model.

1.2.1.2 Model parameterization To describe feature distribution as modalities, we parameterized the four parameterizable modalities and used Bayesian model selection to choose the best model to describe the distribution. Python package `scipy` [11;13] was used to implement Beta distribution. For included (excluded) modality, we fixed β (α) at 1 and linearly increased α (β) from 2 to 20 (**Supplementary Software Figure 5D**). We chose 2 as a starting parameter since it is near the $\alpha = \beta = 1$ uniform distribution, as we wanted to allow excluded and included distributions with noise. For bimodal (middle) modality, we changed α and β simultaneously, monotonically decreasing (increasing) the parameters from $\alpha = \frac{1}{12}$, $\beta = \frac{1}{12}$ ($\alpha = 2, \beta = 2$) to $\alpha = \frac{1}{30}, \beta = \frac{1}{30}$ ($\alpha = 20, \beta = 20$). The parameters for bimodal start at $\frac{1}{12}$ rather than $\frac{1}{2}$ because starting the parameters from $\frac{1}{2}$ resulted in more false positive "bimodal" events, whereas starting the parameters from $\frac{1}{2}$ ensures any density near 0.5 is downweighted.

The fit of feature distribution is assessed to the four configurations using Bayes Factors, represented by K,

$$K^{(m)} = \frac{P(D|M_1^{(m)})}{P(D|M_0)} \tag{4}$$

$$= \frac{\sum_i P(\alpha_i^{(m)}, \beta_i^{(m)}|M_i^{(m)})P(D|\alpha_i^{(m)}, \beta_i^{(m)}, M_i^{(m)})}{\sum P(\alpha_0, \beta_0|M_0)P(D|\alpha_0, \beta_0, M_0)} \tag{5}$$

$$= \frac{\sum_i P(\alpha_i^{(m)}, \beta_i^{(m)}|M_i^{(m)})P(D|\alpha_i^{(m)}, \beta_i^{(m)}, M_i^{(m)})}{1} \tag{6}$$

$$= \sum_i P(\alpha_i^{(m)}, \beta_i^{(m)}|M_i^{(m)})P(D|\alpha_i^{(m)}, \beta_i^{(m)}, M_i^{(m)}) \tag{7}$$

Where $M_i^{(m)}$ is the model of interest (e.g. $M_i^{(\text{bimodal})}$) and $\alpha_i^{(m)}, \beta_i^{(m)}$ are the corresponding parameters from the parameterization shown in **Supplementary Software Figure 5D**. The null model, M_0 is the uniform distribution, where $\alpha_0 = \beta_0 = 1$, and thus $P(D|M_0) = 1$ for all datasets. We use a Bayes Factor cutoff of K_{cutoff} to indicate the threshold where the model begins to explain the data reasonably well. In practice we set $K_{\text{cutoff}} = 2^5$ ($\log_2 K_{\text{cutoff}} = 5$).

The excluded and included modalities vary only one parameter at a time, whereas middle and bimodal modalities vary both α and β simoutanously. Models with more parameters are more likely to fit, thus we fit to the one-parameter models first, assessing whether $K > K_{\text{cutoff}}$ for either excluded or included. No distribution can fit both excluded and included modalities, thus it is assigned to the modality with highest K. Next, the distribution is fitted to the two-parameter bimodal and middle models, checking if $K > K_{\text{cutoff}}$. If neither modality applies, we assign the modality to *multimodal* (**Figure 2C**).

As exact 0 and 1 are not in the range of the Beta distribution, we implement this model selection by adding a small number (0.001) to 0 and subtracting this small number from 1. Thus, we approximate the data-derived distribution from the invalid closed interval [0, 1] to the valid open interval of (0, 1).

1.2.2 Simulations

We optimized the algorithm parameters using test datasets and visually inspecting random samples from both the best- and worst-fitting data and ensuring that the even the worst fitting data was still believably categorized as the modality (**Supplementary Software Figure 6**).

1.2.2.1 Dataset 1: "Perfect Modalities" with noise To test the limits of `anchor`, we simulated perfectly excluded, middle, included, and bimodal distribution, added uniform random noise with 100 iterations, and estimated modality at each noise level with iteration (**Supplementary Figure 3A**). As expected, the most frequently predicted modality was "multimodal," since the dataset was created from randomly added noise (**Supplementary Figure 3B**). The next frequent modality was bimodal, followed by a tie with excluded and included, and the least frequent one is middle modality. We found that these parameterizations can accurately predict modality with up to 35% noise added to the middle modality, 50% noise added to excluded and included modalities, and up to 70% noise added to the bimodal modality(**Supplementary Figure 3D**). By visual inspection of distributions fit best or worst to each modality (**Supplementary Software Figure 6A**), we observed that the bimodal distributions are sufficiently different from other parameterizations, demonstrating the robustness of the algorithm.

1.2.2.2 Dataset 2: "Maybe Bimodals" with noise To test the proportions of zeros and ones that able to constitute "bimodal" distribution, we created another dataset comprised 100 samples of varying amounts of 0s and 1s, and adding random uniform noise (**Supplementary Figure 3H**). The primary predicted modality was bimodal, then multimodal, and finally included and excluded (**Supplementary Figure 3I**). No distribution was predicted as the middle modality, indicating the bimodal and middle modalities are drastically different with little chance of mis-assignment. The falloff of correctly predicting bimodality is at adding 70% noise (**Supplementary Figure 3K**), consistent with the previous simulation with "Perfect Modalities" dataset (**Supplementary Figure 3D**). We found that bimodality is determing with a 90:10 (10:90) proportion of samples of 0:1 (0:1) (**Supplementary Figure 3L**). Visual inspection of distributions fit best or worst to each modality confirmed the assignment of each modality(**Supplementary Software Figure 6B**).

To summarize, simulation with two different datasets indicates that 1) bimodal modality can tolerate to up to 70% of uniform random noise, and middle modality is least tolerable to noise at only 30%, 2) included and excluded modalities are drastically different, so as the middle and bimodal modalities, thus the two step modality assignment procedure (**Figure 2**) is well-grounded, 3) `anchor`is able to determine a bimodal modality with up to 90:10 proportion of zeros and ones.

