GBS PIPELINES

[Jeff Glaubitz, Terry Casstevens, Rob Elshire, James Harriman, Ed Buckler]

Launching the pipeline via the perl script for running any plugin

```
run_pipeline.pl -fork1 -PluginName -option –endPlugin -runfork1
```

Allocate memory directly in the command line for any plugin

```
run_pipeline.pl –Xmx4g -fork1 -PluginName -option –endPlugin
-runfork1  0723788959
```

STEP 01: Sequence file and folder structure

Recommended directory (folder) structure for a GBS analysis

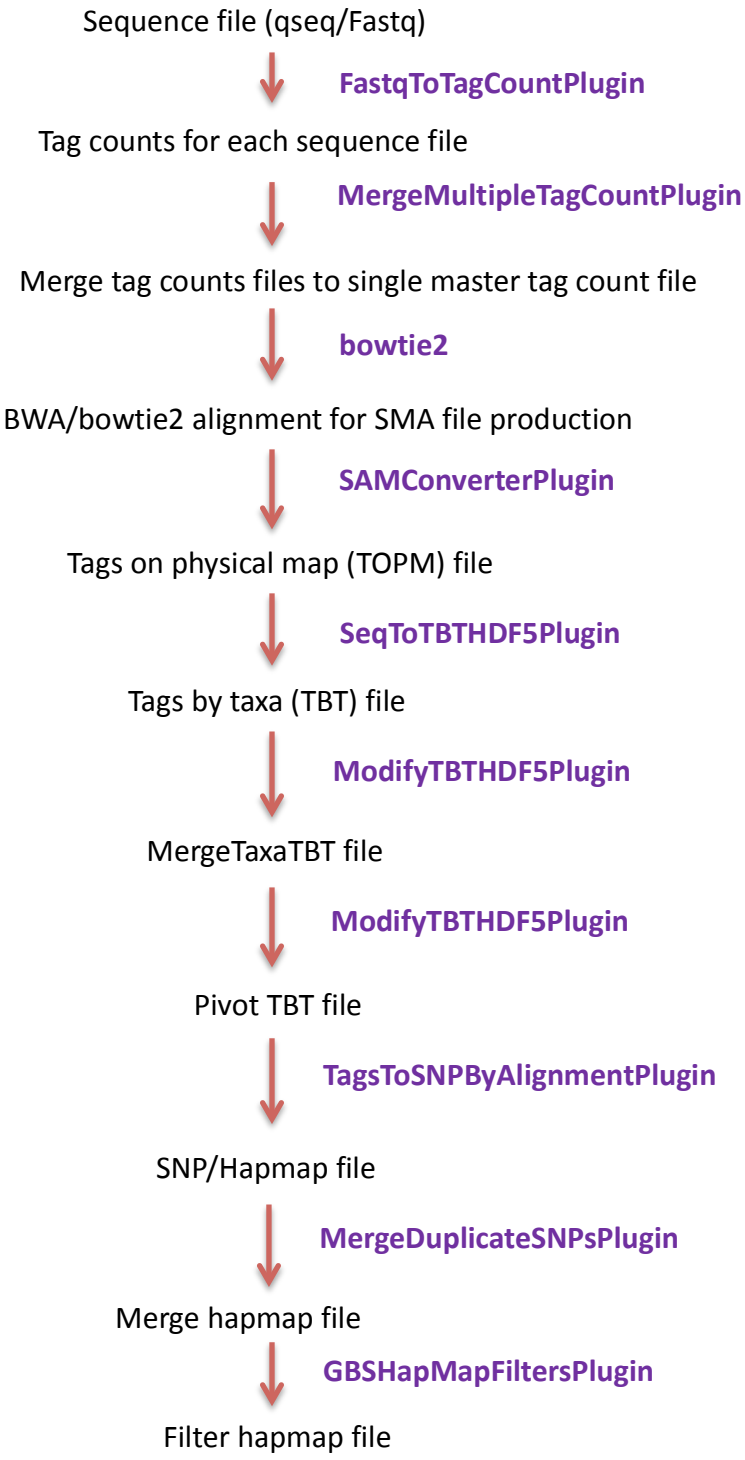
A dot (.) will represent the working directory (folder) for your analysis, which will be your current working directory (e.g., /home/user#/sorghum)

The example commands below don't create the directories (and will fail if they don't already exist), so at the start of the analysis, create the following directories inside your working directory.

01_qseq (original raw data files, one file per flowcell lane)
02_tagcounts (for output from FastqToTagCountPlugin OR QseqToTagCountPlugin)
03_mergedTagcounts (for output from MergeMultipleTagCountPlugin)
04_topm (for output from SAMConverterPlugin
05_tbt (for output from FastqToTBTPlugin)
06_mergedTBT (for output from MergeTagsByTaxaFilesPlugin)
07_hapmapRaw (for output from TagsToSNPByAlignmentPlugin)
08_hapmapMergedSNPs (for output from MergeDuplicateSNPsPlugin)
09_hapMapfilter (for output from GBSHapMapFiltersPlugin)
50_keyFile (folder containing key file)
bowtie2-2.2.2
Genome
Logfile
Maize_ref_chr9_10_ILRI
Tassel4.0_standalone

scripts

GBS Work flow:



STEP 02: FastqToTagCountPlugin

FastqToTagCountPlugin

Summary:
Derives a tagCount list for each FASTQ file in the input directory (and all subdirectories thereof). Keeps only good reads having a barcode and a cut site and no N's in the useful part of the sequence. Trims off the barcodes and truncates sequences that (1) have a second cut site, or (2) read into the common adapter.

