Baltica Documentation

Table of Content

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

1 - Index

Baltica enables integrated differential junction usage (DJU).

Baltica is a framework that provides Snakemake workflows for quality control and supports three DJU methods, Junctionseq, Majiq, and Leafcutter. It uses if *de novo* transcript assembly with Stringtie to reconcile the differently spliced junctions to transcripts. Baltica also provides methods to process and integrate the results from the three tools, allowing us to analyze the global changes in transcription biotype and do further downstream analysis. Our long term aim is to help users to systematically prioritize splice-junctions (SJ) for experimental validation.

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

Contact¶

Please contact us via the GitHub issue tracker *(https://github.com/dieterich-lab/Baltica/issues)*

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

2 - Introduction

The life of RNA transcripts is complex¶

Alternative promoter usage, alternative splicing (AS), and alternative polyadenylation site usage are processes that contribute to the transcriptome complexity by producing different RNA transcript isoforms. AS is defined by a series of enzymatic reactions by which ribonucleoprotein complexes (spliceosomes) sequentially excise introns from a precursor mRNA (pre-mRNA) and ligate the donor exon, at the 5' end of the intron, to the acceptor exon, at the 3' end of the intron. Once complete, the series of splicing events produces an mRNA isoform, which entails the protein-coding sequence. Alternative combinations of exons produce different mRNA isoforms; thus, AS enables one pre-mRNA to generated many transcripts and protein isoforms, consequently diversifying the gene function. In addition, the combinatorial usage of exons augments the possibilities of differential regulation on the transcript level.

Elements of splicing regulation¶

Cis- and trans-acting elements regulate RNA Splicing by orchestrating processes such as the exon-intron boundaries definition. The cis-acting elements are local features, such as the primary and secondary mRNA structure. Trans-acting elements are RNA binding proteins (RBPs) that, once deposited on the mRNA, regulate many steps of splicing. AS and its regulation are essential for many physiological processes, such as development [1] and tissue remodeling [2]. Moreover, defective pre-mRNA splicing, or mis-splicing, has been extensively linked to human disease, as reviewed by [3], and is often associated with genetic variation. A recent study has demonstrated that up to 10% of human genetic variants with causal links to neurodevelopmental disorders are predicted to cause mis-splicing [4].

Consequence of splicing and motivation¶

Changes in pre-mRNA splicing can have drastic consequences for protein function. Spliced isoforms may lead to truncated or extended protein domains, with a significant impact on protein function, for example, AS have been linked to the differential subcellular trafficking of proteins [5]. AS also modulates post-transcriptional regulation events. The inclusion of a premature stop codon, which could be the consequence of AS, activates the nonsense-mediated decay pathway, which leads to the degradation of the transcript and, thus, depletion of the encoded protein [6],[7]. However, experimental evidence to support AS biological consequence is limited in the scientific literature because of the many challenges in detecting and prioritizing AS events. For example, VastDB, a database of experimentally validated splicing events, contains only 1148 events (version 1.8). [8].

Classes of computational methods for AS identification¶

The number of reported AS events has increased enormously over the past ten years. This increase is primarily due to advances in the RNA-Sequencing (RNA-Seq) technology. AS detection benefited primarily of the longer read-length, and the development of computational methods to detect AS from RNA-Seq. Reads that align to two or more exons provide evidence for intron excision. These reads are named exon-exon junctions or splice junctions (SJ) and represented by the N cigar in the read alignment. The splice graph is a network representation in which nodes and edges represent exon and SJ, respectively, and the different network paths form the different transcripts. Methods to detect AS from RNA-Seq build a model based on the feature abundances and its difference between two or more experimental conditions to test for AS differences. These methods fit into three classes: (a) different transcript usage or expression (DTU) [9], [10], [11]; (b) different exon usage [12]; and (c) different junction usage (DJU) [13], [14], [15], [16], [17]. The genomic feature used for statistical modeling determines the class of each method. Most DTU methods depend on a known transcriptome as input, and the results they report rely heavily on the quality of this annotation [18].

DEXSeq applied to differential exon usage¶

DEXSeq is a popular method for DEU that is currently maintained.
To overcome the complexity of the splice graph representation, DEXSeq uses a flat transcript structure representation. It split exons in genomic bins, and partly overlapping exons with alternative a start or end coordinates are split in distinct bins. The bin RNA-Seq read coverage is then modeled with a generalized linear model, which enables the comparison of experimental conditions in context to its covariates. This approach, however, also requires the exon coordinates as input.

DJU methods and their advantages¶

In contrast, DJU methods are less dependent on the annotation and are more robust to ambiguous sequencing that reads that map to many features. The main advantage of DJU versus DTU methods is that PSI, the proportion of reads supporting a path in the splice graph, can be computed directly from the sequencing reads, opposing the transcript level quantification, which depends on the complexity of the splice graph.
Thus, DJU methods enable the *de novo* identification and quantification of SJ from the RNA-Seq data independent of transcriptome assembly methods. This feature enables the identification of many more splicing events that otherwise would be missed. There are considerable

differences in terms of implementation and assumption among DJU methods. Nevertheless, these methods share a set common set of steps that represent the core of a DJU workflow. First, the SJ are extracted from the alignment files; second, testable AS events are selected; third, SJ read counts for the selected AS events are modeled; finally, the test results are reported. Most of the methods also include a visualization of the results to facilitate interpretation.

Current challenges of DJU methods¶

However, these state-of-the-art DJU methods produce results with a low agreement with each other. To demonstrate that, we included the Spike-in RNA Variant Control Mixes (SIRVs) in a recent dataset with knockdown and knockout of the CASC3 gene [19]. By computing DJU with JunctionSeq, Majiq and, Leafcutter, we report little intersection of significant identified genes and SJ on the SIRV artificial transcriptome, for which the ground truth is known (see the Benchmark chapter). The lack of consensus among the different tools represents a barrier to users who want to compare the various methods to select alternatively spliced exon-exon junctions to be experimentally validated.

The aim of this project¶

Despite the increase in the number of known AS events, there are still many challenges to advance our understanding of the molecular mechanism underlying tissue and disease-specific RNA splicing. The lack of consensus among the tools leads to difficulties in selecting which junctions are biologically relevant. This issue challenges the detection and validation of AS events. Here, we introduce Baltica, a framework that simplifies the execution and integration of results from DJU methods and summarizes AS's potential biological consequences from changes in the annotation. Besides presenting Baltica, we also provide a benchmark of the three DJU methods using the Spike-In RNA Variants (SIRVs) as ground-truth for alternative splicing detection.

