Baltica Documentation

Table of Content

(https://github.com/dieterich-lab/Baltica/tree/master/docs/index.md)

Baltica: integrated splice junction usage analysis *(https://github.com/dieterich-lab/Baltica)*¶

Features¶

```
- Snakemake workflows for DJU: `junctionseq`, `majiq`, `rmats`, and
- Snakemake workflow for de novo transcriptome annotation with `str
- Process, integrate and annotate the results from the methods
- Summarise AS class of differently spliced junctions
- DJU method benchmarks
- Report on the integrative analysis
```

To get started, use the menu on the left-hand side or search function to navigate over this documentation.

Citation¶

Thiago Britto-Borges, Volker Boehm, Niels H. Gehring and Christoph Dieterich (2020) **Baltica: integrated splice junction usage analysis**. Manuscript in preparation.

Baltica is based on the work of many scientists and developers. Thus, if you use the results of their tools in your analysis, consider citing their work.

License¶

Baltica is free, open-source software released under an MIT License *(https://github.com/dieterich-lab/Baltica/blob/master/LICENSE)*.

Contact¶

Please get in touch with us the GitHub issue tracker *(https://github.com/dieterich-lab/Baltica/issues)*.

(https://github.com/dieterich-lab/Baltica/tree/master/docs/intro.md)

Introduction¶

For motivation and review of state of the art, please check:

Thiago Britto-Borges, Volker Boehm, Niels H. Gehring and Christoph Dieterich (2020) Baltica: integrated splice junction usage analysis. Manuscript in preparation.

Tips on RNA-Seq aiming differential splicing detection¶

If you aim to resolve mRNA isoforms with relatively low abundance, you should design the RNA-seq experiment accordingly. The expert suggestion *(https://support.illumina.com/bulletins/2017/04/considerations-for-rna-seq-read-length-and-coverage-.html)* is to sequence around 40 to 60 million reads pairs. This parameter is particularly relevant for complex RNA libraries, but it can be insufficient to saturate novel SJ, in our experience. Read length and paired-end reads are also critical for SJ identification, and longer reads offer more coverage of the exons boundaries. Thus, the target read length should be around 100 nucleotides for Illumina RNA-seq to maximize the read overhang length and, consequently, maximize the quality of the alignments.

In the Baltica manuscript, we propose an approach to integrate DJU results from Illumina to DJU results from third-generation sequencing.

Also, databases such as the CHESS *(http://ccb.jhu.edu/chess/)* can provide additional evidence for splice sites absent in the annotation.

(https://github.com/dieterich-lab/Baltica/tree/master/docs/setup.md)

Getting started¶

Quick example:¶

If Baltica dependencies, baltica configuration and cluster configuration are available, use:

```
baltica <workflow> <config> --use-singularity --profile <cluster>
```

- workflow:
  - all: run end-to-end wokflows
  - qc: run quality control
  - stringtie: run *de novo* and guided transcriptome assembly
  - rmats
  - junctionseq
  - majiq
  - leafcutter
  - analysis: run scripts for integration, annotation and reporting
- config: project configuration file
- cluster: Snakemake cluster profile (https://snakemake.readthedocs.io/en/stable/executing/cli.html#profiles)

Or, read below.

> **Warning**
>
> Snakemake is under active development. Please contact us (https://github.com/dieterich-lab/Baltica/issues) if you have any issues with this documentation.

Software environment:¶

Baltica framework is based on:
- A python command-line interface
- Snakemake (https://snakemake.readthedocs.io/en/stable/) workflows
- Docker containers used with Singularity (https://sylabs.io/singularity/)
- R scripts for processing, integrating, annotating, assigning biological features, and reporting
- a Rmarkdown report

We have developed it on the following computer environments:

- Linux version 4.19.0-16-amd64 Debian 4.19.181-1 (2021-03-19)

- gcc version 8.3.0
- Python version 3.7.7
- Singularity version 3.7.3
- Snakemake version 6.4.1
- Git version 2.20.1

These versions should not matter because the workflows are run within Docker containers, as long **Snakemake version > 6** and a **recent Singularity version**.

Baltica depends on python3, Singulary, and Snakemake. - How to install Singularity *(https://sylabs.io/guides/3.0/user-guide/installation.html)*
- How to install Snakemake *(https://snakemake.readthedocs.io/en/stable/getting_started/installation.html)*

Installation¶

```
git clone https://github.com/dieterich-lab/baltica
cd Baltica
pip install .
```

Will install Baltica and its python dependencies. You may want to create a virtual environment *(https://realpython.com/python-virtual-environments-a-primer/)* before installing Baltica.
All other requirements are resolved with singularity containers.