1.2.3 Comparison to other methods

1.2.3.1 Simple binning We can compare this to other methods we attempted, such as fixing bins of $[0, 0.3, 0.7, 1]$ and using cutoffs for the densities, which does not account for the continuous nature of the underlying distributions. We found the

modality whose binned distribution was the smallest distance (measured by Jensen-Shannon Divergence[2]) away from each binned event. In both the simulated modalities and simulated bimodal datasets, we found a sharp increase in multimodal distributions and by eye, poorer categorization of the bimodal modality, especially at the decision boundary of low JSD (**Supplementary Figure 3C, E, J, L, P**).

1.2.3.2 Bimodality index Another test for bimodality is the Bimodality Index[17] (BI), which requires estimating each feature as a mixture of Gaussian models. We used the implementation of Generalized Mixture Models in `scikit-learn`[14] to estimate two Gaussian distributions for each model, and calculated the BI. For perfect bimodal featues, the value is large, for example, we found that for the zero-noise bimodal event, the BI = 402) and was the single bimodality index that was larger than 100 for any feature (**Supplementary Figure 3F, L, P**). This shows that our method is more sensitive to finding bimodal features with the addition of noise, which BI cannot handle.

1.2.3.3 Hartigan's Dip test A commonly used test for unimodality is Hartigan's dip test[7]. If the distribution fails the unimodality test, then it is considered bimodal. To define a cutoff for when the dip statistic becomes reliable, we calculated the dip statistic using a Python implementation of the test, called `diptest`[12]. We used a p-value cutoff of $p < 0.05$ as our threshold for assigning an event as bimodal. We used the diptest statistic on the two datasets, and found that while the zero-noise bimodal event was not detected as bimodal, adding as small amount of noise *improved* the diptest's detection of bimodal events (**Supplementary Figure 3G, M, Q**), and the accuracy dropped off at a very high noise level - 90%. As expected, the excluded, included, and middle modalities weren't detected as bimodal, except at higher noise levels, which we also saw with `anchor`.

1.3 `bonvoyage`: Transformation of distributions to *waypoints* and *voyages*

1.3.1 Algorithm overview

The goal of `bonvoyage` is to be able to summarize the entire distribution of a feature into a single point in space, enabling visualization multiple distributions at a time with intuitive interpretation. To accomplish this, we will transform one-dimensional vectors into two-dimensional space. Specifically, the x-axis will represent the *excluded* dimension and the y-axis will represent the *included* dimension, and all points will be described as a sum of excluded and included components (**Figure 6A**, left). For example, for two distinct cell-types, we can imagine a feature that starts at a included modality in the first and changes to a excluded event in the second, or changes from middle to bimodal (**Figure 6A**, right).

1.3.1.1 Data discretization We will use a reduced representation of our splicing data by binning each feature on bins b of size 0.1, where b_n represents the nth bin. We represent the binned splicing matrix with B_Ψ, where $B_\Psi[k, j]$ represents the fraction of non-null samples in feature j with Ψ value contained in b_k. In practice, we pre-filter the data by using only features for which there are enough samples. In the main text for this paper, we used a minimum of 10 cells.

1.3.1.2 Dimensionality reduction via non-negative matrix factorization Non-negative matrix factorization (NMF) is a parts-based dimensionality reduction algorithm which results in meaningful, interpretable results[9]. It is an alternative to other dimensionality reduction methods such as principal- and independent- component analyses (PCA and ICA) because its features are both independent, and non-negative, and thus each feature is composed of a sum of the underlying structure of the data, without pesky negative terms.

Thus, for NMF, we will be reducing B_Ψ as such,

$$B_\Psi \approx W \times H, \tag{8}$$

Where W is a (features, 2)-size matrix of the composition of each feature as a sum of how many samples are excluded and included. We found that in the alternative splicing data, the primary components were the included and excluded values, but in other datasets, this may not be the case. Thus, as the components of NMF are the most prominent features, to ensure reproducibility of the axes across datasets, we seeded the NMF transformation with a matrix that is composed of features that are primarily excluded plus a single included feature (**Supplementary Figure 6A**). We used the Python package `scikit-learn`[14] for the Projected Gradient NMF implementation.

We call the projected distributions "waypoint space," and the distance between two points a "voyage," such as the voyage of the MXE event in PKM (**Figure 6C**).

1.3.2 Simulations

1.3.2.1 Transformation of static distributions To demonstrate the ability of `bonvoyage`, we created a simulated dataset which we call "Maybe Everything" consisting of every combination of 0s, 1s, and 0.5s (**Supplementary Figure 6A-D**), essentially incorporating both the "Perfect Modalities" (from 1.2.2.1) and "Maybe Bimodals" (from 1.2.2.2) into a single

dataset. Again, we added uniform random noise at 5% intervals. We transformed the entire simulated dataset into the *"waypoint"* space.

To identifying features which change in distribution, we calculate the *"voyage"* between them in waypoint space. As a demonstration, we shuffle the simulated data to create two different *in silico* phenotypes. We will use each feature as a *"waypoint"* along the voyage, and calculate total travel distance of each feature between the phenotypes.