Tips on RNA-Seq aiming differential splicing detection¶

For RNA-Sequencing experiments aiming to detect genes and transcripts with relatively low expression, a higher sequencing depth (40-60 million reads) is required, in contrast, to experiment that only aim finding the most abundant genes, and so only demand around 10 million reads read here for more details *(https://support.illumina.com/bulletins/2017/04/considerations-for-rna-seq-read-length-and-coverage-.html)*. This parameter is particularly relevant for samples with novel SJ. Read length and paired-end are also critical for splice junction identification, and longer reads offer more coverage of the exons boundaries. The target nominal read length should be between 75-100 nucleotide, maximize the read overhang size, and, consequently, maximize the quality of the alignments.

Also, databases such as the CHESS 2 *(http://ccb.jhu.edu/chess/)* can provide additional evidence for splice sites absent in the annotation.

---

1. Justin J.-L. Wong, William Ritchie, Olivia A. Ebner, Matthias Selbach, Jason W.H. Wong, Yizhou Huang, Dadi Gao, Natalia Pinello, Maria Gonzalez, Kinsha Baidya, Annora Thoeng, Teh-Liane Khoo, Charles G. Bailey, Jeff Holst, and John E.J. Rasko. Orchestrated intron retention regulates normal granulocyte differentiation. *Cell*, 154(3):583–595, August 2013. URL: https://doi.org/10.1016/j.cell.2013.06.052 *(https://doi.org/10.1016/j.cell.2013.06.052)*, doi:10.1016/j.cell.2013.06.052 *(https://doi.org/10.1016/j.cell.2013.06.052)*.

2. Jing Liu, Xu Kong, Mengkai Zhang, Xiao Yang, and Xiuqin Xu. RNA binding protein 24 deletion disrupts global alternative splicing and causes dilated cardiomyopathy. *Protein & Cell*, 10(6):405–416, September 2018. URL: https://doi.org/10.1007/s13238-018-0578-8 *(https://doi.org/10.1007/s13238-018-0578-8)* , doi:10.1007/s13238-018-0578-8 *(https://doi.org/10.1007/s13238-018-0578-8)*.

3. Marina M. Scotti and Maurice S. Swanson. RNA mis-splicing in disease. *Nature Reviews Genetics*, 17(1):19–32, November 2015. URL: https://doi.org/10.1038/nrg.2015.3 *(https://doi.org/10.1038/nrg.2015.3)*, doi:10.1038/nrg.2015.3 *(https://doi.org/10.1038/nrg.2015.3)*.

4. Kishore Jaganathan, Sofia Kyriazopoulou Panagiotopoulou, Jeremy F. McRae, Siavash Fazel Darbandi, David Knowles, Yang I. Li, Jack A. Kosmicki, Juan Arbelaez, Wenwu Cui, Grace B. Schwartz, Eric D. Chow, Efstathios Kanterakis, Hong Gao, Amirali Kia, Serafim Batzoglou, Stephan J. Sanders, and Kyle Kai-How Farh. Predicting splicing from primary sequence with deep learning. *Cell*, 176(3):535–548.e24, January 2019. URL: https://doi.org/10.1016/j.cell.2018.12.015 *(https://doi.org/10.1016/j.cell.2018.12.015)* , doi:10.1016/j.cell.2018.12.015 *(https://doi.org/10.1016/j.cell.2018.12.015)*.

5. Steffen Link, Stefanie E. Grund, and Sven Diederichs. Alternative splicing affects the subcellular localization of drosha. *Nucleic Acids Research*, 44(11):5330–5343, May 2016. URL: http://dx.doi.org/10.1093/nar/gkw400 *(http://dx.doi.org/10.1093/nar/gkw400)*, doi:10.1093/nar/gkw400 *(https://doi.org/10.1093/nar/gkw400)*.

6. B. P. Lewis, R. E. Green, and S. E. Brenner. Evidence for the widespread coupling of alternative splicing and nonsense-mediated mrna decay in humans. *Proceedings of the National Academy of Sciences*, 100(1):189–192, Dec 2002. URL: http://dx.doi.org/10.1073/pnas.0136770100 *(http://dx.doi.org/10.1073/pnas.0136770100)*, doi:10.1073/pnas.0136770100 *(https://doi.org/10.1073/pnas.0136770100)*.

7. S. J. Baserga and E. J. Benz. Beta-globin nonsense mutation: deficient accumulation of mrna occurs despite normal cytoplasmic stability. *Proceedings of the National Academy of Sciences*, 89(7):2935–2939, Apr 1992. URL: http://dx.doi.org/10.1073/pnas.89.7.2935 *(http://dx.doi.org/10.1073/pnas.89.7.2935)* , doi:10.1073/pnas.89.7.2935 *(https://doi.org/10.1073/pnas.89.7.2935)*.

8. Javier Tapial, Kevin C.H. Ha, Timothy Sterne-Weiler, André Gohr, Ulrich Braunschweig, Antonio Hermoso-Pulido, Mathieu Quesnel-Vallières, Jon Permanyer, Reza Sodaei, Yamile Marquez, and et al. An atlas of alternative splicing profiles and functional associations reveals new regulatory programs and genes that simultaneously express multiple major isoforms. *Genome Research*, 27(10):1759–1768, Aug 2017. URL: http://dx.doi.org/10.1101/gr.220962.117 *(http://dx.doi.org/10.1101/gr.220962.117)*, doi:10.1101/gr.220962.117 *(https://doi.org/10.1101/gr.220962.117)*.

9. Cole Trapnell, Brian A Williams, Geo Pertea, Ali Mortazavi, Gordon Kwan, Marijke J van Baren, Steven L Salzberg, Barbara J Wold, and Lior Pachter. Transcript assembly and quantification by rna-seq reveals unannotated transcripts and isoform switching during cell differentiation. *Nature Biotechnology*, 28(5):511–515, May 2010. URL: http://dx.doi.org/10.1038/nbt.1621 *(http://dx.doi.org/10.1038/nbt.1621)* , doi:10.1038/nbt.1621 *(https://doi.org/10.1038/nbt.1621)*.

10. Malgorzata Nowicka and Mark D. Robinson. DRIMSeq: a dirichlet-multinomial framework for multivariate count outcomes in genomics. *F1000Research*, 5:1356, December 2016. URL: https://doi.org/10.12688/f1000research.8900.2 *(https://doi.org/10.12688/f1000research.8900.2)*, doi:10.12688/f1000research.8900.2 *(https://doi.org/10.12688/f1000research.8900.2)*.

11. Kimon Froussios, Kira Mourão, Gordon Simpson, Geoff Barton, and Nicholas Schurch. Relative abundance of transcripts (RATs): identifying differential isoform abundance from RNA-seq. *F1000Research*, 8:213, February 2019. URL: https://doi.org/10.12688/f1000research.17916.1 *(https://doi.org/10.12688/f1000research.17916.1)*, doi:10.12688/f1000research.17916.1 *(https://doi.org/10.12688/f1000research.17916.1)*.

12. 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)*.