Input:
- Barcode key file (see example in Appendix 1)
- Directory (folder) containing FASTQ files

Output:
- Directory (folder) containing a corresponding tagCount (.cnt) file for every FASTQ file in the input directory

Arguments:

FastqToTagCountPlugin	
-i	Input directory containing FASTQ text (_fastq.txt) or gzipped FASTQ (_fastq.gz) text files. NOTE: Directory will be searched recursively, and should be written without a slash after its name.
-k	Key file listing barcodes for each sample and plate layout. See Appendix 1.
-e	Enzyme used to create the GBS library (*Ape*KI, *PstI* or several others).
-s	Maximum number of good, barcoded reads per lane. Default: **300,000,000**. Make less up to 2000000000
-c	Minimum number of times a tag must be present to be output. Default: 1
-o	Output directory to contain ouput .cnt (tag count) files, one per input FASTQ file. Defaults to input directory (the default is not recommended - it is best to use a separate directory).

Example command:
```
/programs/tassel/run_pipeline.pl -fork1 -FastqToTagCountPlugin -i fastq -k
myGBSProject_key.txt -e ApeKI -o tagCounts -endPlugin -runfork1
```

```
module load tassel/4.0

run_pipeline.pl -Xmx6g  -fork1 -FastqToTagCountPlugin -i ./01_qseq/ -k
./50_key/Pipeline_Testing_key.txt -e ApeKI -s 200000000 -o ./02_tagcounts/ -
endPlugin -runfork1 | tee -a ./Logfile/FastqToTagCount
```

GOAL: Main goal of this plugin is to capture all unique sequence tags separately from each sequence (QSEQ/FASTQ) file.

STEP 03: MergeMultipleTagCountPlugin

Summary:
Merges each tagCount file in the input directory into a single "master" tagCount list. Only keeps tags with a total count (after merger) greater than or equal to that specified by the **-c option** (***minimum number of times a tag must be present to be output***). It has two output formats: (1) a binary output format (.cnt) that is used by the FastqToTBTPlugin to construct tags by taxa (TBT) files, and (2) a fastq text format (.fq) that is used as an input to BWA or bowtie2 to align tags to the reference genome. For clarity, the latter functionality (conversion of a master tagCount list into fastq format) has been made into its own plugin (see TagCountToFastqPlugin below).

Input:
- Input directory (folder) containing tagCount (.cnt) files

Output:
- Merged tagCount file (it is best to send this to a separate directory from the input directory)

Arguments:

MergeMultipleTagCountPlugin	
-i	Input directory containing tagCount (.cnt) files.
-o	Output file name (should be in a separate directory from the input).
-c	Minimum number of times a tag must be present to be output. Default: 1
-t	Specifies that reads should be output in FASTQ text format (for use as input to either BWA or bowtie2 for alignment to the reference genome).

Example command:
```
/programs/tassel/run_pipeline.pl -fork1 -MergeMultipleTagCountPlugin -i
tagCounts -o mergedTagCounts/myMasterGBSTags.cnt -c 5 -endPlugin -runfork1
```

MergeMultipleTagCountPlugin for fastq format of masterTag counts file

```
rajneesh@icrisatbioinfo:~/MaizeGBS_batch8and9rawGBSdata/GBSpractice
$ perl ./tassel4.0_standalone/run_pipeline.pl -Xmx6g  -fork1 -
MergeMultipleTagCountPlugin -i ./02_tagcounts  -o
./03_mergedTagcounts/ILRI_mergeTagcounts.cnt -c 5  -endPlugin -runfork1 |
tee -a ./Logfile/mergeMultipleTagCount
```

GOAL: Unique sequence tags from each separate sequence file will be merged together. In this example, this master tag counts file contains unique tags for all tags which were present at least 5 times in the combined dataset. The same plugin is used to generate fastq file format of tags to align them against reference genome.

STEP 04: TagCountToFastqPlugin

Summary:

Converts a master tagCount file containing all the tags of interest for your species/experiment (*i.e.*, all of the tags with a minimum count greater than the **-c** parameter used in the MergeMultipleTagCountPlugin) from binary (.cnt) format into a FASTQ format file (.fq) that can then be used as input to one of the aligners BWA or bowtie2.

Input:

- A binary tag count (.cnt) file containing all tags of interest (= master tag list).

Output:

- The master tag list in FASTQ format (.fq). Can be used as input to BWA or bowtie2.

Arguments:

TagCountToFastqPlugin	
-i	Input binary tag count (.cnt) file
-o	Output FASTQ file to use as input for BWA or bowtie2.
-c	Minimum count of reads for a tag to be output (default: 1)

Example command:

```
/programs/tassel/run_pipeline.pl -fork1 -TagCountsToFastqPlugin -i
myMasterGBSTags.cnt -o myMasterGBSTags.fq -c 5 -endPlugin -runfork1
```

```
rajneesh@icrisatbioinfo:~/MaizeGBS_batch8and9rawGBSdata/GBSpractice
$ perl ./tassel4.0_standalone/run_pipeline.pl -Xmx6g  -fork1 -
TagCountToFastqPlugin -i
./03_mergedTagcounts/ILRI_mergeTagcounts.cnt/  -o
./03_mergedTagcounts/ILRI_mergeTagcounts.fq -c 5  -endPlugin -
runfork1 | tee -a ./Logfile/TagCountsToFastPlugin
```

STEP 05: Bowtie2 Alignment (Not in the TASSEL pipeline)

Indexing with bowtie2

Summary:
Creates a series of support files needed to operate bowtie2. You must have bowtie2 installed on your computer. For more details, consult the bowtie2 manual (http://computing.bio.cam.ac.uk/local/doc/bowtie2.html).

Input:
- A FASTA file containing one record for each chromosome or contig in the genome. In order for the SAMConverterPlugin (see below) to work correctly, the header of each record should contain only a single integer corresponding to that chromosome or contig's number. The suffix "chr" prior to the chromosome number is permissible, but no others. For example, the headers ">1", ">2", ">3", etc. are acceptable, as are ">chr1", ">chr2", ">chr3", etc.

