

Bayesian Analysis of Allele Specific Expression (BayesASE): Galaxy Interface User Guide

Brecca Miller, Kelsey Sinclair, Zihao Liu, Gavin Gamble, Alison Morse

University of Florida

Gainesville, Florida

Updated September 2020

TABLE OF CONTENTS:

BayesASE Structure Overview 3

BayesASE Data Formats 4

Genotype Specific References Module 8

Module Overview

Module Inputs

Module Outputs

Module Considerations

Workflow Execution Protocols

Check Alignment Design File Module

Module Overview

Align and SAM Compare Module

Module Overview

Module Inputs

Module Outputs

Module Considerations

Workflow Execution Protocols

Prior Calculation Module

Module Overview

Module Inputs

Module Execution Protocols

Module Outputs

Module Considerations

Bayesian Model Module

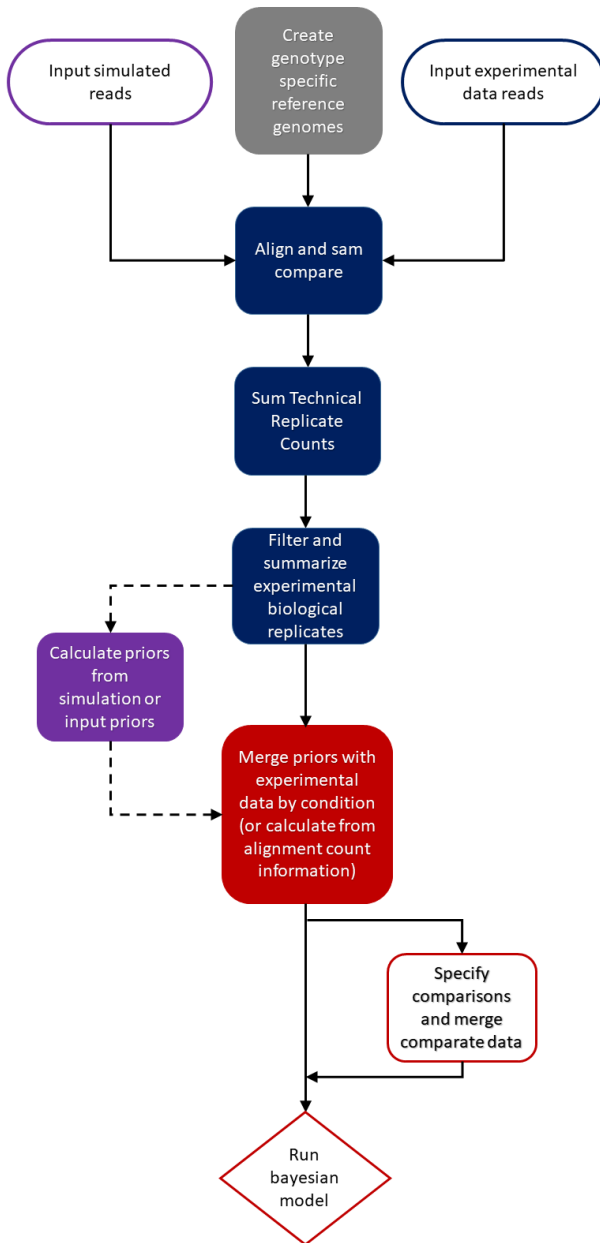
Module Overview

Module Inputs

Module Execution Protocols

Module Outputs

Module Considerations



BayesASE Structure Overview

BayesASE is divided into four distinct modules. Modules are flexible. These modules can be used as part of other pipelines, and other tools with similar functions can be used in place of individual modules. “Genic feature” is used here as a general term to describe a segment of sequence interesting to the researcher (ex: gene, exon).

Module 1:

The *Genome Specific References Module* can be used to generate the genome specific references from a user-supplied FASTA reference and two VCF files. Genome specific references are required as input into the *Align and Sam Compare Module*.

Module 2:

The *Align and Sam Compare Module* aligns reads from individual FASTQ files to user-specified genome specific references and generates files containing read counts for each genic region (bed file). Read counts for technical replicates (e.g. FASTQ data generated from the same library run on separate lanes) can be summed together and the average coverage per nucleotide (APN) is calculated for each genic feature. The data are then summarized for each replicate using a user-specified APN value to determine whether a genic feature has enough read coverage for Bayesian analysis (flag_analyze = 1 if at least 1 compare replicate has an APN value greater than the user-specified cutoff value).

Module 3:

The *Prior Calculation Module* takes summarized read counts as input in the format of the *Align and Sam Compare Module* (Module 2). The data are then

used to estimate priors for the Bayesian model. Simulated data, DNA data, or the RNA read counts can be used. The *Prior Calculation Module* calculates priors in the format expected by the Bayesian Model Module.

Module 4:

The *Bayesian Model Module* (Module 4) merges the prior estimation values with the RNA count data (output from the *Align and Sam Compare Module*) and runs the R code to test allelic imbalance (AI) for the user-specified comparison(s). The comparison(s) to be evaluated are specified using a user supplied design file. A format of the design file exists that allows the model to be evaluated in a single condition.

Running BayesASE Workflows BayesASE workflows are collections of tools designed to be run together. Individual workflows are run in a specific order as the output of one workflow is the input for a following workflow. Before running a workflow, users should confirm inputs are correct. We note that some Galaxy installs may hide input parameters that are not datasets when running workflows, so it is important to ensure that all inputs are correct.

2. Input the Collection of Files with Merged Priors and Newly Generated Headers - Input the file or collection of files that contain the ASE Counts tabled merged with the prior probability calculations, with the correct headers needed for recognition by the Bayesian tool. This is output from the Merge Priors workflow

No data dataset collection available.

3. Enter the number of compare conditions - Type in the number of compare conditions being tested for allelic imbalance. (Default number is 2)

4. Enter the Number of model Iterations - Default is 100000

5. Enter the Warmup Number - Default is 10000. Enter number of iterations needed for tuning the Markov Chain Monte Carlo process before keeping estimates

The example above gives an example of a hidden parameter, denoted by the closed eye. Click the closed eye on the right hand corner to expand it and entering any necessary values.

BayesASE platform availability

BayesASE is implemented in Galaxy, SLURM based sbatch scripts and Nextflow scripts on the McIntyre Lab Github in the BayesASE project repository (<https://github.com/McIntyre-Lab/BayesASE>). Tools can be downloaded and installed from the Galaxy toolshed (https://testtoolshed.g2.bx.psu.edu/repository?repository_id=ef69fe5507b8d8c7&changeset_revision=8b2027117ce5). Conda (<https://anaconda.org/bioconda/bayesase>) and Pypi (<https://pypi.org/project/BayesASE/>) packages are available for download and install.

BayesASE Data Formats

1. VCF standard format

The variant call format (VCF) text file contains location data for SNPs. The Sanger Institute website can be visited for more information on the standard format of a VCF. Indels and structural variants change the location of genic features. This complexity is not accounted for in this version of BayesASE.

2. FASTA standard format

A text-based format used for storing nucleotide or protein sequences.

3. TSV format

Intermediate and output text format files in Galaxy are tab-separated.

4. FASTQ standard format

A text-based format used for storing nucleotide data and associated quality scores.

5. BED 4-column format

4-column tab-separated format for storing genomic regions as coordinates with associated annotations. The four columns, in order, are:

- Chromosome name
- start position
- end position
- name/annotation

7. SAM standard format

A SAM file is a tab-separated text format containing an optional header section and an alignment section.

8. Design file descriptions

All design files used in Galaxy are tab-separated text format. Spaces are NOT allowed in the contents of any design file field (e.g. 'W55_M_1' is ok but 'W55 M 1' is not). There are 2 design files that must be created and supplied by the user: (1) the Alignment Design File and (2) the Compare Design File. The Alignment Design File describes the FASTQ files and samples from which they were derived. The Compare Design File describes the compares and the comparison the user wants to test. Other design files (Sample Design File and Priors Design File) are created within the BayesASE workflows and used as intermediate files. Here we use the term 'compare' to indicate one of two conditions to be compared. For example, in a comparison of female to male, compare 1 would be female and compare 2 would be male. Note that designation of compare 1 and 2 is for tracking and the results are not sensitive to this designation. For example, in the comparison between the two sexes, if compare 1 was male and 2 was female the test of AI between male and female would not be different.

Alignment Design File

The unique key in this file is an individual FASTQ filename. The Alignment Design File is supplied by the user and is required by the *Align and Sam Compare Module*. It consists of seven fields of information that describe the FASTQ files and associated samples. The sampleID names will be used in subsequent tools for generating and labeling data files, therefore, the

FASTQ files in the design file must be sorted according to biological replicate number in the sampleID column. If running simulated data or data without technical replicates, the techRep field is still required – in this case the techRep column should contain a '1' for all FASTQ files.

1. G1: Name of genome specific reference 1.
2. G2: Name of genome specific reference 2.
3. sampleID: Sample identifier where the final “_#” contains the biological replicate number.
4. fqName: Name of the FASTQ file. Do not include extension. These should be unique.
5. fqExtension: FASTQ suffix (e.g. .fq or .fastq) .
6. techRep: Number indicating the technical replicate for a sampleID.
7. readLength: Length of the reads in the FASTQ file.

The column names must be labeled as above.

BayesASE in Galaxy is designed to be run using collections of FASTQ files – collections allow the same analyses to be carried out on all FASTQ files in the collection. Therefore, the G1 and G2 columns in the Alignment Design File supplied by the user **MUST** come from the same F1 because the same analyses will be for each F1. Note here that reciprocal crosses can be included in the same design file as long as the user has a naming convention for the sample IDs and does not switch the designation of G1 and G2. That is, the tracking of G1 and G2 as maternal and paternal should be part of the users naming convention and is not an explicit part of the structure of BayesASE.

An example is shown below:

G1	G2	sampleID	fqName	fqExtension	techRep	readLength
W1118	W55	W55_M_1	W55_M_1_1	.fastq	1	96
W1118	W55	W55_M_1	W55_M_1_2	.fastq	2	96
W1118	W55	W55_M_1	W55_M_1_3	.fastq	3	96
W1118	W55	W55_M_1	W55_M_1_4	.fastq	4	96
W1118	W55	W55_M_2	W55_M_2_1	.fastq	1	96
W1118	W55	W55_M_2	W55_M_2_2	.fastq	2	96
W1118	W55	W55_M_2	W55_M_2_3	.fastq	3	96
W1118	W55	W55_M_2	W55_M_2_4	.fastq	4	96
W1118	W55	W55_M_3	W55_M_3_1	.fastq	1	96
W1118	W55	W55_M_3	W55_M_3_2	.fastq	2	96
W1118	W55	W55_M_3	W55_M_3_3	.fastq	3	96
W1118	W55	W55_M_3	W55_M_3_4	.fastq	4	96

Sample Design File

The unique key in this file is an individual sample name. As part of the *Align and Sam Compare Module* (specifically by the Combine ASE Counts Tables Tool), a design file is generated from the user-supplied Alignment Design File, designated as the Sample Design File. This file is used as input into the Summarize SAM Compare Counts tool to merge samples belonging to the same condition). The Sample Design File contains the following fields:

1. G1: Name of genome specific reference 1.
2. G2: Name of genome specific reference 2.
3. sampleID: Sample identifier where the final “_#” contains the replicate number.
4. compare: Contains the conditions being tested for allelic imbalance

An example design file is shown here:

```
G1      G2  sampleID compare
W1118 W55 W55_M_1  W55_M
W1118 W55 W55_M_2  W55_M
W1118 W55 W55_M_3  W55_M
W1118 W55 W55_V_1  W55_V
```

Priors Design File

Each genotype may have different priors associated with mapability of the genic features for the two parental genotypes. Priors are specified for each genotype. This file specifies which two genotypes are associated with each individual condition.