Note

We plan to submit Baltica to the Python Package Index.

Warning

majiq requires an Academic or Commercial license for use. Users are required to obtain their licenses. *(https://majiq.biociphers.org/app_download/)*.

Executing Baltica¶

Use baltica `cli` for current help documentation:

```
baltica --help
```

Baltica executor takes a single optional argument `--verbose`, to detail it execution. Every other option is passed to Snakemake.

Cluster profile¶

Snakemake supports distributed workflow execution in many different high-performance computer clusters, as detailed here *(https://snakemake.readthedocs.io/en/stable/executing/cluster.html?highlight=profile#cluster-execution)*. We recommend using cluster profiles *(https://snakemake.readthedocs.io/en/stable/executing/cli.html#profiles)* and using it like:

```
baltica <workflow> <config> --use-singularity --profile <cluster>
```

(Advanced) Baltica workflows directly from Snakemake¶

Baltica workflows can be used directly with Snakemake without installation. However, there is limited support for it.

References¶

1. If you use Baltica, also please cite Snakemake *(https://bioinformatics.oxfordjournals.org/content/28/19/2520)*
2. If you use majiq results, please cite it *(https://elifesciences.org/articles/11752)*
3. If you use leafcutter results, please cite it *(https://www.nature.com/articles/s41588-017-0004-9)*
4. If you use rmats, please cite it *(https://www.pnas.org/content/111/51/E5593)*
5. If you use junctionseq results, please cite it *(http://nar.oxfordjournals.org/content/early/2016/06/07/nar.gkw501.full)*
6. If you use the Baltica's analysis module, please also cite Stringtie *(https://www.nature.com/articles/nbt.3122)*

(https://github.com/dieterich-lab/Baltica/tree/master/docs/proj-config.md)

Baltica project configuration:¶

Baltica requires a project configuration file as input. For a template see here *(https://raw.githubusercontent.com/dieterich-lab/Baltica/master/baltica/config.yml)*. For a programmatic solution to generate the configuration, use the script baltica/write_new_config.R *(https://github.com/dieterich-lab/Baltica/blob/8bc5fe5f71e948b3971ea5db5f1456b2d9e2f838/baltica/write_new_config.R)*. Method specific parameters are detailed in the Workflow implementation page.

Warning

Baltica requires a full path to files.

Note

The `Required` column flags parameters without default.

Parameters description¶

Parameter	Description	Required
path	project path	✓
sample_path	path to the parent directory for aligment files	✓
samples	sample name and directory formated as "{sample_name}: {path_to_sample)	✓
contrasts	list of contrasts names (format)	✓
assembly	assembly name on UCSC Browser	
strandness	one of fr-firststrand, fr-secondstrand or none (unstranded)	✓
read_len	maximun read length	✓
ref	path to reference annotation in the GTF format	✓
ref_fa	path to reference annotation in the FASTA format	✓
project_authors	project author name, used in the report	
project_title	project title name, used in file names and report	
orthognal_result	result from Nanopore-seq in GFF or BED with a valida score column, and optionally a comparisons column with contrasts	

Pairwise comparisons¶

```
contrasts:
  {case1}-vs-{control}:
    - {case1}
    - {control}
  {cas2}-vs-{control}:
    - {case2}
    - {control}
```

Note

junctionseq and leafcutter support more complex experimental designs, which were not yet implemented in Baltica.

(https://github.com/dieterich-lab/Baltica/tree/master/docs/workflows.md)

Workflow implementation¶

Warning

This chapter is partially outdated.

This chapter details the implementation and usage of each workflow in Baltica.

Baltica comprises a collection of Snakemake workflows (in the SMK format). Each file determines a series of sub-tasks (rules). The sub-tasks run in a specific order; once the output of every rule is complete, the workflow is considered successful. We implemented the workflows following instructions and parameters suggested by the methods authors unless otherwise noted.