Output:
- A series of support files with the same name as the output base name but with different suffixes.

Key Arguments:

bowtie2-build	
input fasta files	a comma separated list of input fasta files (see above for header information)
output base name	output file base name (e.g. ZmB73_RefGen_v2.fa)

Example command:
```
bowtie2-build
chr1.fasta,chr2.fasta,chr3.fasta,chr4.fasta,chr5.fasta,chr6.fasta,chr7.fasta,
chr8.fasta,chr9.fasta,chr10.fasta,chrPt.fasta,chrMt.fasta,chrUNKNOWN.fasta
ZmB73_RefGen_v2.fa
```

Gory Details:
This command builds a set of indices from the reference genome. These indices are subsequently used by the "bowtie2" command for fast alignment of tags. For more details see the bowtie2 manual.

The Tassel3 GBS pipeline can use tag alignments produced either by BWA or by the default (-M) mode of bowtie2. In our experience, bowtie2 is more sensitive than BWA, and produces results that are more similar to BLAST. The tradeoff of the greater sensitivity of bowtie2 is that misalignment of tags will occur more often (e.g., tags from paralogous loci or from inserted sequences not present in the reference).

Alignment with bowtie2

Summary:
Aligns the master set of GBS tags to the reference genome. This input master tag list is stored in the fastq (.fq) file produced by TagCountToFastqPlugin (or the MergeMultipleTagCountPlugin with the -t option).

Input:
- Tag count file in FASTQ format (.fq) produced by the TagCountToFastqPlugin (or the MergeMultipleTagCountPlugin with the -t option).

Output:

- SAM alignment file that can be read by the SAMConverterPlugin of our GBS pipeline.

Key Arguments:

bowtie2	
-M X	The "-M mode" is the default mode of bowtie2, where it "*searches for at most X+1 distinct, valid alignments for each read. The search terminates when it can't find more distinct valid alignments, or when it finds X+1 distinct alignments, whichever happens first.* ***Only the best alignment is reported***" (ties are resolved at random). If multiple valid alignments are found, the alignment score for the second best alignment is stored in the XS:i field of the SAM output (alignment info) for that tag. In bowtie 2.1, this -M flag is deprecated, as it is the default mode. Therefore, omitting the -M flag should work in the same manner, except that search depth is now controlled by the -D and -R options (defaults of 15 and 2, respectively). Consult the bowtie2 manual for more details: http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml
-p X	The number of processors to be used. More is faster.
--very-sensitive-local	This sets the sensitivity.
-x	The basename of the reference genome index.
-U	The input fastq file.
-S	The output sam file name.

Example command:

```
bowtie2 -M 4 -p 15 --very-sensitive-local -x ../zeareference/bowtie2/ -U
AllZeaMasterTags_c10_20120607.fq -S AllZeaMasterTags_c10_20120613.sam

Rajneeshs-MacBook-Pro:ILRI_GBStraining2 RajneeshPaliwal$ cd ~/Desktop/Test_ILRI-
GBS/ILRI_GBStraining2/bowtie2-2.2.2/

Rajneeshs-MacBook-Pro:bowtie2-2.2.2 RajneeshPaliwal$ bowtie2 -M 4 -p 4 --very-
sensitive-local -x ./Genome/ZmB73_RefGenome -U
./03_mergedTagcounts/ILRI_mergeTagcounts.fq -S ./04_topm/ILRI_mergeTagcounts.sam |
tee -a ./Logfile/bowtieLog
```

GOAL: Tags will be aligned to reference genome in SAM format.

STEP 06: SAMConverterPlugin

SAMConverterPlugin

Summary:
Converts a SAM format alignment (.sam) file produced by one of the aligners, BWA or bowtie2, into a binary tagsOnPhysicalMap (.topm) file that can be used by the DiscoverySNPCallerPlugin for calling SNPs.

Input:
- SAM format alignment (.sam) file produced by BWA or by the default (-M) mode of bowtie2

Output:
- binary tagsOnPhysicalMap (.topm) file that can be used by the DiscoverySNPCallerPlugin for calling SNPs

Arguments:

SAMConverterPlugin	
-i	Alignment file in SAM format (.sam) produced by BWA or bowtie2.
-o	Output TagsOnPhysicalMap (TOPM) file that can be used by the DiscoverySNPCallerPlugin for calling SNPs (or by the SeqToTBTHDF5Plugin or FastqToTBTPlugin as a master tag list). We recommend using the extension ".topm". In addition to the tags that aligned to a single best genomic position, tags with either multiple positions or no alignment are still included in this output TOPM (in other words, all tags that were fed into BWA or bowtie2 should end up in the TOPM file produced by this plugin, regardless of whether they could be aligned to the genome or not, or of how many positions they aligned to).

Example command:
```
/programs/tassel/run_pipeline.pl -fork1 -SAMConverterPlugin
-i mergedTagCounts/myAlignedMasterTags.sam -o topm/myMasterTags.topm
-endPlugin -runfork1
```

```
Rajneeshs-MacBook-Pro:GBSpractice RajneeshPaliwal$ run_pipeline.pl -Xmx6g -
fork1 -SAMConverterPlugin -i ./04_topm/ILRI_mergeTagcounts.sam -o
./04_topm/ILRI_mergeMasterTagCounts.topm -endPlugin -runfork1 | tee -a
./Logfile/mergeTAGCountsTOPM.log
```

GOAL: SAM format is converted to Tags On Physical Map (TOPM) format which can be used further for SNP calling.