13. Yang I. Li, David A. Knowles, Jack Humphrey, Alvaro N. Barbeira, Scott P. Dickinson, Hae Kyung Im, and Jonathan K. Pritchard. Annotation-free quantification of RNA splicing using LeafCutter. *Nature Genetics*, 50(1):151–158, December 2017. URL: https://doi.org/10.1038/s41588-017-0004-9 *(https://doi.org/10.1038/s41588-017-0004-9)*, doi:10.1038/s41588-017-0004-9 *(https://doi.org/10.1038/s41588-017-0004-9)*.

14. 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)*.

15. Jorge Vaquero-Garcia, Alejandro Barrera, Matthew R Gazzara, Juan Gonzalez-Vallinas, Nicholas F Lahens, John B Hogenesch, Kristen W Lynch, and Yoseph Barash. A new view of transcriptome complexity and regulation through the lens of local splicing variations. *eLife*, February 2016. URL: https://doi.org/10.7554/elife.11752 *(https://doi.org/10.7554/elife.11752)*, doi:10.7554/elife.11752 *(https://doi.org/10.7554/elife.11752)*.

16. 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)*.

17. Timothy Sterne-Weiler, Robert J. Weatheritt, Andrew J. Best, Kevin C.H. Ha, and Benjamin J. Blencowe. Efficient and accurate quantitative profiling of alternative splicing patterns of any complexity on a laptop. *Molecular Cell*, 72(1):187–200.e6, October 2018. URL: https://doi.org/10.1016/j.molcel.2018.08.018 *(https://doi.org/10.1016/j.molcel.2018.08.018)*, doi:10.1016/j.molcel.2018.08.018 *(https://doi.org/10.1016/j.molcel.2018.08.018)*.

18. Charlotte Soneson, Katarina L. Matthes, Malgorzata Nowicka, Charity W. Law, and Mark D. Robinson. Isoform prefiltering improves performance of count-based methods for analysis of differential transcript usage. *Genome Biology*, Jan 2016. URL: http://dx.doi.org/10.1186/s13059-015-0862-3 *(http://dx.doi.org/10.1186/s13059-015-0862-3)*, doi:10.1186/s13059-015-0862-3 *(https://doi.org/10.1186/s13059-015-0862-3)*.

19. Jennifer V Gerbracht, Volker Boehm, Thiago Britto-Borges, Sebastian Kallabis, Janica L Wiederstein, Simona Ciriello, Dominik U Aschemeier, Marcus Krüger, Christian K Frese, Janine Altmüller, and et al. Casc3 promotes transcriptome-wide activation of nonsense-mediated decay by the exon junction complex. *Nucleic Acids Research*, Jul 2020. URL: http://dx.doi.org/10.1093/nar/gkaa564 *(http://dx.doi.org/10.1093/nar/gkaa564)*, doi:10.1093/nar/gkaa564 *(https://doi.org/10.1093/nar/gkaa564)*.

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

3 - Getting started

Getting started¶

Baltica contains a collection of workflows and analysis scripts. Workflows are powered by Snakemake (https://snakemake.readthedocs.io/en/stable/) [1]. Analysis is done with the R using Bioconductor packages. We developed and tested the workflows with the Debian Linux distribution (v8.11 Jesse) and conda (4.8.3). We use the module system to test the workflows, but conda usage is similar. Below, we document how to install the Baltica dependencies.

Install miniconda¶

```
wget https://repo.anaconda.com/miniconda/Miniconda3-latest-Linux-x86
bash Miniconda3-latest-Linux-x86_64.sh
```

Or get miniconda here (https://docs.conda.io/en/latest/miniconda.html).

Follow the instructions to finish then installation, which by default is at `$HOME/miniconda3`.

Make sure you initialize conda with `conda init`. You can test whether your installation was successful or not by running `conda --version` and you may need to restart your shell instance.

Clone and install Baltica¶

```
git clone https://github.com/dieterich-lab/baltica
cd baltica
python setup.py install
```

Install Snakemake¶

```
conda install -c bioconda snakemake">=5.2" --yes
```

Danger

Snakemake requires python versions between > 3.4 and < 3.6.

Some of dependencies can be directly created with conda, go to `baltica/envs` directory and install with:

```
conda env create -f stringtie.yml
```

and

```
conda env create -f qc.yml
```

In general, R packages do not play nicely with conda, but we still use it because it's flexibility and the ability to create isolated software environments.

Stringtie citation [5].

Install Majiq¶

Warning

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

Majiq [2] can installation can be problematic, but the recipe below works for us:

```
conda create --name majiq_env python=3.6 htslib "Cython==0.29.14" gi
conda activate majiq_env

ENV_PATH=$HOME/miniconda3/envs/majiq_env
export HTSLIB_LIBRARY_DIR=$ENV_PATH/lib
export HTSLIB_INCLUDE_DIR=$ENV_PATH/include
pip install git+https://bitbucket.org/biociphers/majiq_academic.git#
```

Danger

Confirm that `pip` used is the one from the `majiq_env` with `which pip`

Please inform us if you have issues with this recipe.

Installation Leafcutter¶

Users can install Leafcutter [3] with conda using the following recipe:

```
conda create --name leafcutter python=2.7
conda activate leafcutter
conda install -c bioconda samtools r-base=3.6

TAR='/bin/tar'
Rscript -e "install.packages('devtools', repos='http://cran.us.r-pro
Rscript -e "devtools::install_github('stan-dev/rstantools')"
```

Warning

Only use `TAR='/bin/tar'` or `set TAR '/bin/tar'` if you have problems with devtools selecting `gtar` instead of `tar`.

Warning

If you are experiencing the following the `ERROR: failed to create lock directory` error when trying to install R packages, add the following option to install.package `INSTALL_opts = c('--no-lock')`.

Install Junctionseq¶

JunctionSeq[4] should be installed directly from BioConductor:

```
conda env create -f envs/junctionseq.ym
conda activate leafcutter
Rscript -e "BiocManager::install('JunctionSeq')"
```

Cloning or installing Baltica?¶

Baltica can either installed as a python package or cloned from Github for each project. The installed version of Baltica is more convenient to be used with: `baltica qc config.yml` (as long the dependencies are available). Users who intend to modify the workflows should clone the framework and keep the change under version. See (workflows)[workflows.md] for details on each available workflow configuration and parameters.

Baltica command-line arguments¶

Use the command below to list the command line arguments and their options:

```
baltica --help
```

Executing a Baltica workflow¶

- with the modules system *(https://modules.readthedocs.io/en/latest/index.html)*: `baltica qc config.yml --use-envmodule`
- with conda enviroments: `baltica qc config.yml --use-conda`
- using an external conda environment, like the one we used for Majiq installation: set `majiq_env_prefix = conda activate majiq_env;` in the configuration file

There are alternatives to provide the software dependencies to Snakemake workflows, so feel free to contact them if you need an option.

---

1. If you use Baltica, please also 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 Junctionseq results, please cite it *(http://nar.oxfordjournals.org/content/early/2016/06/07/nar.gkw501.full)*
5. 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/workflows.md)

4 - Methods

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.