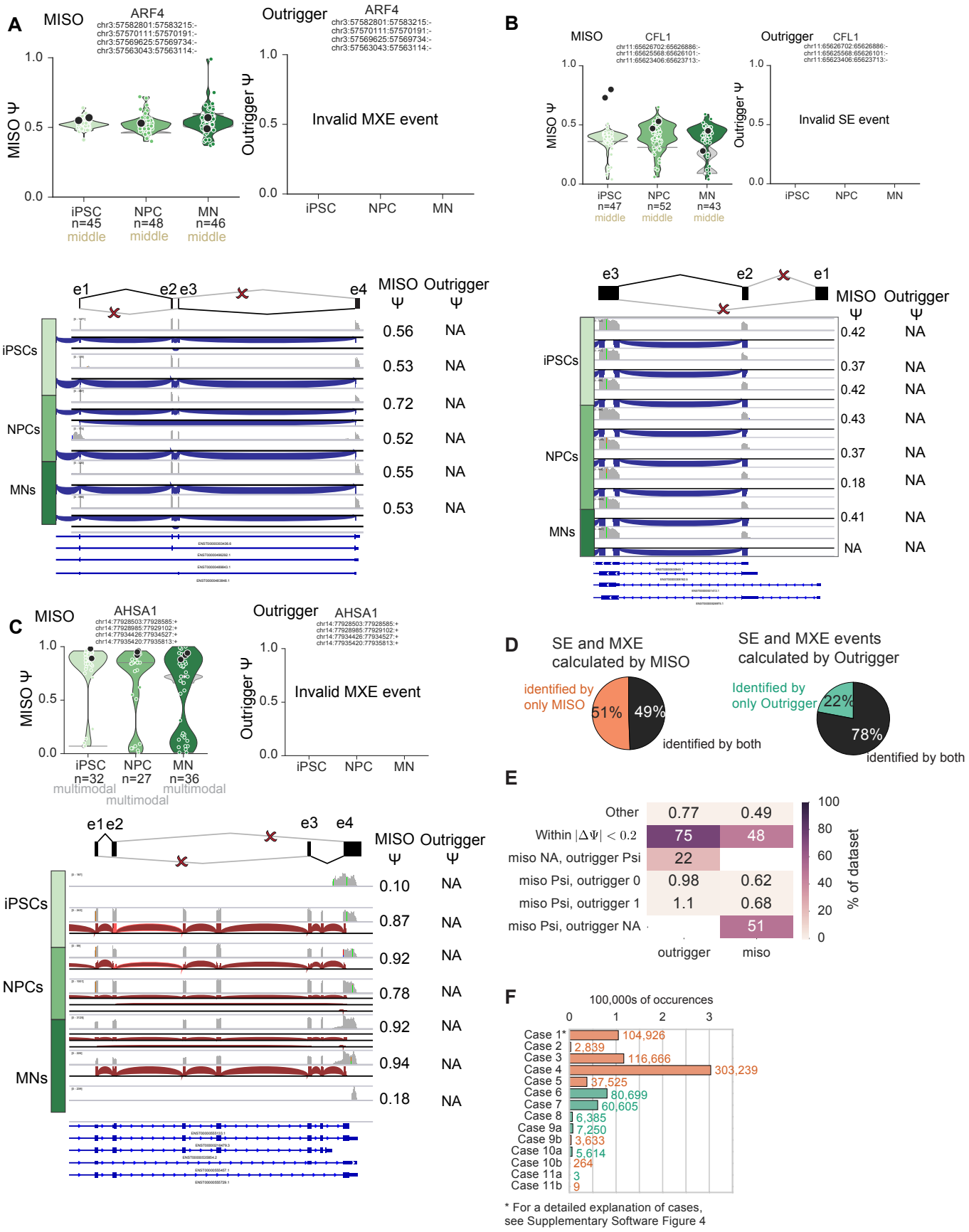
A key aspect of the waypoint space is that while changes from exclusion to inclusion are easy to spot by a change in means, the change from a middle to a bimodal is not, and requires a battery of other tests to find. Here, voyage space has a significant advantage as it gives both the magnitude of change and a directly interpretable direction.

1.3.3 Comparison to other methods

As there exist many methods for comparing distributions, we will show that the magnitude of change obtained from `bonvoyage` is comparable to other metrics for assessing changes in distribution. In particular, we will show the metrics within each modality, and across modalities, compared to Jensen-Shannon Divergence[2] (JSD) in (**Supplementary Figure 6E**). While JSD is more sensitive to slight changes in distribution (their scatterplots are skewed towards the right), it does not also encode directionality of change. Thus, `bonvoyage` offers a unique perspective on how to interpret changes in distribution.

2 Supplementary Software Figures

2.1 Supplementary Software Figure 1



Supplementary Software Figure 1: Examples of inconsistencies in MISO's estimation with single-cell data. A-C. Representative examples of SE and MXE AS events measured by MISO, but were unsupported with visual inspection on IGV browser, and were disqualified by `outrigger`. To identify SE and MXE events, `outrigger` constructs a *de novo* splicing index based on the junction reads in all libraries in the dataset (see details in **Supplementary Software Figures 2 to 4**). The following examples are not considered bys `outrigger`as true SE or MXE events, therefore annotated as NA. Note, MISO does not estimate modality for each event, `anchor` (see details in **Supplementary Figure 3**) was used to estimate modality.

A. Top, a MISO-annotated MXE event in ARF4 with MISO estimated Ψs ~ 0.5 and classified as "middle" modality in each of iPSC, NPC, and MN by `anchor`. Yet, in the IGV browser (bottom), this event appears as a twin cassette event, where both exons 2 and 3 are included, indicating that at least in our dataset this event is not consistent with the MISO annotation. Outrigger disqulifies this event as a MXE and assign NA (top left).

B. Top, a MISO-annotated SE event in CLF1 with MISO estimated Ψs ranging from 0.1 to 0.6 and is classified as a "middle" modality event by `anchor` in each of iPSC, NPC, and MN. Yet, in the IGV browser (bottom), exon 1 for this annotation is not covered at all. Given the data, outrigger do not consider this as a bona fide SE event and assign NA to this event.

C. Top, a MISO-annotated MXE event in AHSA1 with a wide range of MISO calculated Ψs and is classified as the "multi-modal" modality in each of iPSCs, NPC, and MN populations by `anchor`. Bottom, in the IGV browser. Exons 2 and 3 are the annotated alternative exons for MXE, however, another two well-covered exons between exon 2 and 3 were observed and one extra exon between exon 3 and 4, which disqualify this event as an MXE event. Furthermore, when both exon 2 and 3 are included, MISO estimated Ψ scores are closer to 1 instead of around 0.5, as was seen in (**A.**). Thus, outrigger rejects this as MXE and assign NA.