STEP 07: FastqToTBTPlugin

Summary:
Generates a TagsByTaxa file for each FASTQ file in the input directory (or in subfolders thereof). One TagsByTaxa file is produced per FASTQ file. Requires a master list of tags of interest, which may come either from a tagCount (.cnt) or tagsOnPhysicalMap (.topm) file. **If your input files are in QSEQ format, use QseqToTBTPlugin instead (same arguments).** To obtain a single TagByTaxa file in HDF5 format, and thus reduce the amount of disk space required for a large analysis, use the SeqToTBTHDF5Plugin instead of this FastqToTBTPlugin

Input:
- Directory (folder) containing FASTQ files
- Barcode key file (see example in Appendix 1)
- Master tag list in the form of either a ***binary*** tagCount (.cnt) file or a tagsOnPhysicalMap (.topm) file

Output:
- Directory (folder) containing a corresponding tagsByTaxa file for every FASTQ file in the input directory

Arguments:

FastqToTBTPlugin	
-i	Input directory containing FASTQ files with raw GBS sequence reads.
-k	Barcode key file. See Appendix 1 for an example.
-e	Enzyme used to create the GBS library (e.g., ApeKI).
-o	Output directory.
-c	Minimum taxa count within a FASTQ file for a tag to be output. Default: 1
-t	Master tagCount (.cnt) file containing the tags of interest. This file must be binary (.cnt). The -t option is mutually exclusive with the -m option.
-m	TagsOnPhysicalMap (.topm) file containing the tags of interest. The -m option is mutually exclusive with the -t option.
-y	Output in TBTByte format (counts from 0-127) instead of TBTBit (0 or 1)

Example command:
```
/programs/tassel/run_pipeline.pl -fork1 -FastqToTBTPlugin -i fastq -k
myGBSProject_key.txt -e ApeKI -o tbt -y -t mergedTagCounts/myMasterTags.cnt -
endPlugin -runfork1
```

```
Rajneeshs-MacBook-Pro:GBSpractice RajneeshPaliwal$ run_pipeline.pl -Xmx6g -
fork1 -FastqToTBTPlugin -i ./01_qseq/ -k ./50_key/Pipeline_Testing_key.txt -
e ApeKI -o ./05_tbt -y -t ./03_mergedTagcounts/ILRI_mergeTagcounts.cnt  -
endPlugin -runfork1 | tee -a ./Logfile/FastqTBTPlugin.log
```

GOAL: Similar to FastqToTagCountPlugin, FastqToTBTPlugin parses FASTQ files containing raw GBS sequence data for good reads that contain a barcode and cut site remnant and that have no N's in the first 64 bases after the barcode, and trims them to 64 bases (not including the barcode). As in FastqToTagCountPlugin, FastqToTBTPlugin appropriately truncates reads that contain either a full cut site or the beginning of the common adapter within the first 64 bases, and pads them to 64 bases with polyA..

STEP 07: MergeTagsByTaxaFilesPlugin

Summary:
Merges all .tbt.bin and/or (preferably) .tbt.byte files present in the input directory and all of its subdirectories.

Input:
- Directory (folder) containing multiple tagsByTaxa (.tbt.byte or .tbt.bin) files (produced by FastqToTBTPlugin). For the best genotyping results (proper calling of heterozygotes), we recommend using .tbt.byte files as input (produced by the FastqToTBTPlugin using the -y option)

Output:
- Merged tagsByTaxa file (it is best to send this to a separate directory from the input directory)

Arguments:

MergeTagsByTaxaFilesPlugin	
-i	Input directory containing multiple tagsByTaxa files (preferably tbt.byte files).
-o	Output file name (should be in a separate directory from the input). Use extension matching the type of input file (".tbt.byte" or ".tbt.bin").
-s	Maximum number of tags the TBT can hold while merging (default: 200,000,000). Reduce this only if you run out of memory (omit the commas).
-x	Merges tag counts of taxa with identical short names. Not performed by default.

Example command:
```
/programs/tassel/run_pipeline.pl -fork1 -MergeTagsByTaxaFilesPlugin -i tbt
-o mergedTBT/myStudy.tbt.byte -endPlugin -runfork1
```

```
Rajneeshs-MacBook-Pro:GBSpractice RajneeshPaliwal$ run_pipeline.pl -Xmx6g -
fork1 -MergeTagsByTaxaFilesPlugin -i ./05_tbt/  -o ./06_mergedTBT/ILRI-
mergedTBTFiles.tbt.byte  -endPlugin -runfork1 | tee -a
./Logfile/MergedTaxbyTexaFiles
```

Gory Details:
This step merges the separate tagsByTaxa files produced by the FastqToTBTPlugin (and/or QseqToTBTPlugin) into a single, experiment-wide tagsByTaxa (TBT) file for all of the flow cell lanes in your experiment.

Step08: SeqToTBTHDF5Plugin

Summary:
This plugin processes all of the raw GBS sequence files (FASTQ or QSEQ format) in the input directory (and all of its subdirectories) and generates a "Tags by Taxa" (TBT) data file in HDF5 format. This plugin and the ModifyTBTHDF5Plugin are newer additions to the pipeline that can be used in place of the FastqToTBTPlugin and the MergeMultipleTagsByTaxaFilesPlugin (which do not produce HDF5 formatted output). Only reads that match one of the tags in the input master tag list will be recorded in the output TBT HDF5. The input master tag list can be in the form of either a binary tag count (.cnt) file or a TagsOnPhysicalMap (.topm) file.