1. G1: Name of genome specific reference 1.
2. G2: Name of genome specific reference 2
3. compare: identifier/label for comparison

An example priors design file:

```
G1      G2  compare
W1118 W55 W55_M
W1118 W55 W55_V
```

Compare Design File

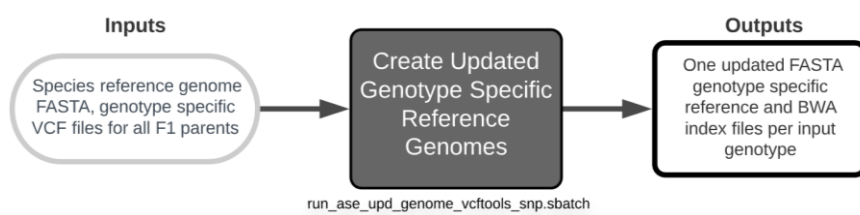
This is a user-supplied design file that specifies the individual compares for each comparison. This design file is used to combine the appropriate compare files together (in the Merge Compares Tool). The Compare Design File must contain the following fields (adjust according to number of compares present):

1. Compare_1: comparison condition 1
2. Compare_2: comparison condition 2
3. compID: identifier/label for comparison

An example below to test for AI between M and V in an F1 cross between W118 and W55 genotypes:

```
Compare_1 Compare_2  compID
W55_M      W55_V    W55_M_V
```

Genotype-Specific References Module



Module Overview

The *Genotype Specific Reference Module* enables BayesASE users to create genotype-specific references given a set of user-supplied VCF files and a reference FASTA file. The genome specific reference is used to reduce mapping biases that may occur when a common reference is used. We explicitly note that this module leverages the `vcftools_consensus` tool already available in the Galaxy Toolshed (Sanger Institute).

VCFTools Consensus identifies differences between the user-supplied FASTA and VCF files, and creates a genotype specific reference by replacing reference bases in the user-supplied FASTA file with the alternate bases in the VCF file. The tool should be run separately for each genotype.

Module Considerations

Users that have genome specific references from another process may omit this module.

Module Workflow

The *Genotype Specific References Module* can be run as a Galaxy workflow by importing and running `genotype_specific_referenceworkflow.ga` (<https://github.com/McIntyre-Lab/BayesASE>).

Module Inputs

The *Genotype Specific Reference Module* requires two inputs for each parental genome:

- 1) The reference genome FASTA file
- 2) A VCF file containing SNPs only

Module Outputs

The module creates one output file for each set of inputs:

- 1) A genotype specific reference FASTA file

Step 1: Component script: `vcftools_consensus`

VCFTools Consensus (Galaxy Toolshed version 0.1.11) takes a reference FASTA file and a VCF file as input and creates a consensus sequence incorporating the variants from the VCF file. We refer to this consensus sequence as a ‘genotype-specific’ reference genome.

Click on the *vcftools_consensus* tool.

Upload your (1) reference FASTA file and (2) the VCF files for both paren4 W1118 R129 mel_R129_F_rep1 mel_R129_F 3tal genotypes

1. Select **History** from the drop-down under Reference Genome Source

2. Select the **Reference genome** FASTA from the drop-down menu.

3. Select a VCF file **containing variants** for one of the parental genotypes from the drop-down menu.

4. Click on the “**Execute**” button to deploy the *vcftools_consensus* tool and create a genotype-specific reference genome for downstream use.

5. Repeat the above steps using the VCF file containing variants for the other parental genome.

Step 1 Outputs

This tool outputs one file:

- A single FASTA file containing a genotype specific reference where the difference in bases found in the input VCF file replace those in the input species reference file for a given parental genome.

More on VCFTools can be found here:

P. Danecek, *et al.* (2011). The variant call format and VCFtools. In *Bioinformatics*, 27 (15), pp. 2156--2158. [doi:10.1093/bioinformatics/btr330]

Check Alignment Design File Workflow

Workflow Overview

The Check Alignment Design File Workflow is a stand-alone workflow (*check_alignment_design_file.ga*) to verify that the user-supplied alignment design file has been

formatted correctly. See the BayesASE Data formats section for more assistance on how to properly format the [Alignment Design File](#). The Alignment Design File is used in downstream tools to link FASTQ names to file names and samples. Checks include ensuring that all headers are present, the columns are in the specified order, and that the required number of columns exist. The tool also examines whether there are any duplicate FASTQ file names present. Note that **fastq files must be unzipped at this stage**. We highly recommend that users look at the output from this tool before proceeding as there are many potential errors that may occur in this process.

The *Check Alignment Design File tool* can be run as a Galaxy workflow by importing and running `check_alignment_design_file.ga` (<https://github.com/McIntyre-Lab/BayesASE>).

Workflow Inputs

1. A user-supplied Alignment Design File

Workflow Outputs

1. An output file describing whether the design file adheres to BayesASE guidelines
2. An output file listing the duplicate FASTQ names present, if any

Description of individual Check Alignment Design File tool

Step 1: Component Script: Check Alignment Design File

Click on the ‘Check Alignment Design file’ tool

Upload your Alignment Design File

1. Select the **Alignment Design File** from the drop down menu



The screenshot shows the Galaxy tool interface for 'Check Alignment Design File for correct formatting and duplicate FASTQ names (Galaxy Version 0.1.0)'. It features a section for 'Alignment Design file' with a file upload button, a file selection button, and a text input field containing '310: alignment_df_one_compare.csv (as tabular)'. Below this is a descriptive text: 'Design file containing FASTQ file names, sampleIDs, etc. Refer to the help section at the bottom of this page for the format. [Required]'. At the bottom is a blue 'Execute' button with a checkmark icon.

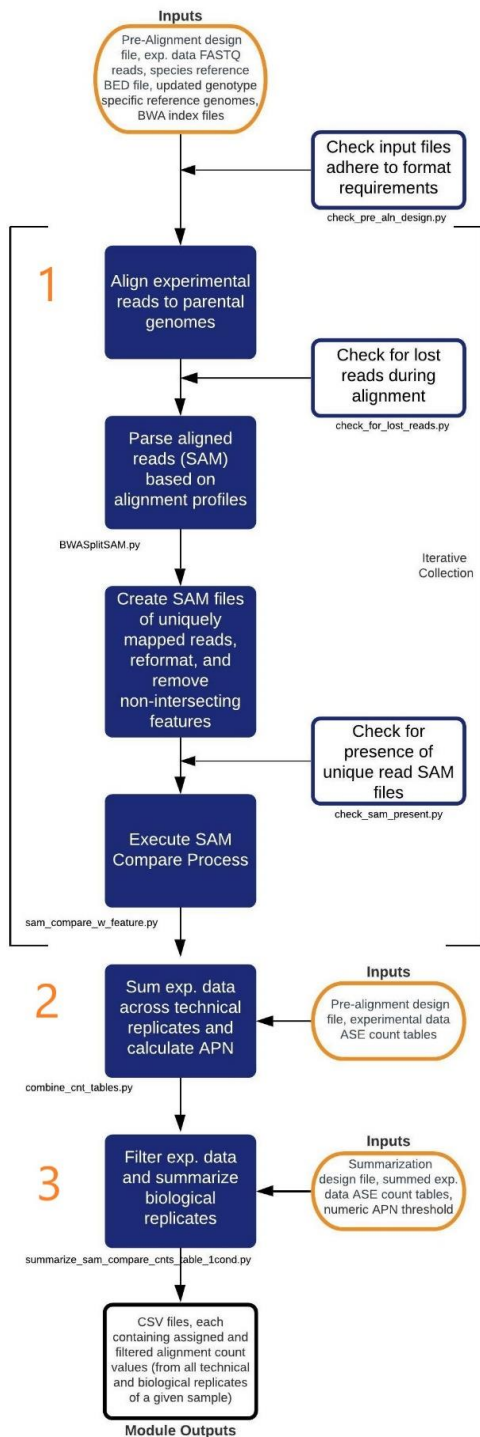
Tool Outputs

This tool outputs two files:

- A TSV file with information about whether the column names are correct and if there are any duplicated FASTQ file names. The file will indicate if:

- (1) the column names follow the specifications (“Column headers align with requirements”)
- (2) there are duplicate FASTQ file names (“No duplicate fqNames in your design file” or “There are duplicate fqNames in your design file!”)
- If any duplicate fqNames are identified in the design file, these rows are output to a TSV file for review. The example below is the output where 2 rows contain the same fqName:

G1	G2	sampleID	fqName	fqExtension	readLength	techRep
W1118	W55	W55_V_2	W55_V_2_3	.fastq	96	2
W1118	W55	W55_V_2	W55_V_2_3	.fastq	96	3



Align and SAM Compare Module

Module Overview

The *Align and SAM Compare Module* quantifies alignment counts for each input FASTQ file aligned to each genotype specific reference genome, and compares the resulting alignment files. The outcome is tabulated in an ASE Counts table composed of the reads that aligned preferentially to each genome, and those that aligned equally to both genomes.

Read counts for technical replicates (e.g. FASTQ data generated from the same library run on separate lanes) are summed. The average coverage per nucleotide (APN) is calculated for each genic feature. Note that the Bayesian model uses the raw count data but the APN can be used in filtering at a later step. A user-specified APN threshold value can be provided to determine whether a genic feature has enough read coverage for Bayesian analysis (**flag_analyze = 1 if at least 1 replicate has an APN value greater than the user-specified cutoff value**).

Module Workflows

The Align and Sam Compare Module is run as 2 consecutive workflows

- (1) align_and_count workflow.ga
 - (2) summarize_counts_workflow.ga
- (<https://github.com/McIntyre-Lab/BayesASE>)

Align and Count Workflow Overview

The align_and_count_workflow.ga aligns a collection of FASTQ files to each genotype-specific reference. Therefore, all FASTQ files in the collection should be the same genotype. The SAM

files are subsequently parsed and filtered to create SAM files containing uniquely mapping reads that overlap with genic features of interest defined in a user-supplied BED file. The genotype specific alignments for each genotype specific reference FASTQ file are compared and the resulting read counts compiled into an ASE Counts Table. This table contains read counts from the input FASTQ file aligned to each reference, by feature, in categories indicating whether the reads map preferentially to one of the two genotype specific genomes or equally well to both.

As the *align_and_count_workflow* uses single end alignments, all input FASTQ files are aligned using the *single* and not *paired* parameter. We note that this workflow leverages the BWA-MEM aligner and other tools available in the Galaxy Toolshed. Users who prefer other alignment strategies can modify the workflow. However, the *sam_compare* script tracks individual reads not pairs and an alteration to the alignment strategy will require a different summarization process as well.

Align and Count Workflow Checkpoints

Three checks are conducted in the Align and Count Workflow to identify errors. It is recommended that the user review these output files to identify any jobs that did not execute cleanly before continuing.

Align and Count Workflow Inputs

The *SAM Align and Compare Module* requires four input files:

- 1) A collection of FASTQ files containing F1 experimental technical replicates for a single genotype
- 2) A genotype specific reference for parental genome 1 (G1)
- 3) A genotype specific reference for parental genome 2 (G2)
- 4) A 4-column BED file containing the locations of genic features (the genic sequences the user is interested in) for the species; in this version of BayesASE this file must be identical for all genotypes. INDELS will by definition change the locations of genic features. In this version that functionality has not yet been enabled.

Align and Count Workflow Outputs

The *Align and Count Workflow* generates two output files:

- 1) An ASE Counts Table containing read counts from the input FASTQ file aligned to each genotype specific genome, by feature, in categories indicating whether the reads map preferentially to an individual parental genome or equally well to both.
- 2) An ASE Totals TSV file summarizing the read counts for the FASTQ file.

Description of individual Align and Sam Compare Workflow Tools

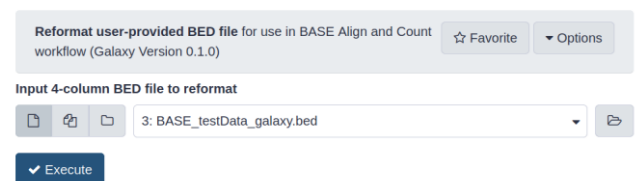
Step 1: Component Script: Reformat user-provided BED file

This tool reorders the columns in a user-specified 4-column BED file. The chrom (chromosome name) and name (name of the feature in the BED line) columns are switched such that the column containing the genic features of interest is column 1 and chrom is column 4. This allows for the downstream SAM alignments to be reformatted such that the RNAME column in the SAM file contains genic featureIDs in place of chromosome name.

Upload your 4-column BED file

Click on the 'Reformat Bedfile for BayesASE' tool

1. Select the **4-column BED file to reformat** from the drop down menu
2. Click on the **Execute** button to reformat the input file



Step 1 Outputs

This tool outputs one file:

- A 4-column BED file that has the names of the genic features in the first column and the associated chromosome name in the fourth column.

Step 2: Component Script: Map with BWA

The Map with BWA-MEM tool [Li H. and Durbin R, Galaxy Version 0.7.17.1], available in the Galaxy Toolshed, takes a user-supplied FASTQ file (or collection of FASTQ files) and maps the reads to a genotype specific reference genome (FASTA) specified by the user using the Burrows-Wheeler Aligner algorithm. The user can generate a genotype specific references using the *Genotype Specific Reference Module*. The sequence alignment data are saved in BAM binary file format. Note that each FASTQ file should be aligned to both genotype specific parental genomes. The Map with BWA-MEM tool must be run separately for each genotype specific reference, see step 7.