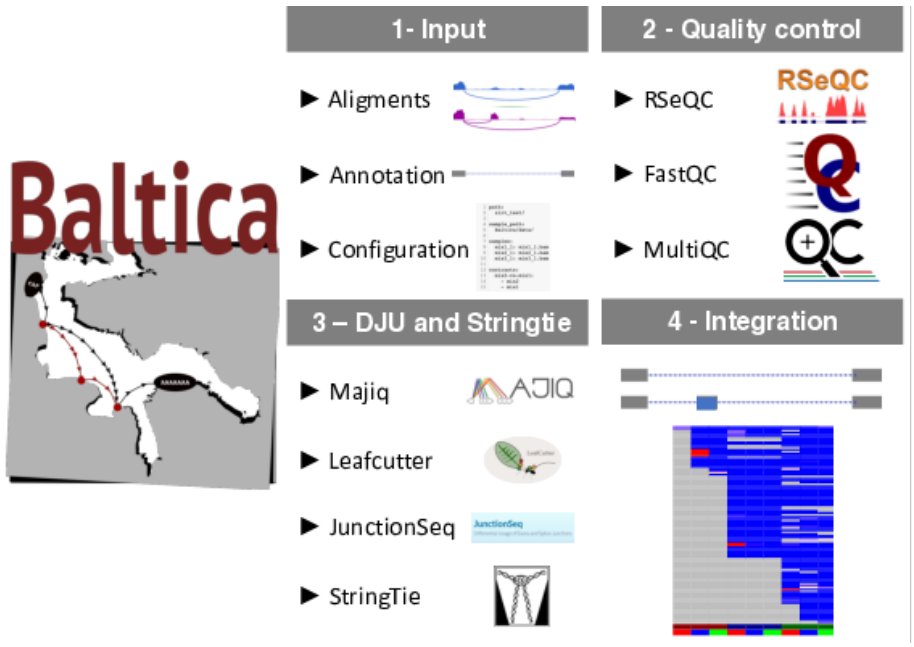

Fig. 1 - Baltica overview: As input (1), Baltica takes the alignment files in the BAM format, and transcriptome annotation and a configuration file that matches the sample names to the alignment file. The file also holds any workflow parameters. In the first optional step (2), Baltica produces a quality control report with MultiQC, which summarizes the results from FastQC and RSeQC. Next (3), Baltica computes the DJU methods and produces a *de novo* transcriptome with Stringtie. The novel exons and transcripts are indispensable for the integration step (4). Finally, the framework parses and integrates the output of the DJU methods.

Which are achieved by successively calling:

1. Input
2. Quality control:

```
baltica qc <config>
```

1. Differential Junction Usage (DJU) and *de novo* transcriptomics:
   ```
   baltica junctionseq <config>
   baltica majiq <config>
   baltica leafcutter <config>
   baltica stringtie <config>
   ```
2. Integration:
   ```
   baltica analysis <config>
   ```

Important

The transcriptome annotation is critical, so make sure you use the same annotation during read alignment and Baltica.

Note

The impact on read alignment on DJU methods results were not fully explored. You should expect different results with different read aligners. We recommend STAR.

Method inclusion criteria¶

There are a plethora of DJU methods, defined as methods that model SJ to identify differential splicing. Bellow is the inclusion criteria we used for Baltica. Methods are required to:

- use as input RNA-Seq read alignment in the BAM format
- detect AS splicing as changes on SJs level, not at the transcript level
- provide test statistic that compares a control group to a case group
- output effect size estimates, such as the deltaPSI
- detect unannotated SJ

We are aware of other methods, such as Suppa [1] and rMATs [2], also fit these criteria and aim to expand the catalog of supported methods in the future.

Quality control workflow¶

The first step comprises the quality control of sequenced libraries and read alignments. This step aims to determine the success of sequencing and alignment. Baltica includes workflows for RSeQC [3] and FastQC. MultiQC [4] summarizes the output from both tools. In addition to the quality control, the tests may suggest biological differences among conditions. For example, RSeQC provides the proportion of reads per feature in the input annotation, which may suggest an enrichment of reads mapping to intronic regions, indicating either intron retention or

accumulation of unspliced mRNA. RSeQC also implements an SJ saturation diagnostic, which quantifies the abundance of known and novel SJ. This metric relates to the sequencing depth. This diagnostic is done by sampling subsets of the dataset to identify which proportion of annotated and novel introns are observed in the sub-samples. In conclusion, the quality control step serves to identify potential problems with the RNA-Seq library alignment and, potentially, direct on further troubleshooting and downstream analysis.

Software dependencies¶

Name	Version
RSeQC	2.6.4
FastQC	0.11.8
MultiQC	0.8

DJU workflow¶

In term of implementation, the DJU tools use the following steps:

1. Extracting split reads from the alignment file
2. Defining which SJ or events should be tested
3. Modeling the SJ/events abundance

Unfortunately, there are differences among the utilized tools that lead to results that are not trivial to compare.

Leafcutter workflow¶

Leafcutter uses a series of scrips to extract the split reads from the BAM files. This step was recently changed to use regtools *(https://github.com/griffithlab/regtools)* to speed up the process. Our test shows that this new step affects the workflow results, and so we have not implemented the change in Baltica.