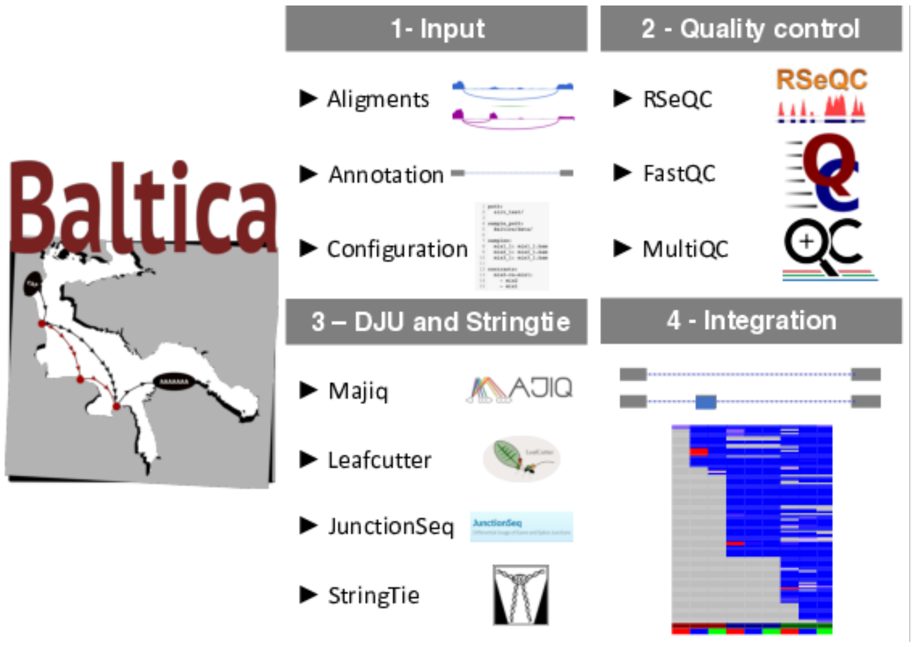


Fig. 4.1 - Baltica overview: As input (1), Baltica takes the alignment files in the BAM format, and transcriptome annotation and a configuration file that matches the sample names to the alignment file. The file also holds any workflow parameters. In the first optional step (2), Baltica produces a quality 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.yml`
3. Differential Junction Usage (DJU) and *de novo* transcriptomics:
   `baltica junctionseq config.yml`
   `baltica majiq config.yml`

```
baltica leafcutter config.yml
baltica stringtie config.yml
```

4. Integration:

```
baltica analysis config.yml
```

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.

Baltica configuration¶

The configuration file contains the parameters for workflows and file paths for input requirements and output destination. We use the JSON file format as a configuration file *(https://snakemake.readthedocs.io/en/stable/snakefiles/configuration.html)*. Please see a minimal working example here *(https://github.com/dieterich-lab/Baltica/blob/master/baltica/config.yml)*.

The following parameters are mandatory:

Parameter name	Description	Note
`path`	absolute path to the logs and results	
`sample_path`	path to the parent directory to the alignment files	
`samples`	sample name to alignment files (BAM format) a condition name	[1]

Parameter name	Description	Note
`comparison`	pair of groups to be tested	
`ref`	full path to the reference transcriptome annotation in the (GTF format)	
`ref_fa`	full path to the reference genome sequence in the FASTA format	[2]
`*_env`	Used if the required dependency is **not** available in the path	[3]
`strandedness`	choice between `reverse`, `forward` or None	[4]
`read_len`	positive integer representing the maximum read length	[5]

Note

Junctionseq and Leafcutter support more complex experimental designs, which were not yet implemented in Baltica.

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 `mincluereads` (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.

Software dependencies¶

Name	Version
python	2.7
R	3.5
samtools	1.9

Parameters¶

Rule	Name	Default	Note
`bam2junc`	`use_strand`	TRUE	
`intron_clustering`	`minclureads`	30	

Rule	Name	Default	Note
`intron_clustering`	`maxintronlen`	100000	
`intron_clustering`	`mincluratio`	0.001	
`differential_splicing`	`checkchrom`	TRUE	
`differential_splicing`	`min_samples_per_group`	2	
`differential_splicing`	`min_samples_per_intron`	2	
`differential_splicing`	`fdr`	0.05	

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

Parameters¶

Rule	Name	Default	Note
create_ini	`assembly`		name of the assembly on the UCSC genome browser
create_ini	`strandness`	reverse	RNA-Sequencing library type
create_ini	`read_len`	100	maximum read length
voila tsv	`majiq_threshold`	0.2	DeltaPSI cutoff for probability calculation

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.

Parameters¶

Rule	Name	Default	Note
qc	`strandness`	reverse	
qc	`read_len`	100	
qc	`is_single_end`	True	

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/benchmark.md)

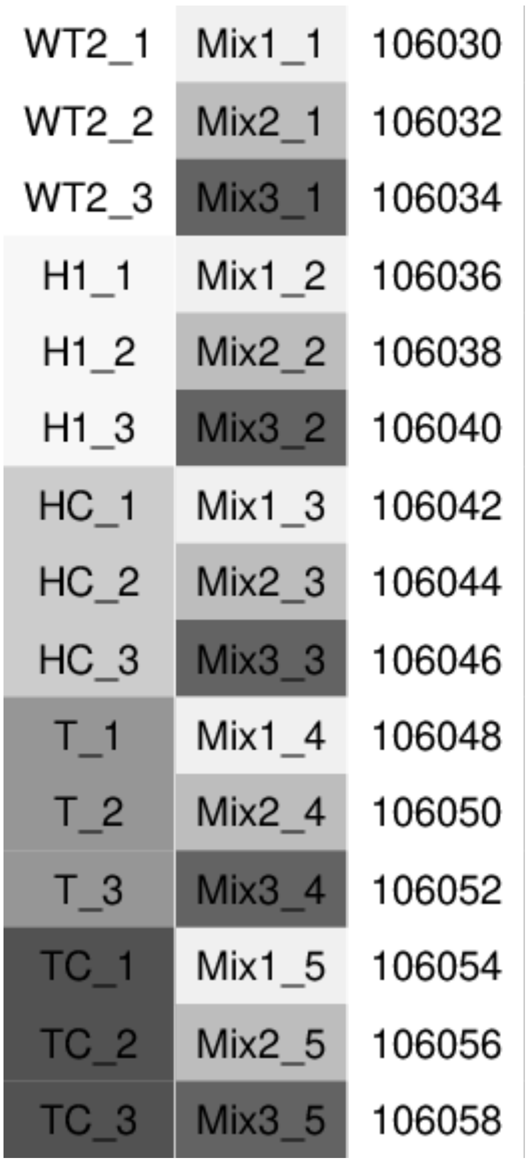
5 - Benchmark with artificial transcripts

We implemented a benchmark method for DJU methods using the Spike-In RNA Variants (SIRVs Set-1, cat 025.03[1]) as ground-truth for alternative splicing identification. Differential splicing methods often use simulated data for benchmark, which does not fully appreciate the complexity of RNA-Sequencing experiment. We use a complementary approach that aims to overcome this limitation. The SIRVs spike-in comprise seven genes, 69 transcript isoforms, 357 exons and 113 intron According to our experimental design (**Fig. 5.1**), differences in transcript abundance lead to differential splicing events in SIRV chromosomes but not in the human contigs, which would represent false-positive calls.