Click on the 'Map_with_BWA-MEM' tool

Upload your (1) FASTQ file (or collection of FASTQ files) and (2) both parental genotype specific fasta files

1. Select **Use a Genome from history and build index** from drop down menu under ‘Will you select a reference genome from your history or use a built-in index?’

2. Select the dataset to use as the **reference sequence** from the drop down menu.

3. Select **Auto**. Let BWA decide the best algorithm to use. If the genome is very large (>2 GB), the algorithm can be switched to BWA-ST.

4. Select **Single** reads. This is the required setting if using the *SAM Align and Compare workflow*.

5. Select the **fastq dataset** from the drop down menu. This can be a collection of FASTQ files or a single FASTQ file. All of the input FASTQ files will be aligned to the same input reference file.

6. Under set read groups information, select **Do not set**.

7. Select **Simple Illumina mode** from the drop down menu under Select Analysis mode. This can be modified if reads are generated from a different sequencing method such as PacBio.

8. Select **Execute** to generate alignments between the FASTQ file and the reference FASTA file

9. **Repeat** to align to the second parental genome

Map with BWA-MEM - map medium and long reads (> 100 bp) against reference genome (Galaxy Version 0.7.17.1)

Will you select a reference genome from your history or use a built-in index?

Use a genome from history and build index

Built-ins were indexed using default options. See 'Indexes' section of help below

Use the following dataset as the reference sequence

1: Consensus on data 1 and data 3

You can upload a FASTA sequence to the history and use it as reference

Algorithm for constructing the BWT index

Auto. Let BWA decide the best algorithm to use

(-a)

Single or Paired-end reads

Single

Select between paired and single end data

Select fastq dataset

14: FQ files M_V

This is a batch mode input field. Separate jobs will be triggered for each dataset selection.

Specify dataset with single reads

Set read groups information?

Do not set

Specifying read group information can greatly simplify your downstream analyses by allowing combining multiple datasets.

Select analysis mode

1.Simple Illumina mode

Execute

Step 2 Outputs

This tool creates one output for each input:

- A BAM file containing read alignments

More on BWA-MEM can be found in the reference below:

Li H. and Durbin R. (2010). Fast and accurate long-read alignment with Burrows-Wheeler Transform. Bioinformatics, Epub. [PMID: 20080505]

Step 3: Component Script: BAM to SAM

Convert the binary BAM alignment files to SAM format to be compatible with the subsequent tools in the *Align and Sam Compare Module*. We leverage the SamTools view command, BAM-to-SAM [Galaxy Version 2.0.1] that is available in the Galaxy to convert the file.

Click on the ‘BAM to SAM’ tool

Upload the (1) BAM file containing alignments of reads from a FASTQ file to a genotype specific reference

1. Select the **BAM file to convert** from drop down menu. This will be the output from the *Map with BWA-MEM tool* above.
2. Select **Exclude Header** from drop down under “*Header options*”
3. Select **Execute** to convert the specified BAM file to SAM file format
4. **Repeat** for the BAM file from the second parental genome



Step 3 Outputs

This tool creates one output for each input:

- A SAM file converted from the input BAM file

More can be read on SAMTools view here:

Li, H. (2010). Mathematical Notes on SAMtools Algorithms. Retrieved from: <http://lh3lh3.users.sourceforge.net/download/samtools.pdf>

Step 4: Component Script: BWASplitSAM

The BWASplitSAM tool subsets a SAM file based on values in the “FLAG” field. The flag values are used to parse the reads into categories according to how they align against the reference genome. The tool outputs a SAM file containing only uniquely mapping reads in addition to a summary TSV file containing read counts for each alignment category. Read type categories are:

mapped, unmapped, opposite, ambiguous, chimeric, and not_primary. Category types 'mapped' and 'opposite' together define the uniquely mapping reads.

Input SAM files must be generated from single-end alignments. The BWASplitSAM Tool works with SAM files generated by BWA-MEM. If using a different aligner, another tool may be substituted in place of BWASplitSAM to generate a SAM file of uniquely mapping reads.

The definition of each read alignment category:

- (1) **Mapped:** Reads that uniquely map on the forward strand
- (2) **Unmapped:** Reads that do not align
- (3) **Opposite:** Reads that uniquely map on reverse strand
- (4) **Ambiguous:** Reads that align to more than one location
- (5) **Chimeric:** Reads that align to distinct positions in the genome (e.g. non-linear alignments)
- (6) **Not_primary:** Reads with non primary alignments (e.g. secondary and supplementary)

Click on the 'BWASplitSAM' tool

Upload the (1) SAM file containing alignments between FASTQ reads and a genotype specific reference

1. Select the **SAM file** from the drop down menu. This will be the output from the 'BWASplitSAM' tool in Step 3 above.

2. Click '**Execute**'

3. **Repeat** for the SAM file aligned to the second parental genome



Step 4 Outputs

For each input SAM file, BWASplitSAM creates two output files:

- A SAM file containing all of the uniquely mapping reads.
- A summary TSV file containing read counts for each of the given alignment categories with the following headers:
 - name: Name of input FASTQ file
 - count_total_reads: Total number of reads in input FASTQ file
 - count_mapped_read_opposite_strand: Number of reads that mapped uniquely on reverse strand
 - count_unmapped_read: Number of reads that did not map

- count_mapped_read: Number of reads that mapped uniquely on forward stand
- count_ambiguous_read: Number of reads that mapped ambiguously
- count_chimeric_read: Number of reads that mapped non-linearly
- count_read_notprimary: Number of reads that mapped with secondary or supplementary alignments

An example summary TSV file:

name	count_total_reads	count_mapped_read_opposite_strand	count_unmapped_read	count_mapped_read	count_ambiguous_read	count_chimeric_read	count_notprimary
W55_M_1_1.fastq	1816	854	0	854	0	108	

Step 5: Component Script: SAM to BED

The SAM to BED tool converts the SAM file containing uniquely mapping reads from the BWASplitSAM step into a 3-column BED file containing the start and end positions of every uniquely mapping read. This BED file is required for entry into step 6, BEDTools Intersect Intervals, to identify uniquely mapping reads that overlap with genic features of interest.

The tool calculates the average read length from the input FASTQ file and creates the output BED file where 'chrom' is the chrom field from the SAM file, 'start' is the POS field from the SAM file and 'end' is the POS field value plus the average read length.

Click on the 'SAM to BED' tool

Upload the (1) FASTQ file used to create the SAM file and the (2) SAM file containing uniquely mapping reads

1. Select the **input FASTQ file** from the drop down menu used to generate the SAM file. The tool uses this file to calculate the average read length.

2. Select the **SAM file to be converted to a BED file** from the drop down menu

3. Click **Execute** to create a BED file from the input SAM file


4. **Repeat** for the SAM file aligned to the second parental genome

SAM to BED converts a SAM file to a BED file for use in BASE
 (Galaxy Version 0.1.0)

☆ Favorite
 Options

Select FASTQ files


14: FQ files M_V

 This is a batch mode input field. Separate jobs will be triggered for each dataset selection.

These will be used to calculate average read length

SAM files to convert

43: BWA Split Sam on collection 29: Uniquely Mapped Reads SAM

 This is a batch mode input field. Separate jobs will be triggered for each dataset selection.

Input SAM files to be converted to BED files

✓ Execute

Step 5 Outputs

This tool creates one output file for each input SAM file:

- A 3-column BED file where each row represents a uniquely mapping read.
 - (1) Chrom
 - (2) Start 5' start position of a read
 - (3) End 5' start position + read length

Step 6: Component Script: BEDtools Intersect Intervals

To identify uniquely mapping reads that overlap with genic features of interest, BayesASE employs the BedTools Intersect Intervals tool [Galaxy ver 2.29.2] available in the Galaxy ToolShed. This tool takes the converted BED file generated by the SAM to BED tool and the user specified BED file containing locations for genic features of interest and performs a left outer join to create a multi-column output file linking read position to genic features.

Click on the 'BEDtools_Intersect_Intervals' tool

Upload the (1) BED file created from the SAM file and the (2) user-supplied BED file containing genic features of interest

1. Select **File A to be intersected with file B** from the drop down menu. File A is the **converted BED file generated in Step 5**.

2. Select **One output file per 'input B' file** from the drop down menu under Combines or separate output files

3. Select **File B to be intersected with file A**. File B is the user supplied BED file containing genic features of interest. Do not use the reformatted BED file from step 1.

4. Under *Calculation based on strandedness?* Select **Overlaps on either strand**

bedtools Intersect intervals find overlapping intervals in various ways (Galaxy Version 2.29.2)

File A to intersect with B

71: SAM to BED on collection 43 and collection 14: SAM2BED

This is a batch mode input field. Separate jobs will be triggered for each dataset selection.

BAM/BED/bedGraph/GFF/VCF/EncodePeak format

Combined or separate output files

One output file per 'input B' file

File B to intersect with A

3: BASE_testData_galaxy.bed

BAM/BED/bedGraph/GFF/VCF/EncodePeak format (-b)

Calculation based on strandedness?

Overlaps on either strand

What should be written to the output file?

☐ Select/Unselect all

☒ Perform a "left outer join". That is, for each feature in A report each overlap with B. If no overlaps are found, rep

Treat split/spliced BAM or BED12 entries as distinct BED intervals when computing coverage.

Yes No

If set, the coverage will be calculated based the spliced intervals only. For BAM files, this inspects the CIGAR N opera inspects the BlockCount, BlockStarts, and BlockEnds fields (i.e., columns 10,11,12). If this option is not set, coverage and would include introns in the case of RNAseq data. (-split)

Required overlap

Default: 1bp

5. Under ‘What should be written to the output file’, select **Perform a “left outer join”**. That is, for each feature in A report each overlap with B. If no overlaps are found, report a NULL feature for B (-loj)

Report only those alignments that ****do not**** overlap with file(s) B

Yes No

6. Select **no** under ‘Treat split/spliced BAM or BED12 entries as distinct BED intervals when computing coverage.’

(-v)

Write the original A entry **_once_** if **_any_** overlaps found in B.

Yes No

Just report the fact ≥ 1 hit was found (-u)

For each entry in A, report the number of overlaps with B.

Yes No

7. Select **no** under the tab for ‘Report only those alignments that ****do not**** overlap with file(s) B’

Reports 0 for A entries that have no overlap with B (-c)

When using BAM input (-abam), write output as BED instead of BAM.

Yes No

8. For the tab, ‘For each entry in A, report the number of overlaps with B’; select **no**.

(-bed)

For coordinate sorted input file the more efficient sweeping algorithm is enabled.

Yes No

9. For the parameter ‘When using BAM input (-abam), write output as BED instead of BAM.’ select **no**.

(-sorted)

Print the header from the A file prior to results

Yes No

10. For the parameter ‘For coordinate sorted input file the more efficient sweeping algorithm is enabled’, select **no**.

(-header)

✓ Execute

11. For the parameter ‘Print the header from the A file prior to results’, select **no**.

12. Click **Execute** to find places of overlap between the input BED files

13. **Repeat** using the converted BED file for the second parental genome

Step 6 Outputs

The tool creates one output file per input file:

- A multi-column BED file where each row is a uniquely mapping read and the columns reveal the overlap between the BED files.
 - Column 1: chrom in the converted SAM to BED file
 - Column 2: start in the converted SAM to BED file
 - Column 3: end in the converted SAM to BED file
 - Column 4: chrom in the user supplied BED file
 - Column 5: start in the user supplied BED file
 - Column 6: end in the user supplied BED file
 - Column 7: feature name in the user supplied BED file

The following references provide more insight into BEDTools:
Bjoern A. Gruening (2014), [Galaxy wrapper](#)

Aaron R. Quinlan and Ira M. Hall (2010). BEDTools: a flexible suite of utilities for comparing genomic features. In *Bioinformatics*, 26 (6), pp. 841--842. [[doi:10.1093/bioinformatics/btq033](https://doi.org/10.1093/bioinformatics/btq033)]

Step 7: Component script: Remove Reads that that do not overlap with genic features

This tool removes reads from the multi-column BED file generated by BEDTools Intersect Intervals that do not overlap with genic features of interest (these are reported with a NULL B feature). The 'Remove Reads' tool will remove non-overlapping reads from the multi-column BED file.