1. Extracting intron from the alignments files: reads with M and N cigar are extracted from the alignments, giving a minimum read overhang
2. Intron clustering: introns with at least `minclureads` (default: 30) reads and up to `maxintronlen` kb (default: 100000) are clustered. The clustering procedure iteratively discards introns supported by less than `mincluratio` reads within a cluster.
3. Differential splicing analysis: Leafcutter uses a Dirichlet-Multinominal model to model the usage (proportion) of a giving SJ within a cluster and compare this usage among conditions

Info

> By default, Leafcutter clustering does not use the read strandedness information. In Baltica, we override this parameter and use the strand information for clustering.

Output¶

The relevant output files from Leafcutter have the `_cluster_significance.txt` and `_effect_sizes.txt` suffix, which are computed for each comparison.

Column description:

`*_cluster_significance.txt`: 1. `cluster`: `{chromosome}:{intron_start}:{intron_end}` 1. `Status`: is this cluster testable? 1. `loglr`: the log-likelihood ratio between the null model and alternative 1. `df`: degrees of freedom, equal to the number of introns in the cluster minus one (assuming two groups) 1. `p` unadjusted p-value dor the under the asymptotic Chi-squared distribution

`*_effect_sizes.txt`: 1. `intron`: intron identifier on the format `chromosome:intron_start:intron_end:cluster_id` 1. `es`: effect size 1. `{cond_1}`: fitted junction usage in condition `cond_1` 1. `{cond_2}`: fitted junction usage in condition `cond_2` 1. `deltapsi`: difference between usage in the two conditions

Majiq workflow¶

Majiq workflow is implemented as follows:

1. Create a configuration file (`majiq/build.ini`)
2. **Majiq build** generates the Splice Graph database with exons and SJ from the RNA-Seq experiment and the reference annotation
3. **Majiq deltapsi**: - computes PSI and deltaPSI and tests the if the deltaPSI changes between comparisons are significant
4. **Voila tsv**: filter and process the Majiq output

Majiq also provides a visualization with the `voila view` that we find helpful.

Software dependencies¶

Name	Version
python	3.6
htslib	1.9

Output¶

Baltica parses the files `*_voila.tsv` (one per comparison). One can read regarding Majiq's output at Majiq's online documentation *(https://biociphers.bitbucket.io/majiq/VOILA_tsv.html)*

-->

JunctionSeq workflow¶

JunctionSeq [5] tests statistical significance over difference usage among exonic and intronic disjoint genomic bins. It takes as input read count matrix obtained with QoRTs [6], for annotated SJ, novel SJ, and exons, so in fact, JunctionSeq fits both the DEU and DJU classifications. Bins selected are testable as modeled with generalized linear models, as described in DEXSeq [7], but reporting a test statistic at the genomic feature (exon or junction) and gene level. Different from other DJU methods, JunctionSeq does not group the SJ in AS events, and so it does not compute PSI events. By default, SJ with p.adjust < 0.05 are called significant.

Software dependencies¶

Name	Version
R	3.6
qorts	1.1.8

Qorts depends on Java. We currently use it with Java 11.0.6. JunctionSeq itself relies on a series of BioConductor packages.

Output¶

Baltica parses files with the `*_sigGenes.results.txt.gz` suffix (one per comparison). Detailed output information are in the JunctionSeq Package User Manual *(https://github.com/hartleys/JunctionSeq/blob/13a323dda5fae2d7e74b82230824affb747d938d/JunctionSeq/vignettes/JunctionSeq.Rnw#L514)*

Stringtie workflow¶

Baltica uses splice graph information to reconcile the SJ coordinates among methods and to assign AS type.

We process *de novo* transcriptomic workflow with Stringtie [@pertea_2015]. First, we merge the alignment files from biological replicates. Next, we compute *de novo* annotation with Stringtie (v1.3.5) with `-c 3`, `-j 3`, and `-f 0.01`. Finally, the merge the multiple annotation with `gffcompare -r {reference_annotation.gtf} -R -V`. Details on the parameter selection are in the Integration chapter.

Software dependencies¶

Name	Version
Stringtie	1.3.5

Parameters¶

Rule	Name	Default	Note
denovo_transcriptomics	`strandness`	reverse	
denovo_transcriptomics	`min_isoform_proportion`	0.001	
denovo_transcriptomics	`min_junct_coverage`	3	
denovo_transcriptomics	`minimum_read_per_bp_coverage`	3	

1. Juan L. Trincado, Juan C. Entizne, Gerald Hysenaj, Babita Singh, Miha Skalic, David J. Elliott, and Eduardo Eyras. SUPPA2: fast, accurate, and uncertainty-aware differential splicing analysis across multiple conditions. *Genome Biology*, March 2018. URL: https://doi.org/10.1186/s13059-018-1417-1 *(https://doi.org/10.1186/s13059-018-1417-1)*, doi:10.1186/s13059-018-1417-1 *(https://doi.org/10.1186/s13059-018-1417-1)*.