To benchmark the dataset, we use Baltica to run the three tools comparing three groups:
- mix 2 versus mix 1
- mix 3 versus mix 2
- mix 3 versus mix 1

Important

The test data currently shipping with Baltica has only a subset of 10% of the reads per group in Gerbracht et al. 2020 (E-MTAB-8461 *(https://www.ebi.ac.uk/arrayexpress/experiments/E-MTAB-8461/)*).



WT2_1	Mix1_1	106030
WT2_2	Mix2_1	106032
WT2_3	Mix3_1	106034
H1_1	Mix1_2	106036
H1_2	Mix2_2	106038
H1_3	Mix3_2	106040
HC_1	Mix1_3	106042
HC_2	Mix2_3	106044
HC_3	Mix3_3	106046
T_1	Mix1_4	106048
T_2	Mix2_4	106050
T_3	Mix3_4	106052
TC_1	Mix1_5	106054
TC_2	Mix2_5	106056
TC_3	Mix3_5	106058

Fig. 5.1:

Experimental design: The first column represents the experimental groups. See [2] for detail on the biological differences among groups. The second column represents the SIRV mixes. The third column is the sample id from the sequencing facility.

RNA-Seq processing and mapping¶

Cell lines, RNA extraction, and RNA-Seq were described by [2]. In short, we obtained 15 libraries from Flp-In T-REx 293 cells, extracted the RNA fraction, with TrueSeq Stranded Total RNA kit (Illumina), followed by ribosomal RNA depletion, with RiboGold Plus kit. Reads were sequenced with an Illumina HiSeq4000 sequencer using PE 100bp protocol and yield around 50 million reads per sample. Data is deposited in ArrayExpress (E-MTAB-8461). Sequenced reads' adapters and low-quality bases were trimmed, and reads mapping to human precursor ribosomal RNA were discarded. The remaining reads were aligned to the human transcriptome (version 38, EnsEMBL 90) extended with the SIRV annotation. Regarding the DJU method benchmarking, we are not interested in the biological condition, but the SIRVs AS event, and so this experiment was designed, so the SIRVs mixes were not confounded to the biological factors.

DJU method produces results with little agreement ¶

To demonstrated the previously observed lack of consensus among DJU methods, we computed DJU with JunctionSeq, Majiq, and Leafcutter with the SIRVs, for three mixes containing a variable abundance of each transcript. The difference abundances are equivalent to differences in junction usage. The mixes were incorporated into the RNA-Seq experiment in non-confounded design with the biological co-variates, so the SJ observed in the natural contigs (human chromosome) could be used as false-positive calls. **Table 5.1** shows the number of SJ identified and called significant (see the Workflow Chapter for detains on how the we reconcile these results). Majiq and JunctionSeq call around 310 significant SJ, while Leafcutter calls 201. Overall, the methods produce a comparable number of SJ called significant.

-	Junctionseq	Majiq	Leafcutter
Total	608	2850	131676
Significant	311	309	201

Table 5.1: Number of SJ considered (Total) and called significant (Significant) for each method.

However, when comparing the resulting gene and SJ sets, we observe a limited intersection among the results from the three methods (**Fig. 5.2**) for genes and SJ. To further understand this issue, we reviewed the step-by-step process implemented by the methods. Overall, the three DJU methods share three steps, as mentioned before:

- Reads are extracted and filtered from the read alignments.
- Testable SJ or splicing events are identified, for example, by clustering.
- The SJ or splicing events counts are modelled.

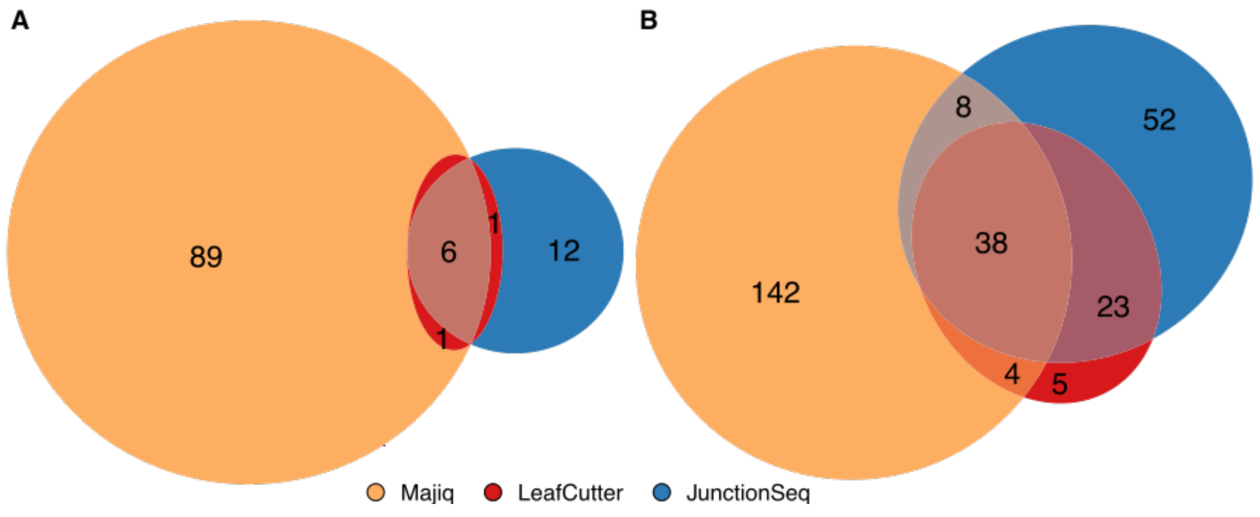


Fig. 5.2: Overlap between genes and SJ called significant by three DJU methods. JunctionSeq, Majiq and, Leafcutter deliver results that show a limited agreement among each other. We computed differential junction usage using the three methods on 15 RNA-Seq libraries with 3

groups, the three SIRV mixes. The resulting intersection shows that only small overlap genes (**A**) and SJ (**B**) called significant by the tools 9% (6 out of 66) and 16% (34 out of 212), respectively.