Input:
- Directory (folder) containing FASTQ or QSEQ raw GBS sequence files
- Barcode key file (see example in Appendix 1)
- Master tag list in the form of either a binary tagCount (.cnt) file or a tagsOnPhysicalMap (.topm) file

Output:
- A single TagsByTaxa (TBT) file in HDF5 format (*TBT.h5) recording how often each GBS tag in the master tag list was observed in each taxon (sample) present in the input FASTQ or QSEQ files. A "taxon" in the output TagsByTaxa file represents an individual DNA sample from a particular flowcell lane that has been distinguished by a particular barcode.
- Arguments:

SeqToTBTHDF5Plugin	
-i	Input directory containing FASTQ or QSEQ raw GBS sequence files
-k	Barcode key file. See Appendix 1 for an example.
-e	Enzyme used to create the GBS library (see FastqToTagCountPlugin for list of available enzymes).
-o	Output TagsByTaxa (TBT) file in HDF5 format. Use the extension ".h5".
-s	Max good reads per lane. (Optional. Default is 500,000,000).
-L	Output log file
-t	Master tagCount (.cnt) file containing the tags of interest. This file must be binary (.cnt). The -t option is mutually exclusive with the -m option.
-m	TagsOnPhysicalMap (.topm) file containing the tags of interest. The -m option is mutually exclusive with the -t option.

Example command:
```
/programs/tassel/run_pipeline.pl -fork1 -SeqToTBTHDF5Plugin -i fastq
-k key/AllZea_key.txt -e ApeKI -o tbt/AllZeaTBT.h5 -s 900000000 -L
tbt/AllZeaTBT.log -t mergedTagCounts/AllZeaMasterTags.cnt -endPlugin
-runfork1
```

```
Rajneeshs-MacBook-Pro:GBSpractice RajneeshPaliwal$ run_pipeline.pl -Xmx6g -
fork1 -SeqToTBTHDF5Plugin -i ./01_qseq/ -k ./50_key/Pipeline_Testing_key.txt -e
ApeKI -o ./06_mergedTBT/ILRISeqtoTBTHDF5Plugin.tbt.h5 -s 900000000 -L
./06_mergedTBT/ILRIseqTOTBTHDF5plugin.log  -t
./03_mergedTagcounts/ILRI_mergeTagcounts.cnt -endPlugin -runfork1 | tee -a
./Logfile/SeqtoTBTHDF5Plugin
```

Gory Details: The purpose of the SeqToTBTHDF5Plugin is to produce a TagsByTaxa (TBT) file in HDF5 format recording how many times each GBS tag of interest was observed in each taxon (sample). It is more recent alternative to the FastqToTBT and MergeMultipleTagsByTaxaFiles plugins, which (together) also produce a single, master TBT file, but that TBT file is not in HDF5 format.

Step09: ModifyTBTHDF5Plugin

Summary:
This plugin makes changes to a TBT HDF5 file. There are three things that it can do, but it can only do one of them at a time:

1. Merge two TBT HDF5 files into one, or
2. Merge taxa with identical LibraryPrepIDs, or
3. Transpose the TBT HDF5 into an orientation that is more efficiently used for SNP calling (by allowing faster access of all the counts across taxa for a particular tag).

Input:

- TBT HDF5 (*TBT.h5) file (one or multiple depending on action)

Output:

- Modified TBT HDF5 (*TBT.h5) file

Arguments:

ModifyTBTHDF5Plugin	
-o	Target TBT HDF5 (*TBT.h5) file to be modified
One of either:	(depending on the modification you wish to make)
-i	TBT HDF5 (*TBT.h5) file containing additional taxa to be added to the target TBT HDF5 file
-c	Merge taxa with same LibraryPrepID in the target TBT HDF5 file
-p	Pivot (transpose) the target TBT HDF5 file into a tag-optimized orientation

Example commands:

Merging two TBT HDF5 files:

```
/programs/tassel/run_pipeline.pl -fork1 -ModifyTBTHDF5Plugin -o
mergedTBT/mergedTBT.h5 -i tbt/part2TBT.h5 -endPlugin -runfork1
```

Merging taxa with the same LibraryPrepID:

```
/programs/tassel/run_pipeline.pl -fork1 -ModifyTBTHDF5Plugin -o
mergedTBT/mergedTBT.h5 -c -endPlugin -runfork1
```

Pivot (transpose) a TBT HDF5 file: Use this one for next step

```
/programs/tassel/run_pipeline.pl -fork1 -ModifyTBTHDF5Plugin -o
mergedTBT/mergedTBT.h5 -p pivotedTBT/pivotedTBT.h5 -endPlugin -runfork1
```

```
Use pivot a TBT HDF5
```

```
Rajneeshs-MacBook-Pro:GBSpractice RajneeshPaliwal$ run_pipeline.pl -Xmx6g -
fork1 -ModifyTBTHDF5Plugin -o ./06_mergedTBT/ILRISeqtoTBTHDF5Plugin.tbt.h5 -
p ./06_mergedTBT/ILRI_modifyTBT.h5   -L ./Logfile/ModifyTBThdf5Plugin -
endPlugin -runfork1
```

Gory Details:
The gory details for this plugin are organized by the three different functions this plugin can perform:

1) Merging two TBT HDF5 files into one TBT HDF5 file (**-i option**):

If you are working on a large project, to reduce the total amount of time it takes to create the "master" TBT HDF5 file, you might chose to run the SeqToTBTHDF5Plugin on multiple computers or processors and

thus create multiple TBT HDF5 files. This will result in multiple TBT HDF5 files which must be combined into one master file. To use the ModifyTBTHDF5Plugin to merge two TBT HDF5 files, use the **-o option** to specify an existing **target TBT HDF5 file** and the **-i option** to specify an existing **input TBT HDF5** to be added to the target. In practice, it is best to make a copy of what is to be the initial target TBT HDF5. This allows the original file to be kept. This plugin is then run repeatedly until all TBT HDF5 files generated in the SeqToTBTHDF5 step are merged into the target. For example, assume that the SeqToTBTHDF5Plugin was run in three stages (with each stage working with a different set of input FASTQ files), and that the output TBT HDF5 files were in a folder named "tbt" and were named

`part1TBT.h5`,
`part2TBT.h5`, and
`part3TBT.h5`.