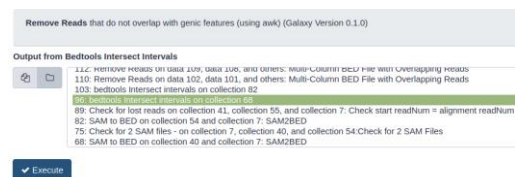
Click on the 'Remove Reads that do not overlap with genic features' tool

(1) Upload the multi-column BED file output from BEDTools Intersect Intervals

1. Select the **output from BEDtools Intersect Intervals tool** from the drop down menu.

2. Select **Execute** to remove reads that do not intersect with genic features

3. **Repeat** using output from BEDtools Intersect Intervals for the second parental genome



Step 7 Outputs

This tool outputs two files for each input file:

- A multicolumn BED file with the same columns (see Bedtools intersect intervals tool but with non-overlapping features (rows with NULL B features) removed.
- A TSV file indicating the number of reads dropped from the multi-column BED file
 - fqName: Name of the FASTQ file
 - total_number_rows: number of rows/reads in starting multi-column BED file
 - number_overlapping_rows: number of rows/reads remaining after omitting NULL B

fqName	number_overlapping_rows	total_number_rows
W55_V_3_2.fastq	1657	1665
W55_V_3_4.fastq	918	919
W55_V_3_3.fastq	1417	1420
W55_V_3_1.fastq	1573	1575
W55_V_2_4.fastq	1606	1609
W55_V_2_3.fastq	4305	4311
W55_V_2_2.fastq	3598	3600
W55_V_2_1.fastq	5829	5839
W55_V_1_4.fastq	2121	2128
W55_V_1_3.fastq	1880	1889
W55_V_1_2.fastq	2381	2383
W55_V_1_1.fastq	3736	3741
W55_M_3_4.fastq	2095	2101
W55_M_3_3.fastq	1952	1953
W55_M_3_2.fastq	2970	2976
W55_M_3_1.fastq	3134	3139
W55_M_2_4.fastq	4673	4682
W55_M_2_3.fastq	4167	4176
W55_M_2_2.fastq	7516	7533
W55_M_2_1.fastq	8015	8026
W55_M_1_4.fastq	1249	1250
W55_M_1_3.fastq	567	568
W55_M_1_2.fastq	1085	1085
W55_M_1_1.fastq	1706	1708

Step 8: Component Script: Create new SAM file

This tool reformats the SAM file containing uniquely mapping reads using the multi-column BED file created by the Remove Reads tool in the previous step. The resulting SAM file contains the name of genic features in the 'chrom' field.

Click on the 'Create_New_SAM_File' tool

Upload the (1) multi-column BED file generated using the Remove Reads Tool and the (2) SAM file containing uniquely mapping reads generated from the BWASplitSAM Tool

1. Choose the **multi-column BED file generated from the Remove Reads Tool** from the drop down menu. This BED file is generated in step 7, Remove Reads that do not overlap with genic features.

2. Choose the **SAM file to convert from chrom to feature**. This is the SAM file containing uniquely mapping reads and is generated by the BWASplitSAM Tool in Step 4.

3. Click **Execute** to create a new SAM file with alignments to genic features

Create new SAM file with features of interest in RNAME field
(Galaxy Version 0.1.0)

☆ Favorite ▼ Options

BED file generated from the Remove reads that do not overlap tool

764: Remove reads on collection 85: Multi-column BED file with overlappin... ▼

⚠

This is a batch mode input field. Separate jobs will be triggered for each dataset selection.

Input SAM file containing Uniquely Mapped Reads

43: BWA Split Sam on collection 29: Uniquely Mapped Reads SAM ▼

⚠

This is a batch mode input field. Separate jobs will be triggered for each dataset selection.

SAM file to convert chromosome names to feature IDs in RNAME field

✓ Execute

4. **Repeat** using the inputs generated from the second parental genome

Step 8 Output

This tool generates one output file:

- A SAM file that contains the name of the genic features in the 'chrom' field

Step 9: Component Script: Compare SAM files and create ASE Counts Tables

Steps 1 through 8 of the Align and Count Workflow described above are run twice, once for a FASTQ file aligned to the G1 genotype specific reference and again for the same FASTQ file aligned to the G2 genotype specific reference. This next step, the Compare SAM files tool in the Workflow, compares the read mapping across these two SAM files.

This tool creates an ASE Counts Table for each FASTQ file by comparing the new SAM files with genic features (created in the previous step by the Create new SAM file tool) that were aligned to both genotype specific references. Reads in each SAM file are processed and compared to determine whether a given read maps better to one genome or equally well to both. The results are also displayed by read count in a summary table.

Click on the '*Compare SAM files and create ASE Counts Tables*' tool

Upload (1) the starting FASTQ file, (2) the new converted SAM file aligned to G1, (3) the new converted SAM file aligned to G2, and (4) the Reformatted BED file

1. Choose the **name of the FASTQ file used to generate the 2 new converted SAM files** from the drop down menu

2. Choose the **new converted SAM file** with alignments to the **G1 reference**. This is the SAM file containing uniquely mapped reads to genotype specific reference G1 that overlap with genic features. The created in step 8 by the Create new SAM file tool.

3. Choose the **new converted SAM file** with alignments to the **G2 reference**. This is the SAM file containing uniquely mapped reads that overlap with genic features created in step 8.

4. Choose the **reformatted BED file containing features to assign reads to**. This is the reformatted BED file that lists the genic features in the first column.

5. Under Check Fastq IDs, check **yes**. This can be changed to 'no' to save computation time if the user is confident FASTQ IDs are correct.

Step 9 Outputs

The SAM Compare with Feature tool outputs two files:

- TSV file containing the ASE Counts Table. This table contains the number of read counts mapped to each feature listed in the input reformatted BED file. The columns depict which features map preferentially to one genome or to both, and whether it was an exact or inexact match. The file contains the following columns with the number of reads that mapped in the specified manner for each feature:
 - BOTH_EXACT: Number of reads that mapped uniquely to both genomes, and it was an exact match
 - BOTH_INEXACT_EQUAL: Number of reads that mapped equally to both genomes (ie did not preferentially to a certain parental genome), but it was an inexact match
 - {SAM_A/B}_ONLY_EXACT: Reads that mapped preferentially to only one of the two genomes and it was an exact match
 - SAM_A_EXACT_SAM_B_INEXACT: Reads that mapped exactly to genome 1 and inexactly to parental genome 2
 - SAM_B_EXACT_SAM_A_INEXACT: Reads that mapped exactly to genome 2 and inexactly to genome 1
 - {SAM_A/B}_ONLY_SINGLE_INEXACT: Reads that mapped preferentially to only one genome, and it was in an inexact manner
 - {SAM_A/B}_INEXACT_BETTER: Reads that mapped preferentially to their opposite parent, but the match is inexact

Compare SAM files and create ASE Counts Tables containing read counts for each feature in the genome (Galaxy Version 0.1.0)

FASTQ File

   14: FQ files M_V

 This is a batch mode input field. Separate jobs will be triggered for each file.

Select the FASTQ file used to generate the 2 SAM files [REQUIRED]

SAM file with G1 reference

   113: Create new SAM file on collection 99 and collection 100

 This is a batch mode input field. Separate jobs will be triggered for each file.

Select SAM file for G1 containing feature ID in RNAME field [REQUIRED]

SAM file with G2 reference

   120: Create new SAM file on collection 106 and collection 107

 This is a batch mode input field. Separate jobs will be triggered for each file.

Select SAM file for G2 containing feature ID in RNAME field [REQUIRED]

Reformatted BED file

   127: Reformat user-provided BED file on data 3

Select file containing features to assign reads to, with feature ID names in first column [REQUIRED]

Check Fastq IDs

☒ Yes
☐ No

Select No to skip checking SAM QNAME against the fastq sequence IDs. Save

 Execute

FEATURE_ID	BOTH_EXACT	BOTH_INEXACT_EQUAL	SAM_A_ONLY_EXACT	SAM_B_ONLY_EXACT	SAM_A_EXACT_SAM_B_INEXACT	SAM_B_EXACT_SAM_A_INEXACT	SAM_A_ONLY_SINGLE_INEXACT	SAM_B_ONLY_SINGLE_INEXACT	SAM_A_INEXACT_BETTER	SAM_B_INEXACT_BETTER
I1JG0196	39	106	0	0	14	1	0	1	7	8
CG0920	8	15	0	0	1	0	0	0	3	3
CG10932	5	3	0	0	0	0	0	0	0	1
Mapmodulin	33	52	0	0	0	1	0	0	2	0

- A text file called ‘ASE Totals’ that contains a summary of the counts by category.

Count totals:		
Count totals:		
1:	a_single_exact	0
2:	a_single_inexact	0
3:	a_multi_exact	0
4:	a_multi_inexact	0
5:	b_single_exact	0
6:	b_single_inexact	1
7:	b_multi_exact	0
8:	b_multi_inexact	0
9:	both_single_exact_same	0
10:	both_single_exact_diff	85
11:	both_single_inexact_same	0
12:	both_single_inexact_diff	200
13:	both_inexact_diff_equal	176
14:	both_inexact_diff_a_better	12
15:	both_inexact_diff_b_better	12
16:	both_multi_exact	3
17:	both_multi_inexact	4
18:	a_single_exact_b_single_inexact	15
19:	a_single_inexact_b_single_exact	2
20:	a_single_exact_b_multi_exact	0
21:	a_multi_exact_b_single_exact	2
22:	a_single_exact_b_multi_inexact	0
23:	a_multi_inexact_b_single_exact	0
24:	a_single_inexact_b_multi_exact	0
25:	a_multi_exact_b_single_inexact	0
26:	a_single_inexact_b_multi_inexact	0
27:	a_multi_inexact_b_single_inexact	0
28:	a_multi_exact_b_multi_inexact	0
29:	a_multi_inexact_b_multi_exact	1
30:	total_count	313

Align and Count Workflow Check Tools

Step 10: Component Script: Check for 2 SAM files

Each FASTQ file is aligned to the updated genotype specific reference for genome1 (G1) and genome2 (G2) - so every FASTQ file should generate 2 SAM files. This tool verifies that there are 2 SAM files for every 1 FASTQ file.

Upload the (1) FASTQ files and (2) the Unique SAM files created by the BWASplitSAM tool

1. Select the **FASTQ file** from the drop down menu. This must be the same FASTQ file that the SAM files were derived from.

2. Select the **Align Type**. If you are running the BayesASE Align and Count Workflow, select Single End

3. Select the **unique SAM file** aligned to the **first updated parental genome (G1)**. This is the output created by the BWASplitSAM tool in step 3.

4. Select the **unique SAM file** aligned to the **second updated parental genome (G2)**.

5. Click **Execute** to verify that 2 unique SAM files have been created per FASTQ file

Check for 2 SAM files - verify 2 SAM files are present for every 1 FASTQ file. (Galaxy Version 0.1.0)

Align Type

☒ Single end
☐ Paired end

Select whether SAM files were created from single end or paired end alignments

Unique SAM for G1

43: BWA Split Sam on collection 29: Uniquely Mapped Reads SAM

This is a batch mode input field. Separate jobs will be triggered for each dataset selection.

Select the SAM file [from BWASplitSAM] for genome 1 containing only uniquely mapped reads.

Unique SAM for G2

57: BWA Split Sam on collection 36: Uniquely Mapped Reads SAM

This is a batch mode input field. Separate jobs will be triggered for each dataset selection.

Select the SAM file [from BWASplitSAM] for genome 2 containing uniquely mapped reads.

FastQ file

14: FQ files M_V

This is a batch mode input field. Separate jobs will be triggered for each dataset selection.

Select the FastQ file used to generate alignments and create the input SAM files

Step 10 Outputs

This tool generates a TSV output file for each FASTQ file with the following columns:
fqName

message:

if 2 SAM files are found: “Have 2 SAM files - good!”,

if 2 SAM files are NOT found: “Do NOT have 2 SAM files - rerun alignments to each genotype specific genome!”

fqName	message
W55_V_1_1.fastq	Have 2 SAM files - good!

Step 11: Component Script: Check for lost reads

This tool checks that all reads in the starting FASTQ file are accounted for in the G1 and G2 SAM files after running the BWASplitSAM tool. Read counts in the input FASTQ file are compared to the ‘count_total_reads’ column in the summary of aligned reads TSV files generated by the BWASplitSam tool.