2. 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*, 111(51):E5593–E5601, Dec 2014. URL: http://dx.doi.org/10.1073/pnas.1419161111 *(http://dx.doi.org/10.1073/pnas.1419161111)* , doi:10.1073/pnas.1419161111 *(https://doi.org/10.1073/pnas.1419161111)*.

3. Liguo Wang, Shengqin Wang, and Wei Li. RSeQC: quality control of RNA-seq experiments. *Bioinformatics*, 28(16):2184–2185, June 2012. URL: https://doi.org/10.1093/bioinformatics/bts356 *(https://doi.org/10.1093/bioinformatics/bts356)*, doi:10.1093/bioinformatics/bts356 *(https://doi.org/10.1093/bioinformatics/bts356)*.

4. Philip Ewels, Måns Magnusson, Sverker Lundin, and Max Käller. Multiqc: summarize analysis results for multiple tools and samples in a single report. *Bioinformatics*, 32(19):3047–3048, Jun 2016. URL: http://dx.doi.org/10.1093/bioinformatics/btw354 *(http://dx.doi.org/10.1093/bioinformatics/btw354)* , doi:10.1093/bioinformatics/btw354 *(https://doi.org/10.1093/bioinformatics/btw354)*.

5. Stephen W. Hartley and James C. Mullikin. Detection and visualization of differential splicing in RNA-seq data with JunctionSeq. *Nucleic Acids Research*, pages gkw501, June 2016. URL: https://doi.org/10.1093/nar/gkw501 *(https://doi.org/10.1093/nar/gkw501)*, doi:10.1093/nar/gkw501 *(https://doi.org/10.1093/nar/gkw501)*.

6. Stephen W. Hartley and James C. Mullikin. Qorts: a comprehensive toolset for quality control and data processing of rna-seq experiments. *BMC Bioinformatics*, Jul 2015. URL: http://dx.doi.org/10.1186/s12859-015-0670-5 *(http://dx.doi.org/10.1186/s12859-015-0670-5)*, doi:10.1186/s12859-015-0670-5 *(https://doi.org/10.1186/s12859-015-0670-5)*.

7. S. Anders, A. Reyes, and W. Huber. Detecting differential usage of exons from RNA-seq data. *Genome Research*, 22(10):2008–2017, June 2012. URL: https://doi.org/10.1101/gr.133744.111 *(https://doi.org/10.1101/gr.133744.111)*, doi:10.1101/gr.133744.111 *(https://doi.org/10.1101/gr.133744.111)*.

(https://github.com/dieterich-lab/Baltica/tree/master/docs/integration.md)

DJU methods result integration

Parsing the results of the method¶

The first step in the analysis workflow is parsing and processing the DJU methods' output with `scripts/parse_{method}_output.R` scripts as follows:

1. The resulting text output from the DJU methods is parsed and loaded as R data frames
2. The data frames are pivoted in a longer format to have one junction and one comparison per row

Warning

rmats, majiq, and leafcutter use AS events to test for DJU, so metrics are associated with a group and not the SJ. In Baltica, we split these groups in SJ, and multiple SJ may have the same metric, for example, test statistics.

Warning

majiq and rmats may be assigned multiple scores to an SJ to different AS types. For these, Baltica selects the maximum score as the representative.

Annotating the results¶

We annotate the results with information from the gene and transcript hosting the SJ. For this, we use the *de novo* transcript annotation at `stringtie/merged/merged.combined.gtf`. It's common the multiple transcripts share an intron, and so a single intron may be annotated with multiple transcripts.

One challenge for the integration of DJU methods' results is the different genomic coordinates system. The coordinates system's differences are due to the method implementation design: methods can be 0-indexed (BED format) versus 1-indexed (GTF format) or use the exonic versus intronic coordinates to represent the SJ genomic position.
We propose a `filter_hits_by_diff` that discards any hits with more than two bp differences from overlapping features between the reference (query) and the SJ (subject). Then, using introns obtained in that annotation as a reference and `filter_hits_by_diff`, Baltica enables the reconciliation of the multiple DJU results.