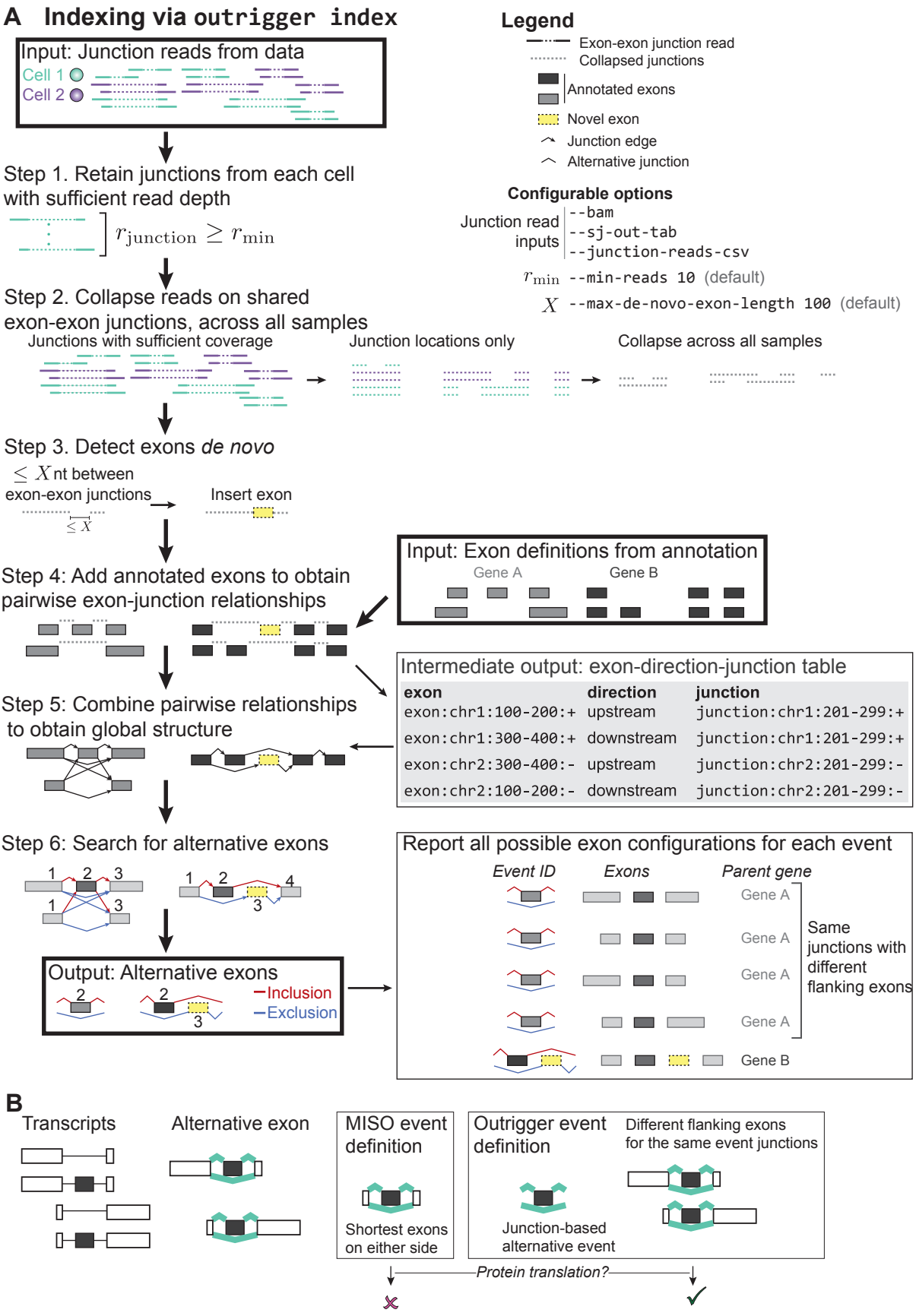
D. Using `outrigger`'s strict rules on MISO annotations, the majority (51%) of the data generated by MISO was rejected by `outrigger` (left). Right, using the exact same annotation from MISO, `outrigger` 22% of events found by `outrigger` had too wide of a confidence interval (> 0.4) by MISO.

E. Heatmap comparing the numbers and percentages of alternative events that were within $|\Delta\Psi| < 0.2$, switched to exactly 1 or 0 in `outrigger`, were NA in either MISO or `outrigger`, or were in another case.

F. Barplot of the number of cases found only in MISO (orange) and rejected as NA by `outrigger`, and of the cases found only by `outrigger`(green) and considered to have too wide of a confidence interval by MISO.

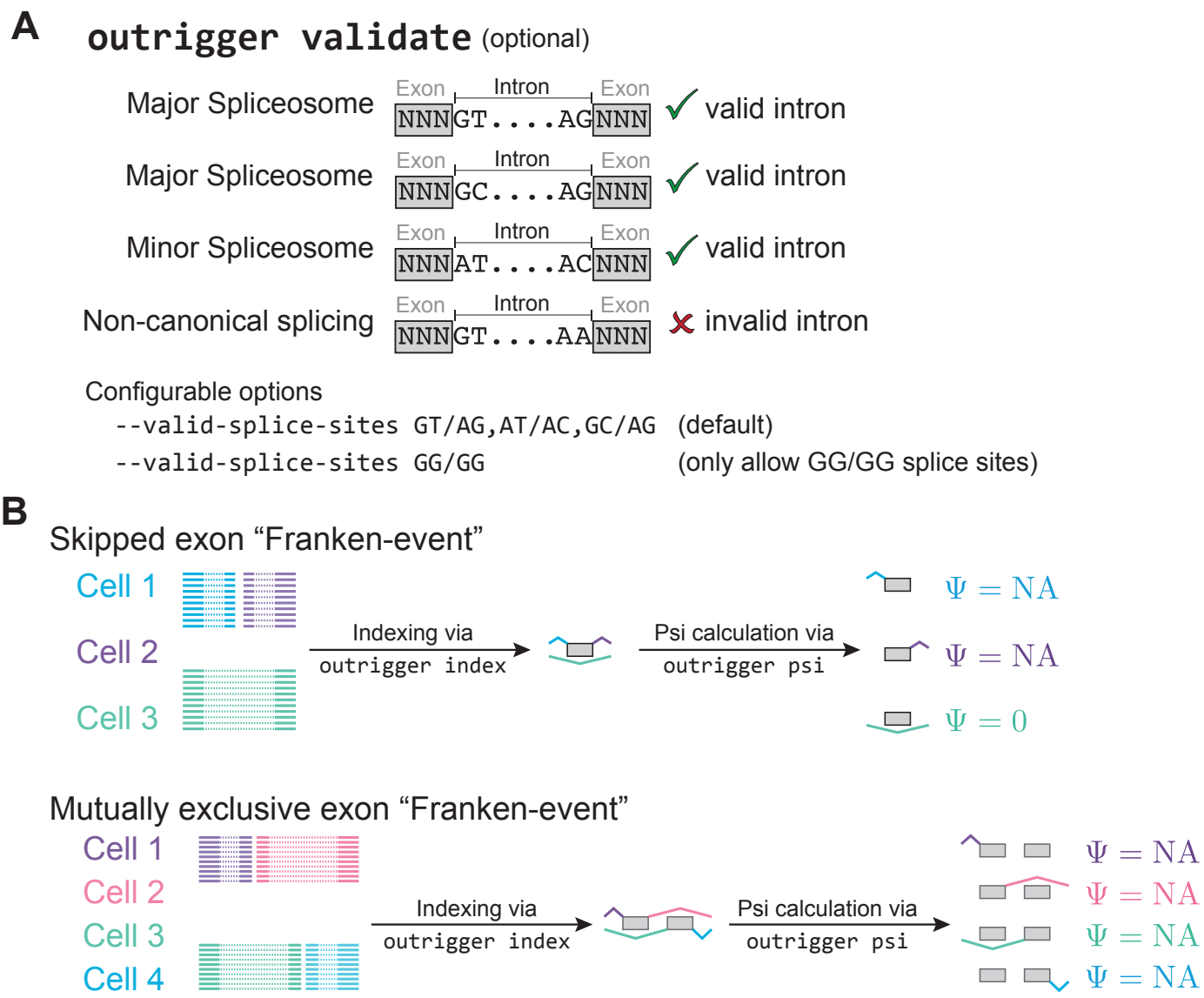
To summarize, `outrigger` follows strict rules to identify alternative splicing (**Supplementary Software Figures 2 to 4**) and provides a Ψ distribution more localized at the extremes of $\Psi = 0$ and $\Psi = 1$. Although `outrigger` may identify fewer events, they are true SE and MXE events.