Upload the (1) Summary file for G1, the (2) Summary file for G2, and the (3) starting FASTQ file

1. Select the **BWASplitSAM Alignment Summary for G1** from the drop down menu. This is the TSV summary file output by the BWASplitSAM Tool for the G1 alignment.

2. Select **the BWASplitSAM Alignment Summary G2** from the drop down menu. This is the TSV summary file output by the BWASplitSAM Tool for the G2 alignment.

3. Select the **FASTQ file** used to generate both alignments.

4. Click **Execute**.

The screenshot shows the BWASplitSAM tool interface. At the top, there is a header bar with a warning message: "Check for lost reads verify starting FASTQ read number equals ending read number after alignments and parsing (Galaxy Version 0.1.0)". To the right of this bar are buttons for "Favorite" and "Options". Below the header, the tool is titled "BWASplitSAM Alignment Summary 1". There are three input fields, each with a file icon, a folder icon, and a dropdown menu. The first dropdown is set to "55: BWASplitSAM on collection 33: Summary of Aligned Reads". Below it is a note: "This is a batch mode input field. Separate jobs will be triggered for each dataset selection." The second dropdown is set to "41: BWASplitSAM on collection 26: Summary of Aligned Reads", with a similar note below it. The third dropdown is set to "7: FQ files", with a note below it: "This is a batch mode input field. Separate jobs will be triggered for each dataset selection." Below these fields, there is a label "Name of the FASTQ file" and a dropdown menu set to "7: FQ files". At the bottom, there is a label "Name of FQ used to generate the alignments selected above" and a blue button with a checkmark and the text "Execute".

Step 11 Output

This tool outputs a TSV file containing the starting read counts from the FASTQ file, the total read counts in each BWASplitSAM summary alignment file and a binary (0/1) indicator flag for whether the starting and ending read counts are the same.

- FqName
- start_read_num: The total number of reads in the FASTQ file
- readNum_G1: The total number of reads in the summary TSV file from BWASplitSAM for genome 1
- readNum_G2: The total number of reads in the summary TSV file from BWASplitSAM for genome 2
- flag_start_readNum_eq_readNum_G1: 0/1 indicator where “1” means that the number of reads in the FASTQ file matches the total read number in the G1 alignment file.
- flag_start_readNum_eq_readNum_G2: 0/1 indicator where “1” means that the number of reads in the FASTQ file matches the total read number in the G2 alignment file.

fqName	start_read_num	readNum_G1	readNum_G2	flag_start_readNum_eq_readNum_G1	flag_start_readNum_eq_readNum_G2
W55_V_1_1.fastq	3779	3742	3742	0	0

Step 12: Component Script: Check Sam Compare Output

This tool is designed to check that the number of reads into and out of the Compare SAM Files and Create ASE Counts Tables Tool are the same. The total of all reads mapped to each feature should be the sum of all unique reads mapped to that feature from the two alignment files. This implies that the total must be at least the number of reads mapping to one genome and no more than the sum of reads mapping to both genomes. Numbers of reads in the ASE Totals file outside of this range indicate that the Sam Compare with Feature Tool should be rerun.

Upload the (1) Summary File output from the Remove reads that do not overlap with genic feature tool for both G1 and G2 the (2) starting FASTQ file, and the (3) The ASE Totals Table

1. Select the **Summary file with removed reads for Genome 1** from the drop down menu. This file is generated by the Remove reads tool in step 7.
2. Select the **Summary file with removed reads for Genome 2** from the drop down menu. .
3. Select the **FASTQ file** used to generate the SAM alignments from the drop down menu
4. Select the **ASE Totals file**. This is the summary file output by the SAM Compare with Feature tool.
5. Click **Execute**.

Check Sam Compare Output to ensure no reads were lost after workflow has finished (Galaxy Version 0.1.0)

Output summary file from drop tool for G1

381: Remove Reads on data 354, data 353, and others: Number Rows Left

Select the summary file containing the total read counts after dropping [Required]

Output summary file from drop tool for G2

383: Remove Reads on data 379, data 378, and others: Number Rows Left

Select the summary file containing the total read counts after dropping [Required]

FASTQ file

28: Full test data

This is a batch mode input field. Separate jobs will be triggered for each d.

Select the FASTQ file used to generate the SAM alignments [Required]

ASE total file

483: SAM Compare with Feature on collection 28, collection 432, and coller

This is a batch mode input field. Separate jobs will be triggered for each d.

Select the ASE Totals tables containing the read counts generated by the SAM Compare tool [Re

✓ Execute

Step 12 Output

This tool generates a summary TSV file with the following columns:

- fqName
- min_uniq_g1_uniq_g2: The number of uniquely mapping reads in either the G1 or G2 alignments, whichever is smaller.

- `sum_uniq_g1_uniq_g2`: The sum of the uniquely mapping reads in the G1 and G2 alignments
- `total_counts_ase_table`: The total number of reads evaluated in the SAM Compare with Feature Tool. Calculated from the ASE Totals file.
- `flag_readnum_in_range`: A binary (0/1) indicator where “1” indicates that the number of reads in the ASE Totals file is greater than the value in the `min_uniq_g1_uniq_g2` column and less than the value in the `sum_uniq_g1_uniq_g2` column.

<code>fqName</code>	<code>min_uniq_g1_uniq_g2</code>	<code>sum_uniq_g1_uniq_g2</code>	<code>total_counts_ase_table</code>	<code>flag_readnum_in_range</code>
W55_V_1_1.fastq	1706	3412	1939	1

Summarize Counts Workflow Overview

The Summarize Counts Workflow is comprised of three tools, the Combine ASE Count Tables Tool, the Summarize and Filter ASE Count Tables, and the Reformat Sample Design File. Inputs into this workflow are the collection of ASE Count tables generated from the Align and Count Workflow, the user supplied BED file containing genic features of interest, and the Alignment Design File. For each genic feature, ASE counts are summed across technical replicates and the average reads per nucleotide (APN) is calculated. Features are then flagged using a user specified APN value to identify features with enough read coverage for analysis. Note that at this stage the flag is per sample and decisions about filtering are downstream as a combination of these flags. For example, for features with 0 counts in all samples these are not analyzable.

The Summarize Counts Workflow requires the user-supplied [Alignment Design File](#) to identify technical replicates and sample replicates. We recommend that the user run the Check Alignment Design File Tool to check their design file.

Summarize Counts Workflow Inputs

1. Alignment Design File
2. BED File containing genic features of interest
3. Collection of ASE Counts Tables generated from the Align and Sam Compare Workflow

Summarize Counts Workflow Outputs

This workflow outputs two files:

1. A collection of summarized and filtered ASE Counts Tables
2. A [ample Design file](#)

Description of individual Summarize Counts Workflow Tools

Step 1: Component Script: Combine ASE Count Tables

The Combine ASE Counts Tables tool sums the input ASE Counts tables across the technical replicates specified in the Alignment Design File. Technical replicates within the same biological replicate are combined for each compare, and the average read per nucleotide (APN) value is calculated for each feature. The tool outputs a single ASE counts file for each sample replicate with APN values. The tool also outputs a [Sample Design File](#) that is required for the Summarize and Filter ASE Count Tables tool that follows.

$APN_{total} = ([\text{total alignment count}] \times [\text{readLength}]) / (\text{genic feature length})$

$APN_{both} = ([\text{reads mapping equally to both parental genomes}] \times (\text{readlength})) / (\text{genic feature length})$

Click on the ‘*Combine ASE Count Tables*’ tool

Upload the following files: (1) Alignment Design File, (2) User-Supplied BED file and (3) Collection of ASE Counts Tables

1. Select the **Alignment Design File** from the drop down menu. The design file must be formatted in the correct order and configuration. To check whether the user-defined design file is in the BayesASE proper layout, use the [Check Alignment Design File Tool](#). Make sure that the design file is sorted according to biological replicate.

2. Select the **BED file** from the drop down. This file is the [original user-supplied BED file](#) containing the locations of the genic features. The chromosome name should be in the first column; do not use the reformatted BED file created in the Align and Counts workflow where the name of the features of interest are first.

3. Select the collection of **ASE Count Tables** from the drop down menu. This collection is generated by the Align and Count Workflow.

Combine ASE Count Tables: sum technical replicates (Galaxy Version 0.1.0)

Design File

Select the Alignment Design File. The design file must be sorted by biological replicate (sampleID)

Bed File

Select the input BED file. Do not use the reformatted BED file created in the Align and Count workflow

ASE Count Tables

Select collection containing ASE count tables

Start

2

Enter start point in design file [OPTIONAL]

End

24

Enter end point in design file [OPTIONAL]

Dataset Type

☒ Real data
☐ Simulated data

Select whether the dataset contains real data or simulated reads

4. [OPTIONAL]: Enter the **Start point** of the Alignment Design File. Enter where the biological replicates of interest start in the design file. Use if only doing analysis on a certain subset within the input design file. Leave blank to run the Tool on all rows of the Pre-Alignment Design File.
5. [OPTIONAL]: Enter the **End point** of the Alignment Design File. Enter the point in design file where user wants to exclude further FASTQ files from data analysis. Use only if analyzing a subset of FASTQ files within the design file. Leave blank to run the Tool on all rows of the Pre-Alignment Design File.
6. Select the **Dataset type** indicating whether the data is composed of real or simulated data.
7. Click **Execute** to sum the ASE Count Tables for each technical replicate within a given biological replicate and calculate APN values.

Step 1 Outputs

Combine Counts Tables Tool outputs two files:

- A TSV file for each sample replicate containing the summed ASE Counts tables and the APN values for the uniquely mapping reads per feature. It contains the following columns:
 - Feature_ID
 - APN_both: Calculated APN values for reads that mapped equally to both parental genomes
 - APN_total_reads: Calculated APN values for all uniquely mapped reads

The following columns are the same as those mentioned in the output ASE Counts Table created in the *Align and Count workflow* by the [Compare SAM Files and Create ASE Counts Tables Tool](#)

- BOTH_EXACT
- BOTH_INEXACT_EQUAL
- {SAM_A/B}_ONLY_EXACT
- SAM_A_EXACT_SAM_B_INEXACT
- SAM_B_EXACT_SAM_A_INEXACT
- {SAM_A/B}_ONLY_SINGLE_INEXACT
- {SAM_A/B}_INEXACT_BETTER

FEATURE_ID	APN_both	APN_total_reads	BOTH_EXACT	BOTH_INEXACT_EQUAL	SAM_A_EXACT_SAM_B_INEXACT	SAM_A_INEXACT_BETTER
I(1)G0196	3.106902274724202	3.842624231182271	253	410	45	35
CG8920	2.1056970410716915	2.642720447519505	84	65	16	18
CG10932	7.305050505050505	8.468686868686868	61	52	2	2
Mapmodulin	4.226828850436868	4.687339386671235	115	142	14	10

SAM_A_ONLY_EXACT	SAM_A_ONLY_SINGLE_INEXACT	SAM_B_EXACT_SAM_A_INEXACT	SAM_B_INEXACT_BETTER	SAM_B_ONLY_EXACT	SAM_B_ONLY_SINGLE_INEXACT
0	1	44	32	0	0
0	0	3	1	0	0
0	0	4	10	0	0
0	0	2	2	0	0

- A sample design file which contains the replicate (sampleID) and names of the G1 and G2 genomes.

Step 2: Component Script: Summarize and Filter ASE Count Tables

The Summarize and Filter ASE Count Tables tool filters data based on whether or not they meet the requirements for input into the Bayesian module to test for allelic specific expression. Using the sample design file, it creates one file with all replicates for a given compare. The data are then filtered, identifying features that don't meet a user-defined APN (average number of reads per nucleotide) threshold.

The default APN threshold value is 5. Features that do not meet this cutoff point are considered to have insufficient information needed to estimate model parameters. For each feature, if at least 1 replicate in a compare has an APN value greater than the user-specified cutoff value then the binary indicator flag in the output (flag_analyze) will be equal to 1, else equal to 0.

Click on the '*Summarize and Filter ASE Count Tables*' Tool

Upload the following files: (1) Sample Design File

1. Select the **Sample Design File** from the drop down.
This is the design file created from the Combine Counts table script. See page ([link](#)) in the BayesASE Data Formats section for more details.
2. Select the summed **ASE Count Table collection** (output by Combine Counts, step 1).
3. Enter the name of the column header designated for **G1** in the design file.
4. Enter the name of the column for **G2** in the design file.
5. Enter the name of the column for the **sample ID** in the design file.
6. Enter the name of the column for the **compare(s)** in the design file.
7. Enter the desired **APN threshold value**. BayesASE will use this to determine which features of compares should be included in further analysis for allelic specific expression.
8. Click **Execute** to filter the features in the summed ASE Counts table for those with enough information for Bayesian analysis.