These are the columns assigned after the annotation:

Column name	Description	Note
comparison	pairwise comparison as `{case}_vs_{control}`	
chr	seqname or genomic contig	
start	intron start position for the SJ	1-index
end	intron end position	1-index
strand	RNA strand that encodes that gene	
gene	the gene symbol	
e2	acceptor exon number, if in + strand otherwise donor exon	
e1	donor exon number, if in the + strand otherwise acceptor exon	
tx_id	transcript identifier from the combined annotation	
transcript_name	transcript name	
class_code	association between reference transcript and novel transcript	see fig 1 from this paper for details *(https://doi.org/10.12688/f1000research.23297.1)*

Table 1: Columns added after annotation

Selecting optimal parameters for *de novo* transcriptome assembly¶

We found that the parameters used to obtain the *de novo* transcriptome are critical for maximum integration between the GTF and the SJ from DJU methods. **Fig 1** shows a parameter scan where we vary the group, `-j` (minimum junction coverage), `-c` (minimum coverage), and `-f` (minimum isoform proportion) and compute the number of transcripts that match with SJ called significantly. As expected, the merged annotation and not the group-specific annotation have the highest rate of annotated introns. The crucial result here is the dependency of the `-f` parameter, which is also associated with an increased number of annotated introns. As we confirmed this behavior in other datasets, we decided to use `-c 3 -j 3 -f 0.01` as default values in Baltica. The higher coverage (`-c` and `-j`) values counter the potential noise of transcripts with low abundance.

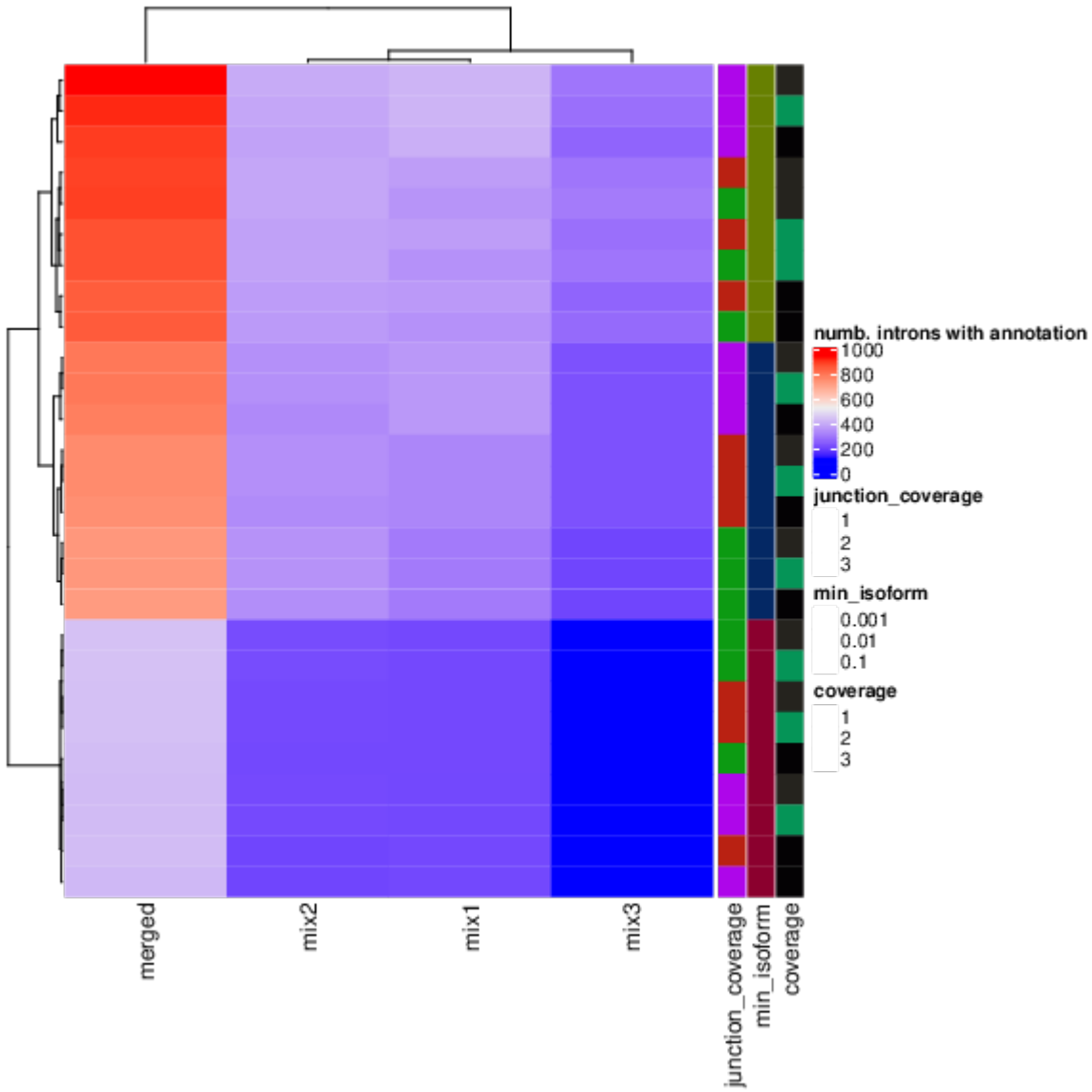

Fig. 6.1: Parameters scan to maximize the number of introns annotated We have run Stringtie with multiple for group annotations and merged annotation. We use a series of parameters: junction coverage of 1, 2, or 3; coverage of 1, 2, 3, and minimum isoform fraction of 0.1, 0.01, or 0.001.