Benchmark¶

We extracted all introns from the SIRV transcripts and classified between diff (changing between mixes) and no diff. Given the binary outcome from each tool, we analysed the following cases:

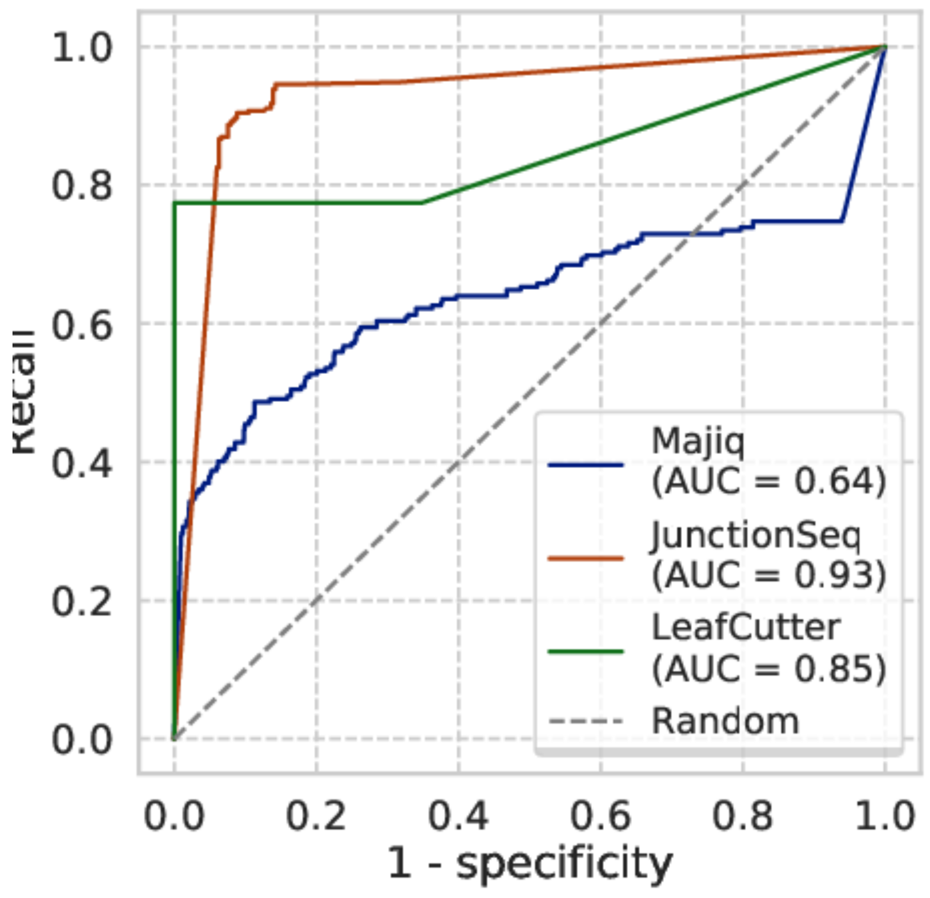
- True positive: diff intron called significant
- False-positive: no diff intron or intron in human contig called significant
- True negative: no diff not called significant
- False-negative: diff intron not called significant

Results from the methods benchmark¶

	TN	**FP**	**FN**	**TP**
Majiq	2492	126	90	82
JunctionSeq	273	43	24	268
Leafcutter	131474	43	1	158

Table 5.2: Methods and number of SJ in each classification case

Junctionseq achieves the highest number of TP calls and Junctionseq and Leafcutter the lowest number of FP calls. Based on that procedure, we proceeded with a two-class validation and computed the Receiver Operating Characteristic (ROC) curve and the precision-recall curve.


Recall
1 - specificity
0.0
0.2
0.4
0.6
0.8
1.0
Majiq (AUC = 0.64)
JunctionSeq (AUC = 0.93)
LeafCutter (AUC = 0.85)
Random

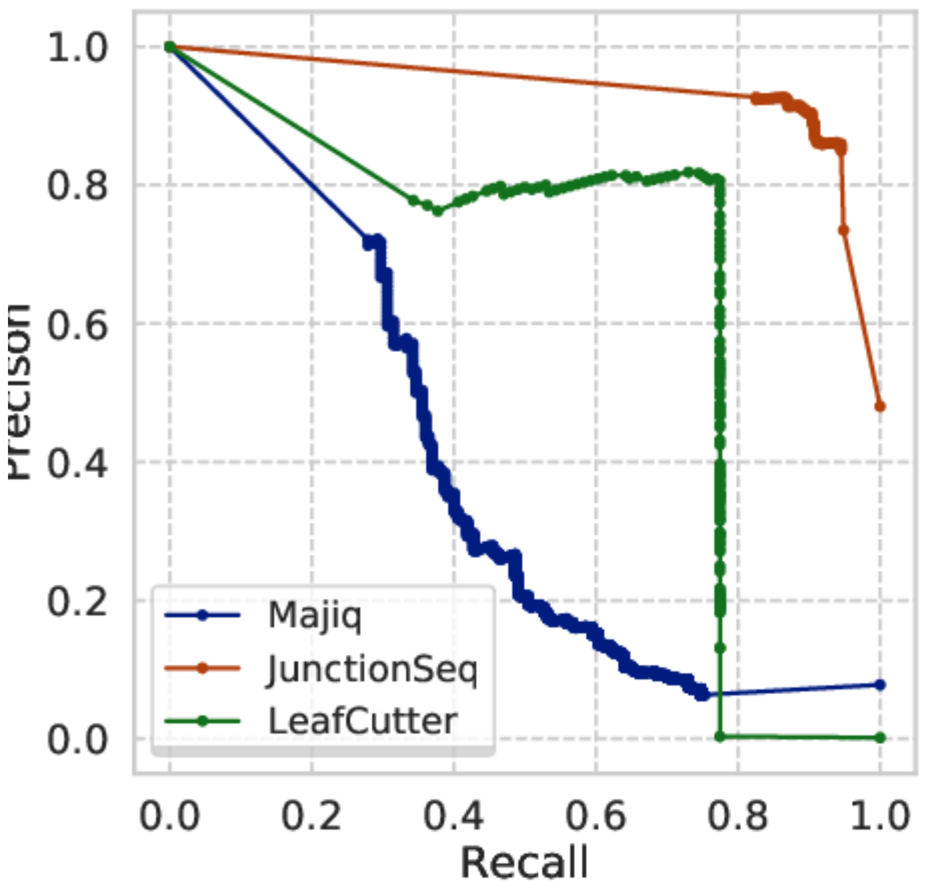


Fig 5.3 ROC and Precision-recall curve: we observe the in this benchmark JunctionSeq outperforms Majiq and Leafcutter.

The curves shows that Junctionseq outperforms the other two methods in the conditions we have analysed.

Despite the punctual similarities, the implementation of these three state-of-art methods vary. Here we present a direct comparison on a set of artificial transcripts. The cumulative differences in who to define which SJ are extracted and tested; how much the method relies on the annotation; and the framework for statistical modelling explain the small intersections in **Fig. 5.2** .