Summarize and Filter ASE Count Tables based on user-defined APN threshold and read coverage

Sample Design file

564: Combine ASE Count Tables: on data 530, data 528, and others: Sample

Select the Sample Design File--this may be created with the Combine Counts Tables tool

Collection of Combined ASE Count Tables

563: Combine ASE Count Tables: on data 530, data 528, and others: Combined and Summed ASE

Select the collection containing combined/summed ASE count tables

Genome 1 (G1)

G1

Enter the header name of genome 1 (eg G1) column in your design file.

Genome 2 (G2)

G2

Enter the header name of genome 2 (eg G2) column in your design file.

Sample ID Column

sampleID

Enter the name of the column in design file containing sampleIDs

Compare Column

compare

Enter the header name of the column in design file containing compare names

APN (Average Reads per Nucleotide) threshold

5

Enter desired APN threshold for flagging a feature as 'expressed' in each replicate. The default setting is 5.

Step 2 Outputs

This tool outputs two files:

- For each compare, one summary TSV file is created containing the flagged indicators recording whether or not a genic feature meets the specified APN threshold. In the below column headers {compare} refers to the 'compare' header that is in the Sample design file (ie W1118_F or W1118_M). The number of columns generated is dependent on the number of replicates.
 - FEATURE_ID
 - G1
 - G2
 - {compare}_num_reps: The number of replicates for the indicated compare.
 - counts_{compare}_g1_total_{replicate_number}: Total number of unique reads from a given replicate that mapped to updated parental genome 1

- `counts_{compare}_g2_total_{replicate_number}`: Total number of unique reads from a given replicate that mapped to updated parental genome 2
- `{compare}_flag_analyze`: 0/1 binary indicator variable where a “1” means that at least one replicate in the compare has an APN greater than the user-specified cutoff value.
- `counts_{compare}_both_total_{replicate_number}`: Total number of unique reads from a given replicate that mapped equally well to both updated parental genomes
- `{compare}_flag_apn_{replicate_number}`: 0/1 flag where a “1” indicates that the APN value for a given feature is above the user-defined APN threshold
- `{compare}_total_reads_APN_{replicate_number}`: The calculated APN value for the total number of unique reads that mapped to a given feature
- `{compare}_both_APN_{replicate_number}`: The calculated APN value for the unique reads that mapped equally to both parental genomes for a given feature

FEATURE_ID	g1	g2	W55_M_flag_analyze	W55_M_num_reps	counts_W55_M_g1_total_rep1	counts_W55_M_g2_total_rep1	counts_W55_M_both_total_rep1	W55_M_flag_apn_rep1	W55_M_APN_total_reads_rep1	W55_M_APN_both_rep1	counts_W55_M_g1_total_rep2
IG100196	W1118	W55	1	3	24	36	378	0	2.0431554204822805	1.7619837938104068	251
CG8920	W1118	W55	1	3	5	7	57	0	0.975121448549976	0.805351096717209	26
CG10932	W1118	W55	1	3	4	0	25	0	1.8747474747474748	1.616161616161616	35
Mapmodulin	W1118	W55	1	3	3	16	220	0	3.930786362857632	3.618297070412883	13

counts_W55_M_g2_total_rep2	counts_W55_M_both_total_rep2	W55_M_flag_apn_rep2	W55_M_APN_total_reads_rep2	W55_M_APN_both_rep2	counts_W55_M_g1_total_rep3	counts_W55_M_g2_total_rep3	counts_W55_M_both_total_rep3	W55_M_flag_apn_rep3	W55_M_APN_total_reads_rep3	W55_M_APN_both_rep3
352	2489	1	14.48950502762388	11.063770279771549	87	132	1125	1	6.298154837449996	5.27189300097628
51	464	1	7.645517444428087	6.55733849625939	14	21	140	0	2.4732341366412484	1.9785077869129987
12	238	1	16.424342434243438	15.388585858585858	10	6	73	1	5.753535353535353	4.719191919191919
75	676	1	15.85471989354636	14.407401062189482	7	45	457	1	8.37142367654817	7.516189823539491

Step 3: Component Script: Reformat Sample design file

The Reformat Sample Design File tool takes the Sample Design File generated by the Combine ASE Counts Tables tool and modifies it to create the [Priors Design File](#).

The input Sample Design File should contain the following columns, in order:

- 1) G1 - name of updated parental genome 1
- 2) G2 - name of updated parental genome 2
- 3) sampleID - sample identifier (no spaces in name). Consists of combination of compare + replicate number
- 4) compare: compares to be tested for allelic imbalance

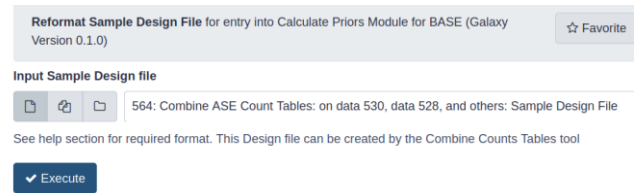
Compare conditions should be in the following format: `{genotype}_{condition}`

e.g. W1118_V

Click on the ‘*Reformat Sample Design File*’ tool

Upload the Sample Design file

1. Select the **Sample Design File** from the drop down menu. This file can be created using the Combine ASE Counts Tables tool.



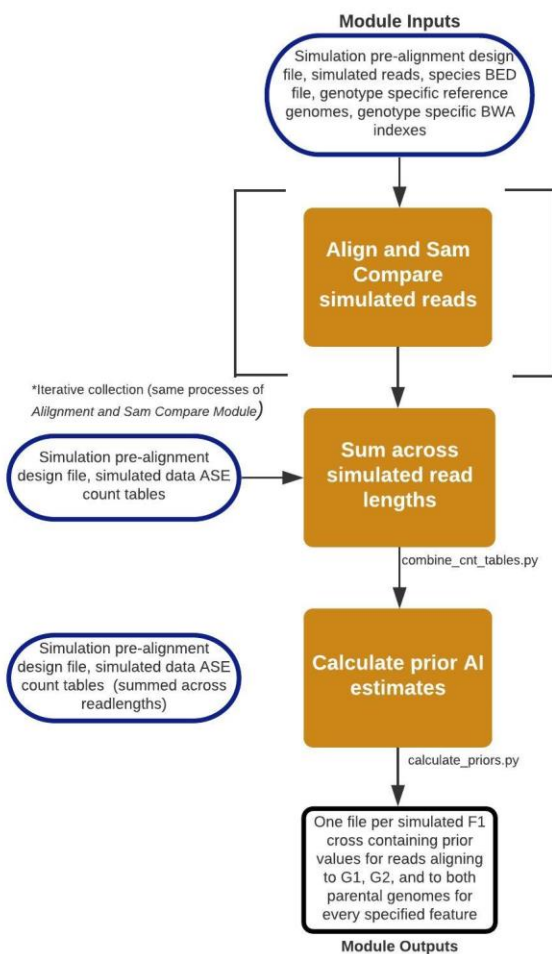
Step 3 Outputs

This tool outputs one file:

- A Priors Design File in TSV format

```
G1      G2      compare
W1118 W55    W55_M
W1118 W55    W55_V
```

Calculate Priors Module



Calculate Priors Module Overview

The *Calculate Priors Module* is comprised of one tool: Calculate Priors. This module calculates the prior probability estimates that a read will align preferentially to one of the genotype specific genomes or equally to both for a given feature.

Calculate Priors Module Considerations

This module contains one tool within a standalone workflow, The `prior_calculations_workflow.ga`, workflow. The input into this workflow is expected to have the same format as that output by the Summarize Counts workflow. Users can run simulated reads or DNA reads through the Align and Count and the Summarize Counts Workflows to generate data in the correct format for input into the Calculate Priors Module.

Calculate Priors Module Inputs

The Prior Calculations module requires two input files:

- 1) A collection of summarized and filtered ASE Counts Tables. This collection can be generated by the Align and Count Workflow followed by the Summarize Counts Workflow.
- 2) A [Priors design file](#) This file can be generated from the Reformat Sample design file tool.

Calculate Priors Module Outputs

This module creates one output file:

- 1) A dataset containing the calculated priors estimates for reads that mapped equally to both parents and those that mapped preferentially to one, for a given compare.

Step 1: Component Script: Calculate Priors from ASE count tables

The probability is calculated in the following manner, where `prior_both` refers to the chance that reads will align equally to both parental genomes, and `prior_G1/G2` is the chance that reads will align preferentially to G1 or G2.

Click on the ‘*Calculate Priors from ASE Counts Tables tool*’

Upload the (1) Priors Design File and the (2) Data to be used for calculating the priors (in the same format as that for the Summarized and Filtered ASE Counts Tables)

1. Select the **Priors Design File** from the drop down menu. This is the Design File created by the Reformat Sample Design File tool.

2. Select the data to be used for calculating priors from the drop down. Data must be in the **Summarized and Filtered ASE Counts Table** format.

Step 1 Outputs

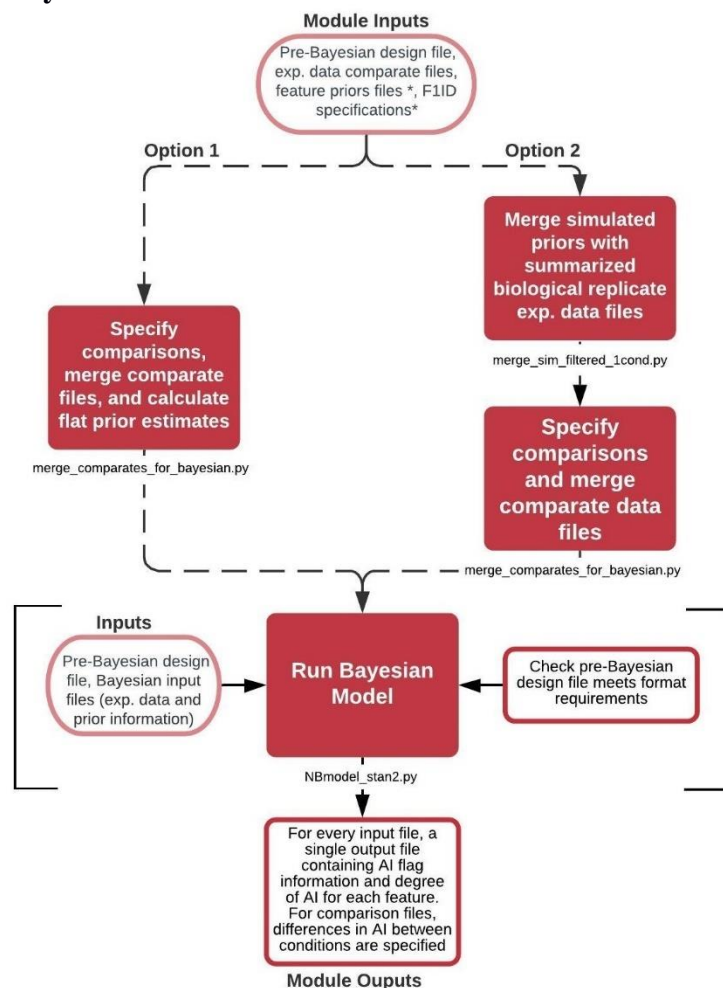
This tool creates one output file for each compare:

- One TSV file containing the calculated prior probabilities for reads that align preferentially to G1, G2, and those that align equally to both, for each feature within a given compare.
 - Unique Feature ID
 - `prior_{compare}_both`: prior probability estimate for the total amount of reads that mapped equally to both updated parental genomes
 - `prior_{compare}_g1`: prior probability estimate for the total amount of reads that mapped to updated parental genome 1
 - `prior_{compare}_g2`: prior probability estimate for the total amount of reads that mapped to updated parental genome 2

An example prior output file is shown below:

FEATURE_ID	prior_W55_M_both	prior_W55_M_g1	prior_W55_M_g2
I(1)G0196	0.8189655172413793	0.07430213464696224	0.10673234811165845
CG8920	0.8420382165605096	0.05732484076433121	0.10063694267515924
CG10932	0.8337468982630273	0.12158808933002481	0.04466501240694789
Mapmodulin	0.907126168224299	0.013434579439252336	0.0794392523364486

Bayesian Model Module



Bayesian Model Module Overview

In the Bayesian Model Module, prior AI (allelic imbalance) estimates per genic feature are incorporated with compare experimental data files before they are analyzed by the Novelo et. al 2014 model. The Bayesian model formally tests AI for each genic feature level while accounting for genomic context and bias in read mapping. Required inputs of prior estimates can be calculated utilizing the *Prior Calculation Module*, an external program,. These “prior” values are alignment proportions to each genome-specific reference genome under the null hypothesis that AI is not influenced by the particular conditions that are being tested. Compare experimental data files are the output of prior BASE modules. If comparisons are specified in the C design file, compare files are merged and become input to the Novelo et. al. model.