To merge these three TBT HDF5 files, first, make a copy of `part1TBT.h5` named `mergedTBT/mergedTBT.h5`:

```
cp tbt/part1TBT.h5 mergedTBT/mergedTBT.h5
```

Then, run this ModifyTBTHDF5Plugin with the arguments:

```
-o mergedTBT/mergedTBT.h5 -i tbt/part2TBT.h5
```

Then, run it again with the arguments:

```
-o mergedTBT/mergedTBT.h5 -i tbt/part3TBT.h5
```

The TBT HDF5 file `mergedTBT/mergedTBT.h5` will then be a merger of all three parts.

2) Merging taxa by LibraryPrepID (**-c option**):

We typically run GBS at 384-plex and, if higher depth of coverage is desired, run the resulting pooled GBS library in replicate on multiple flow cell lanes (usually on four different lanes, with each lane on a different flow cell). In addition to increasing depth of coverage, this has the added benefits of spreading out systematic sequencing errors and of allowing lane effects and sample effects to be distinguished in statistical analyses of read depth per tag. To identify the replicate runs of each library prep (where a library prep is a particular combination of sample DNA and barcode in a particular well of a library prep plate), we assign each library prep a distinct LibraryPrepID which is recorded in the barcode key file (see Appendix 1). If you have run some of your library preps in replicate in this manner, and have recorded distinct LibraryPrepIDs in the key file for each Sample/Barcode/libraryPlateWell combination, then you can use the **-c option** of the ModifyTBTHDF5Plugin to merge the tag counts of the replicate library preps. When LibraryPrepIDs are present in the key file, taxa in the TBT files are named as SampleName:Flowcell:Lane:LibraryPrepID (rather than SampleName:Flowcell:Lane:Well). Replicate library preps will have the same SampleName and LibraryPrepID but the Flowcell and/or Lane portions of their name will be different. Using ModifyTBTHDF5Plugin with the **-c option** will merge the tagCounts for each set of taxa having the same LibraryPrepID by summing the counts for each tag. The resulting, merged taxon will be named SampleName:MRG:4:LibraryPrepID, where the 4 indicates that four replicates with the same LibraryPrepID were merged. The -c option of this plugin operates on only one file (the target TBT HDF5 file specified by the -o option) and changes that file. It is best to make a copy of the original TBT HDF5 file before performing this operation.

3) Pivot (transpose) TBT HDF5 into a tag-optimized orientation (**-p option**):

The TBT HDF5 created so far are in a taxon-optimized orientation best suited for operations involving taxa (adding and merging taxa). In order for the SNP caller (DiscoverySNPCallerPlugin) to run efficiently, the master TBT HDF5 needs to be in a tag-optimized orientation, allowing fast retrieval of the counts across taxa for a particular tag. You can produce a new, tag-optimized TBT HDF5 by using the **-p option** of this plugin. The target TBT HDF5 file (specified by the -o option) will not be changed by this operation.

Step10: DiscoverySNPCallerPlugin

Summary:
Aligns tags from the same physical location against one another, calls SNPs from each alignment, and then outputs the SNP genotypes to a HapMap format file (one file per chromosome).

Input:
- TagsByTaxa file (.tbt.byte or a tag-optimized TBT.h5) indicating the number of times each tag of interest was observed in each taxon. Use of a TBTBit (.tbt.bin) file is not recommended.
- TagsOnPhysicalMap file (.topm) containing genomic position of each tag of interest

Output:
- One HapMap format genotype file (.hmp.txt or .hmp.txt.gz) per chromosome.

Arguments:

DiscoverySNPCallerPlugin	
-i	Input TagsByTaxa (TBT) file. If you are using a TBT in .tbt.byte format, then use the -y option as well.
-y	Indicates that the input TBT specified by the -i option is in TBTByte (.tbt.byte) format (with counts from 0-127) rather than TBT HDF5 (*TBT.h5) format (also with counts from 0-127) or TBTBit (.tbt.bin) format (with counts of 0 or 1). Either TBTByte (.tbt.byte) or TBT HDF5 (*TBT.h5) format are recommended. If you use a TBTBit (.tbt.bin), then heterozygotes will be improperly called at higher coverage SNPs. If you don't use the -y option, then the type of TBT input file (TBT HDF5 or TBTBit) is determined from its file extension (.h5 or .tbt.bin, respectively)
-m	TagsOnPhysicalMap (.topm) file containing genomic position of tags.
-mUpd	Update the TOPM file with variants called during SNP calling. (it will be produce during this SNP calling)
-o	Output HapMap genotype file. Use a plus sign (+) as a wildcard character in place of the chromosome number (*e.g.,* -o hapmap/raw/myGBSGenos_chr+.hmp.txt). If you use a ".gz" suffix at the very end of the filename, the output genotype files will be gzip compressed.
-mxSites	Maximum number of sites (SNPs) output per chromosome (default: 200,000).
-mnF	Minimum value of F (inbreeding coefficient = 1-Ho/He). Not tested by default. 0.8 used
-p	Optional pedigree file containing full sample names & expected inbreeding coefficient (F) for each. Only taxa (samples) with expected F >= mnF used to calculate F (= 1-Ho/He) when applying the -mnF filter. See Appendix 2 for an example pedigree file. Default: use ALL taxa to calculate F.
-mnMAF	Minimum minor allele frequency (default: 0.01). SNPs that pass *either* the specified minimum minor allele frequency (mnMAF) or count (mnMAC) will be output.
-mnMAC	Minimum minor allele count (default: 10). SNPs that pass *either* the specified minimum minor allele count (mnMAC) or frequency (mnMAF) will be output. Used 100000