Assigning AS type¶

Biolofical motivation¶

Identifying the type of AS is critical to understand a potential molecular mechanism for AS events. Many factors influence splicing. Splicing factors are probably the most well-studied and can have a specific function in splicing regulation. SRSF2 *(https://www.uniprot.org/uniprot/Q01130)* is a relevant example in this context. SRSF2 is splicing factors from the SR family that are known for auto-regulation. In certain conditions, the SRSF2 transcript can activate the nonsense-mediated decay by either including a new exon containing a premature stop codon or an intron in 3' UTR. These changes lead to transcript degradation and overall reduction of gene expression. Thus, the reduction of SRSF2 protein level leads to widespread exon skipping. Identifying such patterns is critical to understanding which splicing regulators are driving the observed splicing changes.

Implementation¶

In Baltica, we use a geometric approach to define AS in three classes: - ES, for exon skipping - A3SS, for alternative 3' splice-site - A5SS, for alternative 5' splice-site

Figure 2 details how we use the distance between features start and end to determine the AS type.

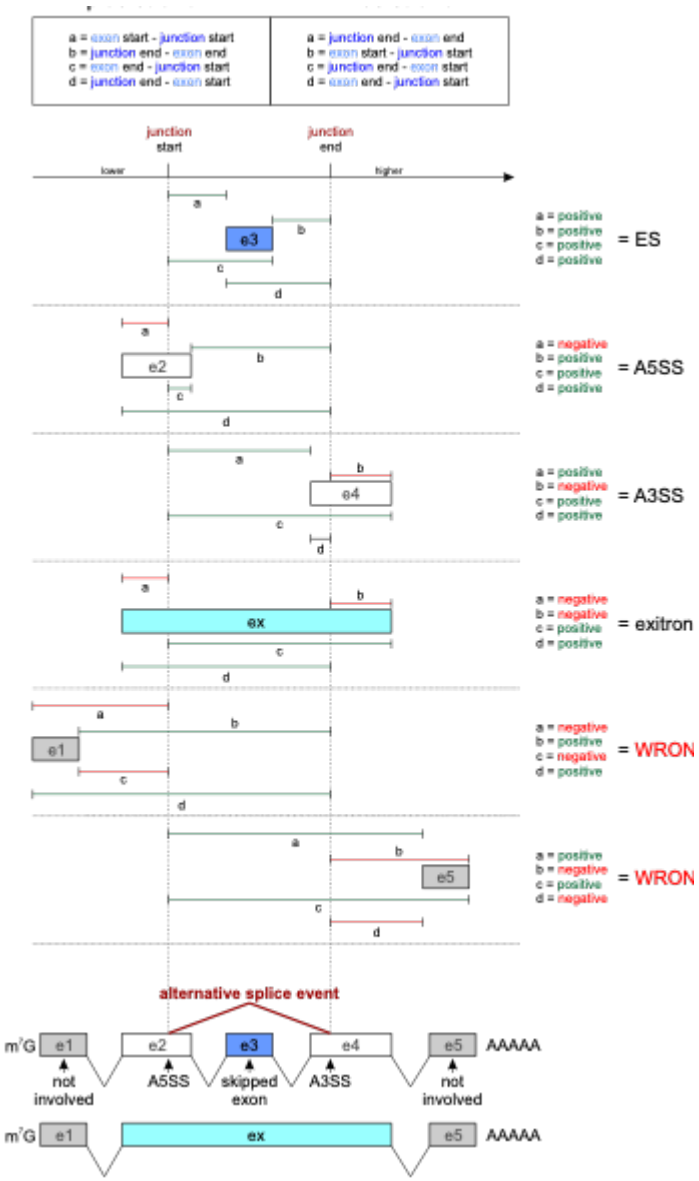

Fig. 2: AS type assignment in Baltica. Baltica uses the genomic coordinates from the SJ and its overlapping exons to assigning AS type to SJ and its overlapping exons. Because many exons may be affected, multiple assignments are output. For example, donor and acceptor exons are assigned as JS and JE, respectively.

Simplify the AS event¶

Because most of the final users are only interested in the list of genomic ranges, gene names, or event types, we offer a simplified output that removes redundant information. This step helps generate a final report.