In this case, Leafcutter suffers from a recall ceiling issue, because it ignores certain introns from the artificial transcripts. JunctionSeq has a clear advantage in this comparison because it relies heavily on the information of the annotation and the input annotation, that in this case, perfectly represents the structure of the transcripts. Majiq suffers from an excess of false-positive calls. Baltica aims to further study these implementations differences and potentially suggestion advice to minimize the effect of the limitations mentioned above.

1. https://www.lexogen.com/sirvs/#sirvsdownload - Lexogen took no part in the experimental design neither we have any relationship with the company.

2. Jennifer V Gerbracht, Volker Boehm, Thiago Britto-Borges, Sebastian Kallabis, Janica L Wiederstein, Simona Ciriello, Dominik U Aschemeier, Marcus Krüger, Christian K Frese, Janine Altmüller, and et al. Casc3 promotes transcriptome-wide activation of nonsense-mediated decay by the exon junction complex. *Nucleic Acids Research*, Jul 2020. URL: http://dx.doi.org/10.1093/nar/gkaa564 *(http://dx.doi.org/10.1093/nar/gkaa564)*, doi: 10.1093/nar/gkaa564 *(https://doi.org/10.1093/nar/gkaa564)*.

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

6 - 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
3. Finally, SJ not called significant are discarded

The default significant cut-off is an adjusted p-value < 0.05 for JunctionSeq and Leafcutter or probability of changing > 0.90 for Majiq.

The output results in `{method}/{method}_junction.csv` are in a `tidy` format and can be used for downstream analysis.

> **Note**
>
> Although it is generally good to filter results with small effect sizes, discarding results with small deltaPSI can be problematic. First, it's not trivial to assign deltaPSI value to JunctionSeq results, since the tool does not detect splicing events. Second, SJ with small deltaPSI may indicate RNA degradation in a cell with intact degradation machinery.

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` (below) that discard any hits with more than two bp differences from overlapping features between the reference (query) and the SJ (subject). Using introns obtained in that annotation as a reference, and `filter_hits_by_diff`, Baltica enables the reconciliation of the multiple DJU results.

```
filter_hits_by_diff <- function(query, subject, max_start = 2, max_e
  stopifnot(is(query, "GRanges"))
  stopifnot(is(subject, "GRanges"))
  hits <- findOverlaps(query, subject)
  query <- query[queryHits(hits)]
  subject <- subject[subjectHits(hits)]
  start_dif <- abs(start(query) - start(subject))
  end_dif <- abs(end(query) - end(subject))
  hits <- hits[start_dif <= max_start & end_dif <= max_end]
  hits
}
```

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 6.1: 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 6.1 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 significant. 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.

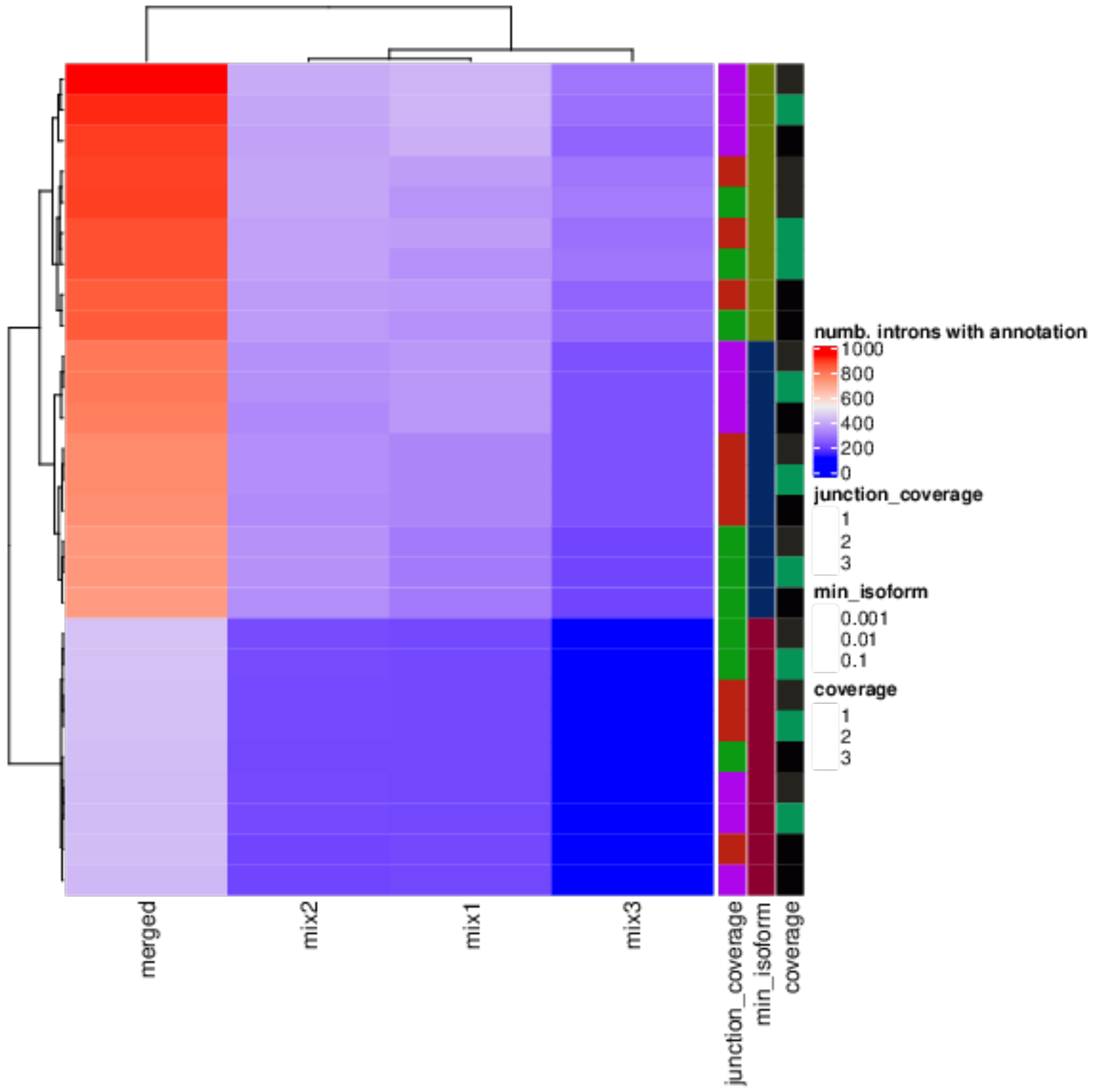


Fig. 6.1: Parameters scan to maximize the number of introns annotated We have run Stringtie with multiple for group annotations and merged annotation. We use a series of parameters: junction coverage of 1, 2, or 3; coverage of 1, 2, 3, and minimum isoform fraction of 0.1, 0.01, or 0.001.