-mnLCov	Minimum locus coverage, *i.e.*, the proportion of taxa (samples) with at least one tag present from the TagLocus covering a SNP (default: 0.1).
-errRate	Average sequencing error rate per base (used to decide between heterozygous and homozygous calls) (default: 0.01).
-ref	Path to reference genome in fasta format. Ensures that a tag from the reference genome is always included when the tags at a locus are aligned against each other to call SNPs. The reference allele for each site is then provided in the output HapMap files, under the taxon name "REFERENCE_GENOME" (first taxon). DEFAULT: Don't use reference genome.
-inclRare	Include the rare alleles (3rd or 4th states) at sites. These are ignored by default (genotypes containing rare alleles are set to missing).
-inclGaps	Include sites where the major or minor allele is a gap. These sites are excluded by default.
-callBiSNPsWGap	For SNPs where the major and minor alleles are nucleotides, but the third allele is a gap (-), include the gap alleles in the genotype calls (default: ignore the gap alleles)
-sC	Start chromosome. Must be an integer. Generally 1 or may 0 in our maize case
-eC	End chromosome. Must be an integer. 10

Example command:

```
/programs/tassel/run_pipeline.pl -fork1 -DiscoverySNPCallerPlugin -i
mergedTBT/myStudy.tbt.byte -y -m topm/myMasterTags.topm -mUpd
topm/myMasterTagsWithVariants.topm -o hapmap/raw/myGBSGenos_chr+.hmp.txt
-mnF 0.8 -p myPedigreeFile.ped -mnMAF 0.02 -mnMAC 100000 -ref
MyReferenceGenome.fa -sC 1 -eC 10 -endPlugin -runfork1
```

```
Rajneeshs-MacBook-Pro:GBSpractice RajneeshPaliwal$ perl
./tassel4.0_standalone/run_pipeline.pl  -Xmx6g -fork1 -
DiscoverySNPCallerPlugin -i ./06_mergedTBT/ILRI_modifyTBT.h5 -m
./04_topm/ILRI_mergeMasterTagCounts.topm -mUpd
./04_topm/ILRI_mergeMasterTagCountsWithVariants.topm -o
./07_hapmapRaw/ILRI_hapmaprawSNP_chr+.hmp.gz -mnF 0.8  -mnMAF 0.02 -mnMAC
100000 -ref ./Maize_ref_chr9_10_ILRI/ZmB73_RefGen_v2_chr9_10_1st20MB.fasta -
sC 9 -eC 10 -endPlugin -runfork1 | tee -a ./Logfile/DiscoverySNPcallerPlugin
```

Gory Details:

In this step, a multiple sequence alignment is created for each "TagLocus" which is defined as a set of tags that align to the exact same genomic position and strand. The genomic position of a tag is defined by that of the first base on its barcoded end (after removing the barcode). SNPs are called from each TagLocus alignment. Tags with multiple or unknown physical genomic positions are not used for SNP calling. The SNP calls from each TagLocus are written to a genotype file in HapMap format, with one HapMap file produced per chromosome.

With the **-o (output file) option**, you must provide the relative path and "generic" name of the output HapMap genotype file. This filename must include the wildcard character '+' in place of the chromosome number. For example, if you use the argument:

```
-o hapmap/raw/myGBSGenos_chr+.hmp.txt
```

then the '+' character will be replaced by each chromosome number (from -sC to -eC) in the output files. If you

only want genotypes for one chromosome (*e.g.*, chromosome 9), then the **-sC (start chromosome) and -eC (end chromosome) options** should both be the same (*e.g.*, `-sC 9 -eC 9`). If you use '.gz' as the suffix at the very end of the output file name, then the output HapMap file will be gzip compressed. Note that *.hmp.txt.gz files can be directly read by the Tassel GUI (you do not have to decompress them first), or by subsequent steps in the GBS pipeline (as long as you include the '.gz' in the generic input file name).

If you are working with highly homozygous inbred lines or a selfing species, then be sure to use the **-mnF (minimum F) option** (we suggest setting mnF to 0.8 or 0.9), where 'F' means 'inbreeding coefficient', and is calculated for each SNP as:

`F = 1 - Ho/He,`

where `Ho` = observed heterozygosity, and

`He` = expected heterozygosity = `2p(1-p)`, where `p` = the frequency of the major allele.

SNPs with a calculated F less than -mnF will be removed from the output. In species like maize which contain abundant paralogs (from ancient chromosomal duplications), this can filter out numerous bad SNPs.

If you are NOT working with inbred lines or a selfing species, then invoke the -mnF option with a low cutoff such as -0.1 (use double quotes to specify a negative number: -mnF "-0.1").

If the samples in your study (discovery build) are a mixture of inbred lines and outbred material, then you can use a **pedigree file** (**-p option**) to specify which samples are inbred. In that case, when applying the -mnF cutoff, only the samples with an expected F in the pedigree file that is greater than or equal to the value specified by the -mnF (minimum F) option will be used in the calculation of F for each SNP (for comparison to the cutoff set by -mnF). For more information on the format and content of a pedigree file, see Appendix 2.

The options **-mnMAF** (minimum minor allele frequency) and **-mnMAC** (minimum minor allele count) can be used to filter out SNPs with rare minor alleles that possibly result from sequencing errors. Keep in mind that SNPs that pass *either* of these criteria will be output. If you are working with a biparental family with 1:1 segregation you might try a mnMAF of 0.2 and an impossible to reach mnMAC much larger your total number of taxa, so that it is irrelevant (in that case, only the mnMAF will matter). With unrelated individuals and no expected range of acceptable minor allele frequencies, you might want to try a mnMAF of 0.02 (and an impossible to reach mnMAC much larger than your total number of taxa).