2.2 Supplementary Software Figure 2



Supplementary Software Figure 2: Internal steps of indexing via `outrigger index`: Exons identification and defining alternative events.
A. Internal workings of the indexing step via `outrigger index`. User-provided inputs junction reads can be either genome-aligned `.bam` files, the `.SJ.out.tab` splice junction files from the STAR aligner, or a compiled table in `.csv` of all junction reads from all samples for the project. Step 1, only junction reads with sufficient depth in a cell/sample are retained. By default, the minimum number of reads is 10 per cell/sample, which can be modified with the flag `--min-reads`. Step 2, junction reads are used to identify junction locations, and reads are aggregated across all cells/samples regardless of which cell/sample it came from. Step 3, if there is a "gap" between two junctions that is smaller than certain length X (by default, $X = 100$ nucleotides but can be modified with the flag `--max-de-novo-exon-length`), then an exon is inserted. Step 4, the identfied exons are compared with the annotated exons to obtain the pairwise relationships between exons and junctions. Step 4 outputs a table of "triples:" of `(exon, direction, junction)` encoding the directional relationship between exons and junctions. Step 5, the output tables from step 4 are utilized to connect exons through junctions and creates a graph database. Finally, in Step 6, alternative exons are identified by traversing the graph database. The output of the indexing step run by the command `outrigger index`, is junction-based, outputting the alternative exon and all possible configurations of flanking exons for each event. For example, on the bottom right, the same skipped exon event using the same alternative junctions, have four possible configurations of flanking exons. They are considered to be the same event, but are reported with all four configurations for the ease-to-use in downstream analysis.
B. Defining alternative events and comparison of biological interpretability of events found by MISO and `outrigger`. For a given alternative exon (black box), there can be multiple transcripts corresponding to the alternative exon but with different flanking exons. MISO chooses to define the alternative event using the shortest exons on both sides. Yet, this MISO-defined alternative event may not actually exist as a transcript in the dataset and will be misleading to interpret. For example, attempts to translate such non-existing transcript(s) will be inappropriate. In contrast, `outrigger` defines the event based on the junctions, and outputs all corresponding flanking exon configurations, thus enabling boader use of the outputs and more relevant biological interpretation.

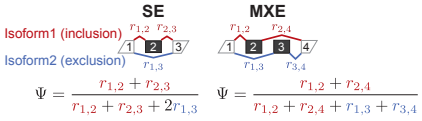
2.3 Supplementary Software Figure 3



Supplementary Software Figure 3: `outrigger` **validation and pathological cases.**
A. Validation via `outrigger validate`: Removal of alternative events with introns lacking consensus splice sites. In this optional step, exons with flanking introns lacking known splice site motifs are removed. This is configurable. By default, the valid splice sites are specified as, `--valid-splice-sites GT/AG,GC/AG,AT/AC`, but can be any pair of two nucleotides. **B.** Possible pathological cases of `outrigger`. These "Franken-events" consist of junctions that were observed in independent samples. At the indexing step, aggregated reads from multiple cells/samples are considered to construct an index of all junctions to maximize the number of AS events. Yet, at the Psi/Ψ calculation step, in each individual cell/sample, insufficient reads may be observed for certain junction resulting in Ψ = NA in some cells/samples for the same event. Top, skipped exons, if each junction is observed only in one cell, the cell with the exclusion junction is assigned a $\Psi = 0$ while the remaining cells are assigned as Ψ = NA. Bottom, mutually exclusive exons, Ψ = NA for all 4 cells, as there is insufficient evidence of exon inclusion or exclusion in any one cell. Thus, the number of detected events output by `outrigger index` can greatly overestimate the number of valid events in the dataset found by `outrigger psi`.