(https://github.com/dieterich-lab/Baltica/tree/master/docs/dev_guide.md)

Development guidelines:¶

For the docs, we use MkDocs *(https://www.mkdocs.org/)* because of its flexibility:
- mkdocs-material *(https://squidfunk.github.io/mkdocs-material/getting-started/)*: look and feel
- mkdocs-bibtex *(https://github.com/shyamd/mkdocs-bibtex)*: literature reference
- MkPDFs *(https://comwes.github.io/mkpdfs-mkdocs-plugin/getting-started.html)*: PDF version

Setting up mkdocs¶

```
# osx specific settings
conda install pango cairo

pip install mkdocs
pip install mkdocs-material
pip install mkdocs-bibtex
pip install -e git+https://github.com/jwaschkau/mkpdfs-mkdocs-plugi

# osx specific settings
export LC_ALL=en_US.UTF-8
export LANG=en_US.UTF-8
```

Contributing to the documentation¶

Modify any of the doc files¶

```
vi docs/setup.md
```

Test the changes locally¶

```
mkdocs serve
```

If everything looks fine you can submit a patch or pull-request.

Deploy changes¶

This requires permissions from the GitHub organization.

```
mkdocs gh-deploy
```

Testing Baltica¶

Baltica's continuous integration testing suite is under development.

(https://github.com/dieterich-lab/Baltica/tree/master/docs/faq.md)

Frequently asked questions

What you mean with `baltica provides a interface between users and DJU methods`:¶

There are many specificities to the DJU methods, and while running one method is not too complicated, figuring out how to run multiple methods is time-demanding. Baltica aims to facilitate this task so that methods results can be produced and compared.

Snakemake `Error: Directory cannot be locked.`:¶

This error happens when there is an error or failure during the workflow execution, and Snakemake's process does not have the opportunity to unlock the directory. Use `baltica <workflow> <config> --unlock` to resolve it. See more here *(https://snakemake.readthedocs.io/en/stable/project_info/faq.html#how-does-snakemake-lock-the-working-directory)*

rmats empty or mostly empty outputs¶

This error can be either issue with: - The read length parameter *(https://github.com/Xinglab/rmats-turbo/issues/95)*. To resolve it, increase the read length parameter or use `--variable-read-length` in Baltica configuration. - Or an error with the stack size limit *(https://github.com/Xinglab/rmats-turbo/issues/91)*. To resolve it increase the stack size in bash with `ulimit -c unlimited`.

Junctionseq bpapply error:¶

This error occurs in the `junctionseq_analysis` rule, which uses multiple threads with BiocParallel. One can overcome this issue by setting threads to 1 on the said rule.

Junctionseq analysis thread error¶

`Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: Index xxx out of bounds for length xxx` It is complaining the maximum read length is longer than the read length input. First, check the maximum read length in the quality control report and then increase the `read_len` parameter on Baltica config.

How does Baltica compute the score for each DJU method?¶

The different DJU methods are pretty different in many aspects, including how they compute the final test statistic, and we use the following rule to compute the score for the Baltica table (higher is better):

- **majiq** score = 1 - non-changing-threshold (probability of $|\Delta\Psi| > 0.2$, by default)
- **leafcutter** = 1 - p.adjust
- **junctionseq** = 1 - padjust
- **rmats** = 1 - FDR

I see a message: `/bin/bash: /root/.bashrc: Permission denied`. What is wrong?¶

This error message is benign and, in our experience, does not affect workflow execution.

(https://github.com/dieterich-lab/Baltica/tree/master/docs/release-notes.md)

Release notes

Change log¶

Master July 23, 2021 (unreleased)¶

- Add rmats workflow
- Add scrips for parsing for rmats and updated analysis to support the method
- Create the benchmark with the ONT Nanopore-seq
- Update benchmaks, included difference comparison for SIRV benchmark
- Splite annotation and AS type assigment functions
- Update baltica table algorithm
- Add support for singularity container via snakemake, with container recipes `baltica qc config.yaml --use-singularity`
- Add parsing method for gffcompare tracking output
- Update configuration file to expose important parameters from the DJU methods
- Add end-to-end analysis with `baltica all config`
- Experiment with meta-score (gradient boosted trees)
- Add baltica report and improved on report summaries
- Add orthogonal dataset use-case, to integrate third generation sequencing to the baltica table
- Change strand parameter to "fr-firststrand": "reverse", "fr-secondstrand": "forward" or unstranded, fix error in rmats strand

1.1 September 17, 2020¶

- Add `is_novel` column, indication introns not into the reference annotation
- Remove unitended columns (X1, ...) from merge

1.0 - July 23, 2020¶

- First public release comprises of DJU methods Leafcutter, Junctionseq and Majiq. Stringtie for *de novo* transcriptomics assembly. FastQC and MultiQC (#1).