Bayesian Model Module Workflows

The Bayesian Model Module contains two workflows:

- 1) *Merge Priors Workflow*- Merges experimental data compare files within comparison specifications.
- 2) *Run Bayesian Workflow*

Merge Priors Workflow

Merge Priors Workflow Overview

The Merge Priors workflow (`merge_priors_tow_comparates_workflow.ga`) takes the output files created by the Summarize Counts workflow and the Calculate Priors workflow and merges them by feature into a single file for each compare as specified in the user-supplied Compare Design File. Individual compare files are subsequently combined merged into one file, creating a file that includes all compares specifications defined by the user.

Merge Priors Workflow Inputs

There are four required inputs for the Merge Priors Workflow.

- 1) The [Priors Design File](#)
- 2) Collection of summarized and filtered ASE Counts Tables for each compare
- 3) Collection of Prior Probability files for each compare
- 4) A user-supplied [Compare Design file](#)

Merge Priors Workflow Outputs

The workflow creates one output file:

- 1) A TSV where a summarized and filtered ASE Counts table and the Prior Probability file for a given compare have been merged into a single file.

Step 1: Component Script: Check Compare Design File

The Check Compare Design file tool verifies that the user-supplied Compare design file adheres to BASE guidelines. This design file is required for both the Merge Priors and Run Bayesian workflows. The necessary headers are case-sensitive, so the input design file must be spelled the same way.

If the design file has mislabeled or missing headers, an error will be raised.

The input Compare Design File must contain the following columns in order. More information on the format of the different design files and what they are used for can be found [here](#).

1. Compare_1: Compare 1 identifier (ex. W55_M)
2. Compare_2: Compare 2 identifier (ex. W55_V)
3. CompID: An unique identifier that specifies the comparison (ex. W55_M_V)

Click on the ‘Check Compare Design File’ tool

Upload the (1) Compare Design file

1. Select the **Compare Design File**
2. Select the **number of compares** being tested for allelic imbalance

This tool outputs one file:

- A file telling the user whether the design file aligns with BASE requirements.
 - (1) the column names follow the specifications (“The columns are labeled correctly and are in correct order” or “Error: Format of design file does not align with BASE requirements”)

Design_file	message
compare_df_two_conditions_BASE.tsv	The columns are labeled correctly and are in correct order

Step 2: Component Script: Merge priors to compare

The Merge Priors to Compare tool merges the file containing prior probability values to the summarized ASE count table for each compare specified in the Compare Design File.

The files are merged on the FEATURE_ID column. Therefore, FEATURE_IDs in the Summarized and Filtered ASE Counts Tables must be identical to those in the Prior file and in the same order.

This tool requires the Priors Design file, containing the comparates of interest for both samples. This design file can be created by using the Reformat Sample Design File tool found in the *Summarize Counts workflow*.

Click on the ‘Merge Priors to Compare’ tool

Upload the (1) Priors Design File, the (2) Collection of Summarized Counts tables and the (3) Collection of Priors Files

1. Select the **Priors Design File** from the drop down menu. The Priors Design File is created by the Reformat Sample Design file tool.

2. Select the **collection of Summarized and Filtered ASE Count tables**.

3. Select the **collection containing the Prior Calculations** files for each compare

Merge Priors to Compare merges filtered/summarized ASE Counts tables to file with calculated prior values for a single compare (Galaxy Version 0.1.0)

Priors Design file

565: Reformat Sample Design File on data 564: Prior Design File

Enter the Priors Design file output from Reformat Sample Design file; formatting is shown below

Summarized and Filtered ASE Count Table Collection

572: Summarize and Filter ASE Count Tables on data 571, data 570, and others: Summarized and Filtered ASE Count Tables

Select the collection containing input filtered ASE count tables per compare

Calculated Priors Collection

575: Calculate Priors Probability Estimates on data 574, data 573, and data 565: Calculate Prior Probability Estimates

Select the collection containing the Prior Probability Estimates for a given compare

Execute

Step 2 Outputs

This tool output one file per compare:

- A TSV file containing the Prior Calculation file merged to the Summarize ASE Count Tables file by FEATURE_ID. The following columns headers contain the same information as was specified in those datasets ([link to Priors headers](#), [link to Summarize ASE Counts headers](#)).
 - Unique Feature ID
 - prior_{compare_name}_both
 - prior_{compare_name}_{G1/G2}
 - {compare}_flag_analyze
 - {compare}_num_reps
 - {compare}_g1_total_{biological_replicate_number}

- {compare}_g2_total_{biological_replicate_number}
- {compare}_both_total_{biological_replicate_number}
- {compare}_flag_apn_{biological_replicate_number}
- {compare}_total_reads_APN_{biological_replicate_number}
- {compare}_both_APN_{biological_replicate_number}

FEATURE_ID	g1	g2	W55_M_flag_analyze	W55_M_num_reps	W55_M_g1_total_rep1	W55_M_g2_total_rep1	W55_M_both_total_rep1	W55_M_flag_apn_rep1
I(1)G0196	W1118	W55	0	2	21	18	157	0
CG8920	W1118	W55	0	2	1	11	37	0
CG10932	W1118	W55	0	2	4	2	19	0
Mapmodulin	W1118	W55	0	2	1	8	58	0

W55_M_APN_total_reads_rep1	W55_M_APN_both_rep1	W55_M_g1_total_rep2	W55_M_g2_total_rep2	W55_M_both_total_rep2	W55_M_flag_apn_rep2	W55_M_APN_total_reads_rep2	W55_M_APN_both_rep2
0.9184809137947868	0.7357219564580689	0	0	9	0	0.04217514400078102	0.04217514400078102
0.6924775504195495	0.5228912115412925	0	0	2	0	0.02826438981304284	0.02826438981304284
1.616161616161616	1.2282828282828282	0	0	0	0	0.0	0.0
1.1019359259893782	0.9539146821997602	0	0	4	0	0.06578721946205242	0.06578721946205242

Step 3a: Component Script: Merge Compare Datasets and Generate Headers

The Merge Compares and Generate Headers tool merges compare datasets containing the priors (generated by the Merge Priors tool) as specified in the Compare Design File. New headers are created by mapping the user-specified compare names to a general variable. Note – for testing AI for one compare, use the corresponding “Generate Headers for One Condition” tool instead, which generates the new headers needed for the Bayesian model on a single compare.

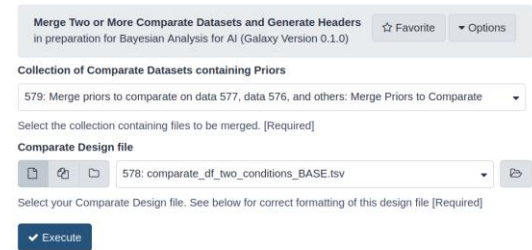
This tool requires a Compare Design File that explicitly lists the comparison the user wants to do.

Click on the ‘Merge Compares Datasets and Generate Headers’ Tool

Upload the (1) Collection created by the Merge Priors to Compare tool and the (2) User-supplied Compare Design file

1. Select the collection output from **Merge Priors to Compare**

2. Select the **Compare Design file**. This must be a TSV file



The screenshot shows the Galaxy tool interface for 'Merge Two or More Compare Datasets and Generate Headers'. The tool is used for Bayesian Analysis for AI (Galaxy Version 0.1.0). It features a dropdown menu for 'Collection of Compare Datasets containing Priors' with the selected value '579: Merge priors to compare on data 577, data 576, and others: Merge Priors to Compare'. Below this is a section for 'Compare Design file' with a dropdown menu showing '578: compare_df_two_conditions_BASE.tsv'. A blue 'Execute' button is at the bottom.

Step 3 Outputs

-The tool generates one file:

- A TSV file containing the compares listed in the Compare Design File merged into a single file. The number of columns is contingent upon the number of replicates for each compare. The headers are:
 - Unique Feature ID
 - `prior_{compare}_both`: The calculated probability estimate that a feature will map equally to both parents, for a given compare (ie C1, C2, C3, C_n)
 - `prior_{compare}_{g1/g2}`: The calculated probability estimate that a feature will map preferentially to one of the updated parental genomes for a given compare
 - `{compare}_flag_analyze`: 0/1 binary indicator variable where a “1” means that at least one replicate in the compare has an APN greater than the user-specified cutoff value.
 - `{compare}_num_reps`: Number of replicates present for a given compare
 - `count_{compare}_{g1/g2}_total_{replicate number}`: Total number of uniquely mapped reads aligning preferentially to one of the updated parental genomes for a feature within a given compare
 - `count_{compare}_both_total_{replicate number}`: Total number of uniquely mapped reads aligning equally to both updated parental genomes for a feature within a given compare
 - `{compare}_flag_apn_{replicate number}`: 0/1 where “1” indicates that a feature has a APN greater than or equal to the user-defined APN threshold
 - `{compare}_APN_total_reads_{replicate number}`: Calculated APN value for all unique reads for a replicate within a compare

- {compare}_APN_both_{replicate number}: Calculated APN values for unique reads mapping equally well to both updated parental genomes for a replicate within a compare

An example output file (split for visibility):

FEATURE_ID	prior_c1_both	prior_c1_g1	prior_c1_g2	c1_flag_analyze	c1_num_reps	c1_g1_total_rep1	c1_g2_total_rep1	c1_both_total_rep1	c1_flag_apn_rep1	c1_APN_total_reads_rep1	c1_APN_both_rep1	c1_g1_total_rep2
K(1)G0196	0.8097560975609757	0.1024390243902439	0.08780487804878047	0	2	21	18	157	0	0.9184809137947868	0.7357219564580689	0
CG8920	0.7647058823529411	0.0196078431372549	0.215686274509804	0	2	1	11	37	0	0.6924775504195495	0.5228912115412925	0
CG10932	0.76	0.16	0.08	0	2	4	2	19	0	1.616161616161616	1.2282828282828282	0
Mapmodulin	0.8732394366197183	0.014084507042253518	0.11267605633802814	0	2	1	8	58	0	1.1013959259893782	0.9539146821997602	0

c1_g2_total_rep2	c1_both_total_rep2	c1_flag_apn_rep2	c1_APN_total_reads_rep2	c1_APN_both_rep2	prior_c2_both	prior_c2_g1	prior_c2_g2	c2_flag_analyze	c2_num_reps	c2_g1_total_rep1	c2_g2_total_rep1	c2_both_total_rep1
0	9	0	0.04217514400078102	0.04217514400078102	0.8033273915626857	0.0885323826500297	0.1081402257872846	0	2	80	102	742
0	2	0	0.02826438981304284	0.02826438981304284	0.8154269972451791	0.07988980716253444	0.1046831955922865	0	2	14	21	158
0	0	0	0.0	0.0	0.8226600985221675	0.11822660098522167	0.059113300492610835	1	2	10	10	92
0	4	0	0.06578721946205242	0.06578721946205242	0.9051724137931034	0.02155172413793104	0.07327586206896551	0	2	5	20	241

c2_flag_apn_rep1	c2_APN_total_reads_rep1	c2_APN_both_rep1	c2_g1_total_rep2	c2_g2_total_rep2	c2_both_total_rep2	c2_flag_apn_rep2	c2_APN_total_reads_rep2	c2_APN_both_rep2
0	4.329981450746852	3.477106316508835	69	80	610	0	3.5567704773991995	2.858537537830714
0	2.7275136169586345	2.2328867952303844	15	17	138	0	2.402473134108641	1.950242897099956
1	7.240404040404041	5.947474747474747	14	2	75	1	5.8828282828282825	4.848484848484849
0	4.374850094226487	3.9636799725886585	5	14	179	0	3.256467363371596	2.943978070926846

Step 3b: Component Script: Generate Headers for one condition

The Generate Headers for One Condition tool creates new by mapping the user-specified compare names to a general variable. This tool is used instead of the Merge Compares and Generate Headers tool for testing AI in on condition.