2.4 Supplementary Software Figure 4

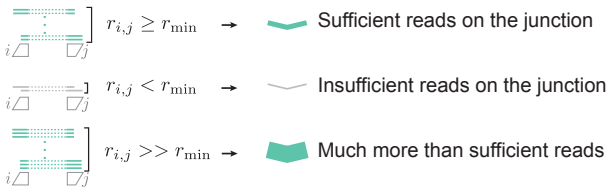
Psi (Percent spliced-in) calculation via **`outrigger psi`**



$$\Psi = \frac{\text{inclusion reads}}{\text{inclusion + exclusion reads}}$$

		Notes	Compatible w/ annotation?	
Case 1	Not applicable	Incompatible junctions with sufficient reads	✗	$\Psi = \text{NA}^*$
Case 2		Zero observed reads	✗	$\Psi = \text{NA}$
Case 3		All compatible junctions with insufficient reads	✗	$\Psi = \text{NA}$
Case 4		Only one junction with sufficient reads	✗	$\Psi = \text{NA}$
Case 5		One junction with >10x reads than the other**	✗	$\Psi = \text{NA}$
Case 6		Exclusion: Isoform2 with sufficient reads and Isoform1 with zero reads	✓	$\Psi = 0$
Case 7		Inclusion: Isoform1 with zero reads and Isoform2 with sufficient reads	✓	$\Psi = 1$
Case 8		Sufficient reads on all junctions	✓	$0 < \Psi < 1$
Case 9		Isoform2 with sufficient reads but Isoform1 has one or more junctions with insufficient reads	? a. Total reads ≥ $r^{***}_{\text{threshold}}$ b. Total reads < $r_{\text{threshold}}$	$0 < \Psi < 1$ / $\Psi = \text{NA}$
Case 10		Isoform1 with sufficient reads but Isoform2 has one or more junctions with insufficient reads	? a. Total reads ≥ $r_{\text{threshold}}$ b. Total reads < $r_{\text{threshold}}$	$0 < \Psi < 1$ / $\Psi = \text{NA}$
Case 11	Not applicable	Isoform1 and Isoform2 each have both sufficient and insufficient junctions	? a. Total reads ≥ $r_{\text{threshold}}$ b. Total reads < $r_{\text{threshold}}$	$0 < \Psi < 1$ / $\Psi = \text{NA}$

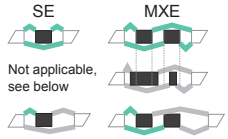
Legend



$r_{i,j}$ Reads on junction spanning exon *i* to exon *j*

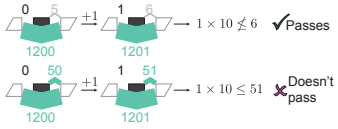
r_{min} Minimum number of reads per junction, default 10 and can be user-defined with the flag `--min-reads`

* $\Psi = \text{NA}$ can mean three things:
1. Transcript was not expressed
2. Insufficient evidence to confidently call exon inclusion or exclusion
3. Junctions map to different alternative or flanking exon(s) – considered as distinct events during the indexing step, `outrigger index`



For a SE event, if the junctions map to different alternative exon (small black exon on the top), then the event with smaller exon has a Ψ value ranging from zero to one, but for the wider exon (on the bottom), which doesn't have matched inclusion reads, this event is called excluded with Ψ=0

** The multiplier for how much greater one side junction can be is user-defined with the flag `--uneven-coverage-multiplier`, here shown with the default value of 10.
To deal with 0 reads, a pseudocount of 1 is added to all junctions for this test only:



*** $r_{\text{threshold}}$ Threshold for total junction reads in the event

$$r_{\text{threshold}} = n_{\text{junctions}} \times r_{\text{min}}$$

e.g. for an MXE event (4 junctions) and a minimum of 10 reads per junction: $\sum_{i,j} r_{i,j} = 4 \times 10 = 40$

$\sum_{i,j} r_{i,j}$ Total Junction reads

$n_{\text{junctions}}$ Number of junctions in splicing event type (e.g. 3 for SE or 4 for MXE)