Assigning AS type¶

Identifying the type of AS is critical to understand a potential molecular mechanism for AS events. Many factors influence splicing. Among then, splicing factors 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 it's 3' UTR. These changes lead to the transcript degradation and overall reduction of the gene expression. SRSF2 increased the probability of exons with weak splice site signals to be included in the transcript. 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.

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

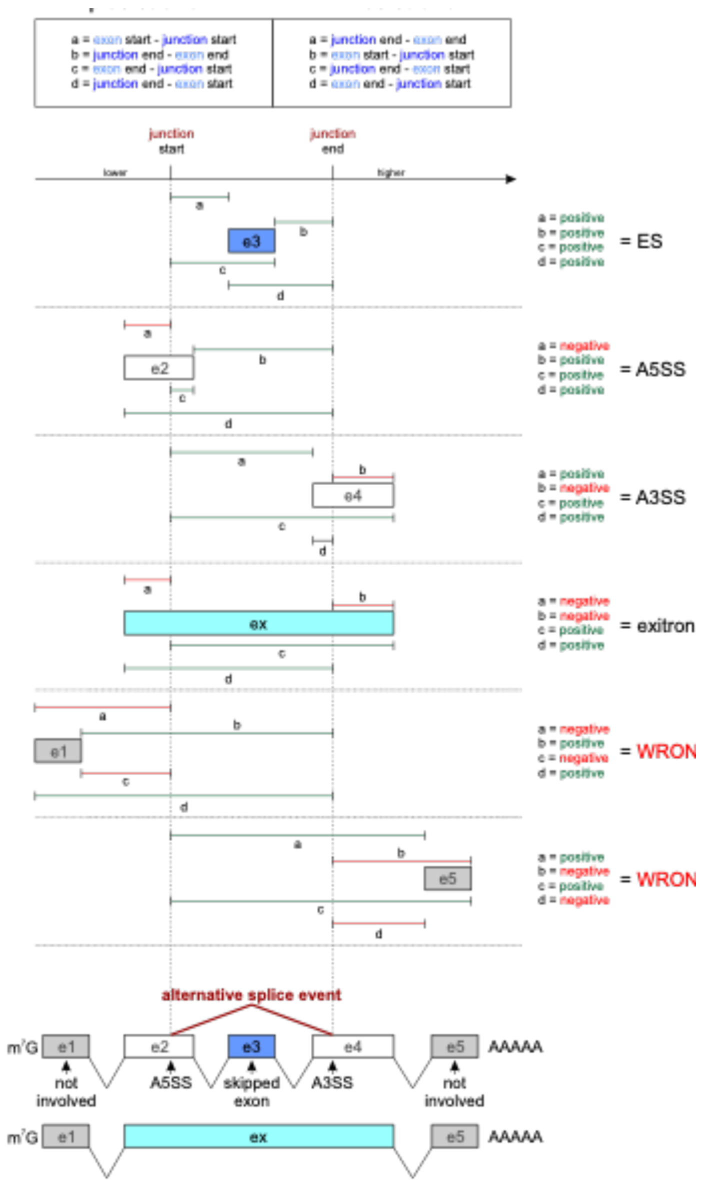

a = exon start - junction start
b = junction end - exon end
c = exon end - junction start
d = junction end - exon start
a = junction end - exon end
b = exon start - junction start
c = junction end - exon start
d = exon end - junction start
junction start
junction end
lower
higher
e3
a = positive b = positive c = positive d = positive = ES
e2
a = negative b = positive c = positive d = positive = A5SS
e4
a = positive b = negative c = positive d = positive = A3SS
ex
a = negative b = negative c = positive d = positive = exitron
e1
a = negative b = positive c = negative d = positive = WRON
e5
a = positive b = negative c = positive d = negative = WRON
alternative splice event
m7G e1 e2 e3 e4 e5 AAAAA
not involved
A5SS
skipped exon
A3SS
not involved
m7G e1 ex e5 AAAAA

Fig. 6.2: AS type assignment in Baltica. Baltica uses the genomic coordinates from the SJ and its overlapping exons to assigning AS type to SJ and it's overlapping exons. Because many exons may be affected, multiple assignments are output. 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 the excess of redundant information. This step is useful for generating a final report.

Below the `{r} sessionInfo()` output for packages loaded for the analysis workflow.

```
R version 3.6.0 (2019-04-26)
Platform: x86_64-pc-linux-gnu (64-bit)
Running under: Debian GNU/Linux 8 (jessie)
```

```
Matrix products: default
BLAS:   /beegfs/biosw/R/3.6.0/lib/R/lib/libRblas.so
LAPACK: /beegfs/biosw/R/3.6.0/lib/R/lib/libRlapack.so

locale:
 [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C
 [3] LC_TIME=en_US.UTF-8        LC_COLLATE=en_US.UTF-8
 [5] LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
 [7] LC_PAPER=en_US.UTF-8       LC_NAME=C
 [9] LC_ADDRESS=C               LC_TELEPHONE=C
[11] LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C

attached base packages:
[1] parallel  stats4    stats     graphics  grDevices utils     data
[8] methods   base

other attached packages:
 [1] openxlsx_4.1.5       reshape2_1.4.4       rtracklayer_1.46.0
 [4] GenomicRanges_1.38.0 GenomeInfoDb_1.22.1  IRanges_2.20.2
 [7] S4Vectors_0.24.4     BiocGenerics_0.32.0  optparse_1.6.6
[10] dplyr_0.8.5          readr_1.3.1          stringr_1.4.0
[13] tidyr_1.0.3

loaded via a namespace (and not attached):
 [1] zip_2.0.4                   Rcpp_1.0.4.6
 [3] plyr_1.8.6                  pillar_1.4.3
 [5] compiler_3.6.0              XVector_0.26.0
 [7] bitops_1.0-6                tools_3.6.0
 [9] zlibbioc_1.32.0             lattice_0.20-38
[11] lifecycle_0.2.0             tibble_3.0.1
[13] pkgconfig_2.0.3             rlang_0.4.6
[15] Matrix_1.2-17               DelayedArray_0.12.3
[17] cli_2.0.2                   GenomeInfoDbData_1.2.2
[19] Biostrings_2.54.0           vctrs_0.2.4
[21] hms_0.5.3                   grid_3.6.0
[23] getopt_1.20.3               tidyselect_1.0.0
[25] Biobase_2.46.0              glue_1.4.0
[27] R6_2.4.1                    fansi_0.4.1
[29] BiocParallel_1.20.1         XML_3.99-0.3
[31] purrr_0.3.4                 magrittr_1.5
[33] matrixStats_0.56.0          GenomicAlignments_1.22.1
[35] Rsamtools_2.2.3             ellipsis_0.3.0
[37] SummarizedExperiment_1.16.1 assertthat_0.2.1
[39] stringi_1.4.6               RCurl_1.98-1.2
[41] crayon_1.3.4
```

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

Release notes

Change log¶

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).