This tool requires a Compare Design File that explicitly lists the comparison the user wants to do.

Click on the 'The Generate Headers for One Condition' Tool

Upload the (1) Collection created by the Merge Priors to Compare tool and the (2) User-supplied Compare Design file specifying one condition

1. Select the collection output from **Merge Priors to Compare**

2. Select the **Compare Design file**. This must be a TSV file

Merge Two or More Compare Datasets and Generate Headers
in preparation for Bayesian Analysis for AI (Galaxy Version 0.1.0)
Favorite Options

Collection of Compare Datasets containing Priors

579: Merge priors to compare on data 577, data 576, and others: Merge Priors to Compare

Select the collection containing files to be merged. [Required]

Compare Design file

578: compare_df_two_conditions_BASE.tsv

Select your Compare Design file. See below for correct formatting of this design file [Required]

Execute

Step 3 Outputs

-The tool generates one file:

- A TSV file containing the compare listed in the Compare Design File. The number of columns is contingent upon the number of replicates for each compare. The headers are:
 - Unique Feature ID
 - prior_{compare}_both: The calculated probability estimate that a feature will map equally to both parents, for a given compare (ie C1, C2, C3, C_n)
 - prior_{compare}_{g1/g2}: The calculated probability estimate that a feature will map preferentially to one of the updated parental genomes for a given compare
 - {compare}_flag_analyze: 0/1 binary indicator variable where a “1” means that at least one replicate in the compare has an APN greater than the user-specified cutoff value.
 - {compare}_num_reps: Number of replicates present for a given compare
 - count_{compare}_{g1/g2}_total_{replicate number}: Total number of uniquely mapped reads aligning preferentially to one of the updated parental genomes for a feature within a given compare
 - count_{compare}_both_total_{replicate number}: Total number of uniquely mapped reads aligning equally to both updated parental genomes for a feature within a given compare
 - {compare}_flag_apn_{replicate number}: 0/1 where “1” indicates that a feature has a APN greater than or equal to the user-defined APN threshold
 - {compare}_APN_total_reads_{replicate number}: Calculated APN value for all unique reads for a replicate within a compare
 - {compare}_APN_both_{replicate number}: Calculated APN values for unique reads mapping equally well to both updated parental genomes for a replicate within a compare

An example output file (split for visibility):

FEATURE_ID	prior_c1_both	prior_c1_g1	prior_c1_g2	c1_flag_analyze	c1_num_reps	c1_g1_total_rep1	c1_g2_total_rep1	c1_both_total_rep1	c1_flag_apn_rep1	c1_APN_total_reads_rep1	c1_APN_both_rep1	c1_g1_total_rep2
I(1)G0196	0.8097560975609757	0.1024390243902439	0.08780487804878047	0	2	21	18	157	0	0.9184809137947868	0.7357219564580689	0
CG8920	0.7647058823529411	0.0196078431372549	0.215686274509804	0	2	1	11	37	0	0.6924775504195495	0.5228912115412925	0
CG10932	0.76	0.16	0.08	0	2	4	2	19	0	1.616161616161616	1.2282828282828282	0
Mapmodulin	0.8732394366197183	0.014084507042253518	0.11267605633802814	0	2	1	8	58	0	1.1019359259893782	0.9539146821997602	0
	0	9	0	0.04217514400078102	0.04217514400078102							
	0	2	0	0.02826438981304284	0.02826438981304284							
	0	0	0	0.0	0.0							
	0	4	0	0.06578721946205242	0.06578721946205242							

Run Bayesian Model Workflow

Run Bayesian Model Workflow Overview

The Run Bayesian workflow deploys the Novelo et. al. 2014 model to test for AI within each of two conditions and between the conditions. The model implements a multi-faceted approach of analysis for Allelic Imbalance (AI) that accounts for the possibility of bias in lieu of the binomial test, a common application utilized to determine statistical significance of AI. The binomial test always returns a higher rate of type I error in comparison to the Bayesian Model, as the model estimates possible points of bias through the results of a level alpha test, thus lowering the type I error rate. The Poisson-Gamma model is used to estimate and correct for points of bias, without the need for data filtering, which can lead to sources of bias being missed.

The model is used to test three hypotheses of interest:

for $i = 1, 2$ and $k = 1, 2, \dots, K_i$.

Allelic balance in Comparete 1 (e.g., Male) or, equivalently, $H_{01}: \alpha_1 = 1$.

Allelic balance in Comparete 2 (e.g., Female), $H_{02}: \alpha_2 = 1$.

The level of AI is not different between comparates.

If either H_{01} or H_{02} are rejected, (at a threshold of posterior p-value 0.05) “AI_{c_n}_decision”, where n refers to the comparete number, is set to 1. If either of these cannot be rejected, these column values remain 0.

Testing the third null hypothesis tests determines if the true proportion of reads coming from the G1 is the same within Comparete 1 as it is within Comparete 2 (Novelo et. al 2018). This is a separate process that effectively tests if $\alpha_1 = \alpha_2$. If H_{03} is rejected, the Bayesian model output columns “AI_diffinc2andc1”=1. All model output is compiled into one wide format TSV file per Bayesian input file to be deposited in the user inputted directory.

More information on the Bayesian Model can be found here:

Leon-Novelo, et al. (2014). A flexible Bayesian method for detecting allelic imbalance in RNA-seq data. *BMC Genomics*. 15(1):920. doi: 10.1186/1471-2164-15-920.

Bayesian Model Workflow considerations

The statistical model is packaged in the script environmentalmodel2.stan (supported on the STAN platform) and wrapped in the R script (stan2_implementation.r). This script has hardcoded the names of conditions used in the original data implementation. A Python wrapper (calling stan2_implementation.r as a subprocess), NBmodel_stan2.py, generalizes the model’s input name requirements.

Bayesian Model Workflow Inputs

The Run Bayesian workflow requires two input files:

1) The TSV file containing merged comparates. This file can be generated from the Merge Priors Workflow (specifically the Merge Two or More Compare Datasets).

2) [Compare Design File](#)

Bayesian Model Workflow Outputs

The Run Bayesian workflow outputs one file:

1) A TSV file containing results.

Component_script: NBmodel_stan2

This python wrapper script sets parameters for the NBmodel, it formats input data and initiates the STAN implementation of the NBModel.

Click on the ‘Run Bayesian Model’ tool

Upload the (1) Compare Design File and the (2) File or Collection of files containing the comparates merged together.

1. Select the **Compare Design File** from the drop down menu. This file must be supplied by the user

2. Select the **Dataset containing the merged comparates with their the prior** probability estimates. This is the output from the Merge Priors workflow

3. Enter the **number of compare conditions** being tested. For example, if testing male versus female for the same genotype, there are two comparates.

4. Enter the **number of iterations**. The default is 100000.

5. Enter the **number of warmup** iterations. This is the number of “burn-in” iterations needed for tuning the Markov Chain Monte Carlo (MCMC) process before keeping estimates. Default is 10000

Run the Bayesian model: NBModel_stan2_flex_prior.R (Galaxy Version 0.1.0)

Design File

578: compare_df_two_conditions_BASE.tsv

Select your Compare Design File.

Select the dataset or collection of datasets containing the comparates merged with the priors

583: Merge Two or More Compare Datasets and Generate Headers on data 578, data 582, and data

Select the dataset or collection of datasets that contain the merged comparates and new headers.

conditions

2

Enter the number of conditions you're comparing (e.g. M v F would be 2)

iterations

100000

Enter the number of iterations [default = 100000]

warmup

10000

Enter the warmup number [default = 10000]

Execute

A TSV file containing the following columns is output from the Bayesian Model,

- comparison: The comparison being tested (this is the same as compID in the Compare Design File)
- FEATURE_ID
- {compare}_num_reps: The number of biological replicates for the indicated compare.
- counts_{compare}_{g1/g2}: The total number of reads that aligned preferentially to G1 (or G2) for the indicated compare. This is, for G1: $\sum_{k=1}^{K_n} x_{1k}$, and for G2: $\sum_{k=1}^{K_n} y_{1k}$ where $n = 1, 2$ indexes the compare.
- counts_{compare}_both: The total number of reads that aligned equally to both parental genomes for the indicated compare.. This is, $\sum_{k=1}^{K_n} z_{nk}$
- prior_{compare}_{g1/g2}: The prior probability estimate that a given read will map to G1 (or G2) for the indicated compare. This is for G1: r_{n1} , and for G2: r_{n2}
- H3_Independence_Bayesian_evidence: Bayesian evidence for testing the null that the alleles are independent variables. Minimum value of ev such that the $1-ev$ central credible interval for $\alpha_1-\alpha_2$
- {compare}_sampleprop: The sample proportion of reads amongst the reads that mapped to G1 or G2, but not to both, that have mapped preferentially to G1 within the indicated compare. This is $\sum_{k=1}^{K_n} x_{nk} / \sum_{k=1}^{K_n} (x_{nk} + y_{nk})$
- {compare}_theta: The point estimate (posterior expected value) of the proportion generated by G1 after adjusting for systematic bias in compare n , θ_{n1} . This proportion different from $1/2$ implies AI in the compare. Since {compare}_theta is an estimate of θ_{n1} , credible intervals for θ_{n1} are used to flag the compare n as in AI or not.
- {compare}_q025: Lower bound for the 95% central credible interval, or equivalently, the 2.5% quartile, of the posterior distribution of θ_{n1}
- {compare}_q975: Upper bound of 95% central credible interval, or equivalently, quantile 97.5% of the posterior distribution of θ_{n1}
- {compare}_Bayes_evidence: The Bayesian evidence. Smaller values can lead to rejection of the null. Ev is the smallest value such that the $1-ev$ central credible interval for θ_{n1} does not contain the value $\theta_{n1}=1/2$, that implies allelic imbalance in compare n .

- {compare}_AI_decision: A 0/1 binary indicator flag where a “1” indicates that the posterior p-value was less than 0.05 for the indicated compare.
- alpha_postmean: Alpha value for compare 1, $\alpha_1 = 0.5$ indicator of allelic imbalance.
- alpha2_postmean Alpha value for compare 1, $\alpha_2 = 0.5$ indicator of allelic imbalance.
- flagAnalyze: 0/1 binary indicator flag where a “1” indicates that the compare_flag_analyze variables going into the model were BOTH equal to 1.

comparison	FEATURE_ID	W55_M_num_reps	W55_V_num_reps	counts_W55_M_g1	counts_W55_M_g2	counts_W55_M_both	counts_W55_V_g1	counts_W55_V_g2	counts_W55_V_both	prior_W55_M_g1
W55_M_V	I(1)G0196	3	3	362	520	3990	413	605	4475	0.0743021346469622
W55_M_V	CG8920	3	3	45	79	661	91	101	723	0.0573248407643312
W55_M_V	CG10932	3	3	49	18	336	41	25	337	0.121588089330025
W55_M_V	Mapmodulin	3	3	23	136	1553	15	188	1912	0.0134345794392523

prior_W55_M_g2	prior_W55_V_g1	prior_W55_V_g2	H3_independence_Bayesian_pvalue	g1_W55_M_sampleprop	g1_W55_M_theta	g1_W55_M_q025	g1_W55_M_q975	g1_W55_M_Bayes_pval
0.10673234811165801	0.0751866011287093	0.110140178408884	0.8691	0.4104	0.4948	0.376	0.6137	0.9294
0.10063694267515899	0.0994535519125683	0.11038251366120198	0.589	0.3629	0.501	0.3528	0.6508	0.9886
0.0446650124069479	0.101736972704715	0.0620347394540943	0.8385	0.7313	0.5272	0.3802	0.6702	0.7083
0.0794392523364486	0.00709219858156028	0.0888888888888889	0.8692	0.1447	0.4701	0.3393	0.6075	0.6559999999999999

g1_W55_M_AI_decision	g1_W55_V_sampleprop	g1_W55_V_theta	g1_W55_V_q025	g1_W55_V_q975	g1_W55_V_Bayes_pval	g1_W55_V_AI_decision	alpha1_postmean	alpha2_postmean	flaganalyze
0	0.4057	0.5084	0.3908	0.6248	0.8794	0	1.0182	0.9904	1
0	0.474	0.4445	0.3034	0.5945	0.4543	0	1.0101	1.1341	1
0	0.6212	0.506	0.3618	0.6491	0.9313	0	0.9569	0.9990000000000001	1
0	0.0739	0.4549	0.3167	0.6005	0.5285	0	1.0734	1.1092	1