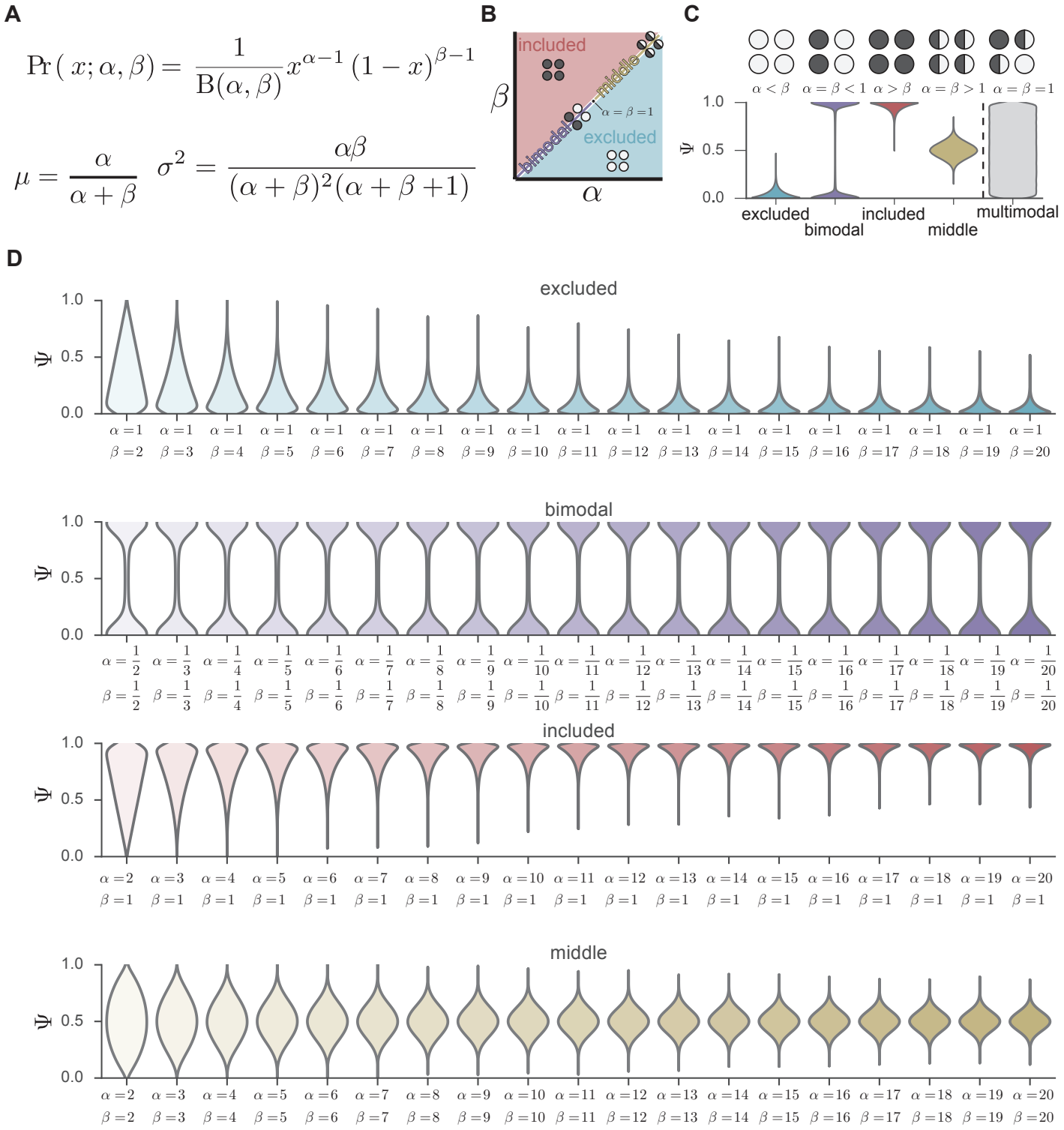
Configurable options

Junction read inputs	`--bam` `--sj-out-tab` `--junction-reads-csv`	(default: reads from `outrigger index`)
r_{min}	`--min-reads 10` (default)	
	`--uneven-coverage-multiplier 10` (default)	

Supplementary Software Figure 4: Cases created by percent spliced-in calculation via the command `outrigger psi`. The table describes the 11-step sequential logic of `outrigger` to reject an event in a cell/sample based on that cell/sample's junction reads. If an event reaches a $\Psi = \text{NA}$ case, then it is rejected from that sample, otherwise, it continues through the cases. If the event is rejected, then it is assigned $\Psi = \text{NA}$, if it is not rejected, then it gets a $0 \leq \Psi \leq 1$ value based on the junction reads.

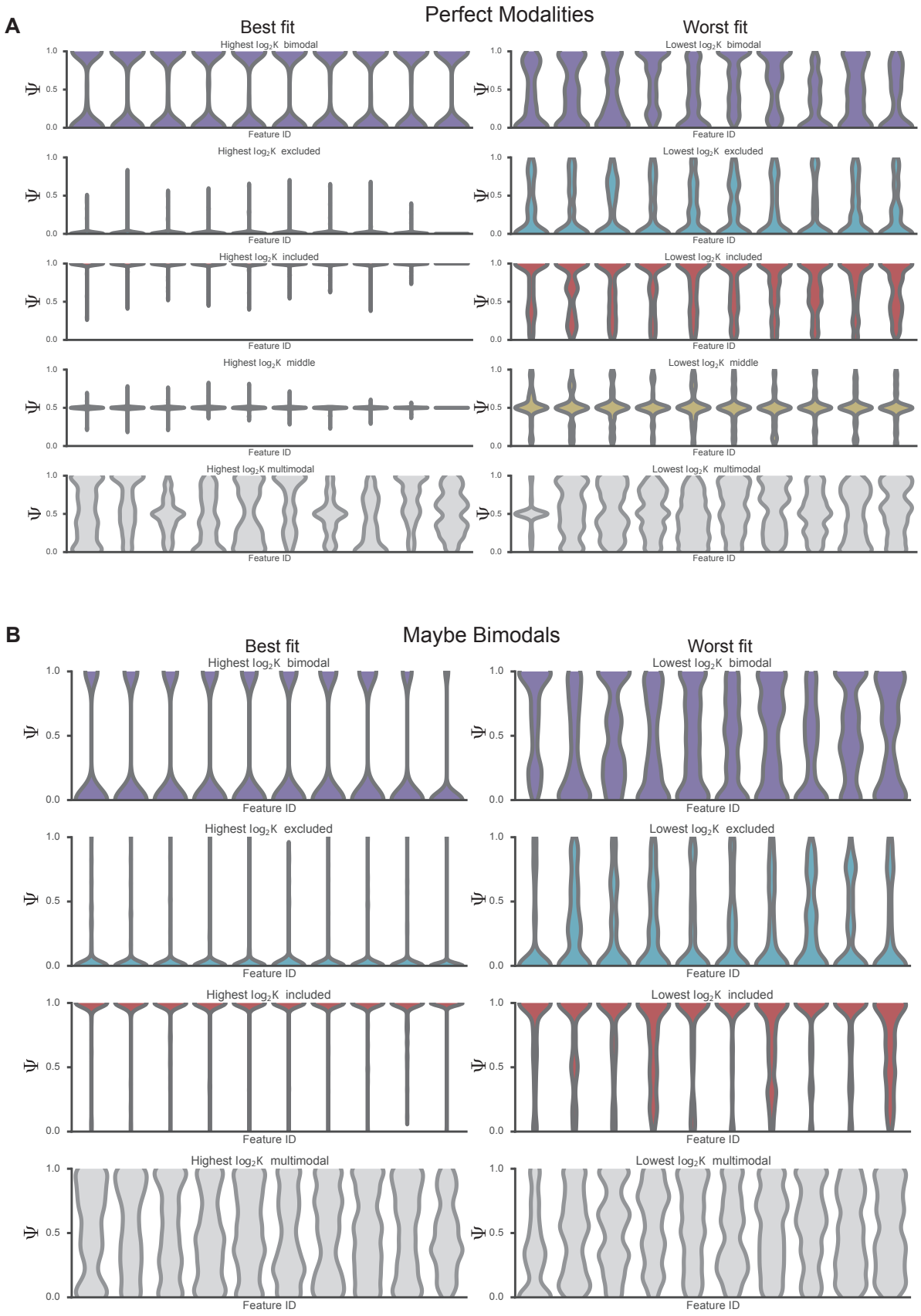
Strict evalution of percent spliced-in (Psi/Ψ). To compute the percent spliced-in (Psi/Ψ) of skipped exon (SE) and mutually exclusive exons (MXE) alternative events during the execution of the command `outrigger psi`, we use $\Psi = \frac{\text{inclusion reads}}{\text{total reads}}$. We represent the number of reads spanning the junction between exon_i and exon_j as $r_{i,j}$.

2.5 Supplementary Software Figure 5



Supplementary Software Figure 5: Overview of `anchor` parameterization of the Beta distribution.
A. Top, equation for the Beta distribution of the random variable x with parameters $\alpha, \beta > 0$. Bottom left, equation for the mean (μ) of the Beta distribution as a function of its parameters. Bottom right, equation for the variance (σ^2) of the Beta distribution as a function of parameters.
B. Cartoon of valid values of α and β parameters of Beta distribution, showing how the space is partitioned by the modalities.
C. Violinplots representing the four ideal modalities, plus the null "multimodal" distribution. Each modality is annotated with examples of four cells representing within-cell distributions of included (dark grey) and excluded (light grey) transcripts, and the corresponding parameters of the Beta distribution.
D. Violinplots of 1 million random samples of the family of Beta distributions specified by the α and β (x tick labels) parameterization of the four modalities: excluded, bimodal, included, and middle.

2.6 Supplementary Software Figure 6



Supplementary Software Figure 6: Best and worst fitting modality data using `anchor`.
Left, 10 events with largest Bayes Factor, K (best fit) from the assigned modality. Right, 10 events with smallest Bayes Factor, K (worst fit) from their assigned modality. For multimodal, as there is no fit, this simply shows 20 random events.
A. Bayesian `anchor` method on "Perfect modalities" dataset.
B. Bayesian `anchor` method on "Maybe bimodals" dataset.

References

[1] eugene-eeo/graphlite. URL https://github.com/eugene-eeo/graphlite.

[2] Thomas M Cover and Joy A Thomas. *Elements of information theory*. Wiley-Interscience, January 1991. doi: 10.1234/12345678. URL http://dl.acm.org/citation.cfm?id=129837&coll=DL&dl=GUIDE&CFID=555373975&CFTOKEN=99027636.

[3] Ryan K Dale. daler/gffutils. URL https://github.com/daler/gffutils.

[4] Alexander Dobin, Carrie A Davis, Felix Schlesinger, Jorg Drenkow, Chris Zaleski, Sonali Jha, Philippe Batut, Mark Chaisson, and Thomas R Gingeras. STAR: ultrafast universal RNA-seq aligner. *Bioinformatics*, 29(1):15–21, January 2013. doi: 10.1093/bioinformatics/bts635. URL http://eutils.ncbi.nlm.nih.gov/entrez/eutils/elink.fcgi?dbfrom=pubmed&id=23104886&retmode=ref&cmd=prlinks.

[5] Mariano A Garcia-Blanco, Andrew P Baraniak, and Erika L Lasda. Alternative splicing in disease and therapy. *Nature Biotechnology*, 22(5), May 2004. doi: 10.1038/nbt964. URL http://www.nature.com/doifinder/10.1038/nbt964.

[6] SAM BAM Format Specification Working Group. Sequence alignment/map format specification, 2014. URL http://scholar.google.com/scholar?q=related:vDsyBhxBicMJ:scholar.google.com/&hl=en&num=20&as_sdt=0,5.

[7] J A Hartigan and P M Hartigan. The dip test of unimodality. *The Annals of Statistics*, 1985. doi: 10.2307/2241144. URL http://www.jstor.org/stable/2241144.

[8] Yarden Katz, Eric T Wang, Edoardo M Airoldi, and Christopher B Burge. Analysis and design of RNA sequencing experiments for identifying isoform regulation. *Nature Methods*, 7(12):1009–1015, November 2010. doi: 10.1038/nmeth.1528. URL http://www.nature.com/doifinder/10.1038/nmeth.1528.

[9] D D Lee and H S Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, October 1999. doi: 10.1038/44565. URL http://eutils.ncbi.nlm.nih.gov/entrez/eutils/elink.fcgi?dbfrom=pubmed&id=10548103&retmode=ref&cmd=prlinks.

[10] C Joel McManus and Brenton R Graveley. RNA structure and the mechanisms of alternative splicing. *Current Opinion in Genetics & Development*, 21(4):373–379, August 2011. doi: 10.1016/j.gde.2011.04.001. URL http://dx.doi.org/10.1016/j.gde.2011.04.001.

[11] K J Millman and M Aivazis. Python for Scientists and Engineers. *Computing in Science & Engineering*, 13(2):9–12, 2011. doi: 10.1109/MCSE.2011.36. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5725235.

[12] Alistair Muldal. alimuldal/diptest. URL https://github.com/alimuldal/diptest.

[13] T E Oliphant. Python for Scientific Computing. *Computing in Science & Engineering*, 9(3):10–20, 2007. doi: 10.1109/MCSE.2007.58. URL http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4160250.

[14] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. Scikit-learn: Machine Learning in Python. *The Journal of Machine Learning Research*, 12, February 2011. URL http://portal.acm.org/citation.cfm?id=1953048.2078195&coll=DL&dl=ACM&CFID=422733224&CFTOKEN=93183407.

[15] Shihao Shen, Juw Won Park, Zhi-xiang Lu, Lan Lin, Michael D Henry, Ying Nian Wu, Qing Zhou, and Yi Xing. rMATS: Robust and flexible detection of differential alternative splicing from replicate RNA-Seq data. *Proceedings of the National Academy of Sciences of the United States of America*, 111(51):E5593–E5601, December 2014. doi: 10.1073/pnas.1419161111. URL http://www.pnas.org/lookup/doi/10.1073/pnas.1419161111.

[16] Eric T Wang, Rickard Sandberg, Shujun Luo, Irina Khrebtukova, Lu Zhang, Christine Mayr, Stephen F Kingsmore, Gary P Schroth, and Christopher B Burge. Alternative isoform regulation in human tissue transcriptomes. *Nature*, 456(7221):470–476, November 2008. doi: 10.1038/nature07509. URL http://www.nature.com/nature/journal/v456/n7221/abs/nature07509.html.

[17] Jing Wang, Sijin Wen, W Fraser Symmans, Lajos Pusztai, and Kevin R Coombes. The bimodality index: a criterion for discovering and ranking bimodal signatures from cancer gene expression profiling data. *Cancer informatics*, 7:199–216, 2009. URL http://eutils.ncbi.nlm.nih.gov/entrez/eutils/elink.fcgi?dbfrom=pubmed&id=19718451&retmode=ref&cmd=prlinks.

[18] Z Ye, Z Chen, X Lan, S Hara, B Sunkel, T H M Huang, L Elnitski, Q Wang, and V X Jin. Computational analysis reveals a correlation of exon-skipping events with splicing, transcription and epigenetic factors. *Nucleic Acids Research*, 42(5):2856–2869, March 2014. doi: 10.1093/nar/gkt1338. URL `https://academic.oup.com/nar/article-lookup/doi/10.1093/nar/gkt1338`.