The **-mnLCov (minimum locus coverage)** option can be used to filter out SNPs with very high amounts of missing data from the output. "Locus Coverage" is the proportion of taxa (samples) that are covered by at least one of the tags comprising the TagLocus to which a SNP belongs. If the coverage at a TagLocus is less than that specified by the -mnLCov option, then none of the SNPs in that TagLocus will be output. TagLoci with high amounts of missing data most likely result from large restriction fragments (>400 bp) that are not amplified as efficiently in the PCR steps of the GBS protocol. The default value of -mnLCov is 0.1. If you want fewer SNPs, but those with higher coverage, then increase -mnLCov.

The **HapMap genotype files** that we generate save disk space and memory by using single letters to represent phase unknown, diploid genotypes. Heterozygotes are represented by IUPAC nucleotide codes:

A = A/A
C = C/C
G = G/G
T = T/T
M = A/C
R = A/G
W = A/T
S = C/G
Y = C/T
K = G/T
N = missing data

Step11: MergeDuplicateSNPsPlugin

Summary:
Finds duplicate SNPs in the input HapMap file, and merges them if they have the same pair of alleles (not necessarily in the same major/minor order) and if their mismatch rate is no greater than the threshold specified by **-maxMisMat**. If **-callHets** is on, then genotypic disagreements will be called heterozygotes; otherwise they will be set to missing (callHets is off by default).

Input:
- HapMap genotype files (.hmp.txt or .hmp.txt.gz). Use a plus sign (+) as a wild card character to specify multiple chromosome numbers (each chromosome in a separate file).

Output:
- HapMap genotype files (.hmp.txt or .hmp.txt.gz) (one per chromosome) in which duplicate SNPs have been merged

Arguments:

MergeDuplicateSNPsPlugin	
-hmp	Input HapMap genotype file(s) (.hmp.txt or .hmp.txt.gz). Use a plus sign (+) as a wildcard character to specify multiple chromosome numbers (each chromosome in a separate file).
-o	Output HapMap genotype file(s) (.hmp.txt or .hmp.txt.gz). Use a plus sign (+) as a wildcard character to specify multiple chromosome numbers (each chromosome in a separate file). If you use a ".gz" suffix at the very end of the filename, the output genotype files will be gzip compressed.
-misMat	Threshold genotypic mismatch rate above which the duplicate SNPs won't be merged. Default: 0.05
-p	Optional pedigree file containing full sample names & expected inbreeding coefficient (F) for each. Only taxa (samples) with expected F >= 0.8 (*i.e.*, S3 or more) will be used to test if two duplicate SNPs agree with each other. See Appendix 2 for an example pedigree file. Default: use ALL taxa to compare duplicate SNPs.
-callHets	When two genotypes at a replicate SNP disagree for a taxon, call it a heterozygote. Defaults to false (=set to missing).
-kpUnmergDups	When a pair of duplicate SNPs are not merged (because they have different alleles, too many mismatches, or the major or minor allele for one of them is a gap), keep them. Defaults to false (=delete them).
-sC	Start chromosome. Must be an integer.
-eC	End chromosome. Must be an integer.

Example command:
```
/programs/tassel/run_pipeline.pl -fork1 -MergeDuplicateSNPsPlugin
-hmp hapmap/raw/myGBSGenos_chr+.hmp.txt -o
hapmap/mergedSNPs/myGBSGenos_mergedSNPs_chr+.hmp.txt -misMat 0.1 -p
myPedigreeFile.ped -callHets -sC 1 -eC 10 -endPlugin -runfork1
```

```
Rajneeshs-MacBook-Pro:GBSpractice RajneeshPaliwal$ perl
./tassel4.0_standalone/run_pipeline.pl  -Xmx6g -fork1 -
MergeDuplicateSNPsPlugin -hmp
./07_hapmapRaw/ILRI_hapmaprawSNP_chr+.hmp.gz.hmp.txt -o
./08_hapmapMergedSNPs/ILRI_mergedSNP_chr+.hmp.txt -misMat 0.1 -callHets  -sC
```

```
9 -eC 10 -endPlugin -runfork1 | tee -a ./Logfile/MergedSNPcallerPlugin
```

Gory Details:
This step is usually run directly after DiscoverySNPCallerPlugin, using the HapMap file(s) from that step as input. Duplicate SNPs arise from overlapping, but separate TagLoci that cover the same SNP. These overlapping TagLoci are usually on different strands, starting on either end of a restriction fragment that is less than 128 bp in length.

If the germplasm is not fully inbred, and still contains residual heterozygosity (like the maize NAM or IBM populations do) then **-callHets** should be on and **-maxMisMat** should be set fairly high (0.1 to 0.2, or even higher, depending on the amount of heterozygosity). Because the sequencing coverage is usually less than 1x, most of the time only one allele at a heterozygous SNP will be detected (particularly for *Ape*KI). Hence, duplicate SNPs genotypes from a true heterozygote may disagree simply because different alleles were sampled by the duplicate assays. Hence, these disagreements are not necessarily errors, and should not necessarily be used to prevent duplicate SNPs from being merged (unless your germplasm *is* highly inbred, with very little residual heterozygosity).

Indels (gaps) are ignored by this plugin: it makes no attempt to merge apparent duplicate sites with the same chromosomal position where either the major or minor allele is a gap.

STEP 12: GBSHapMapFiltersPlugin

GBSHapMapFiltersPlugin

Summary:
Reads HapMap format genotype files (one per chromosome) and filters out SNPs with low taxon coverage (missing data at most taxa), high heterozygosity, low (and/or high) minor allele frequency, or that are not in LD with at least one neighboring SNP. Taxa with low SNP coverage (missing data at most SNPs) can also be removed. All filters are off by default and all cutoffs are adjustable.

Input:
- HapMap genotype file(s) (.hmp.txt or .hmp.txt.gz)

Output:
- HapMap genotype file(s) (.hmp.txt or .hmp.txt.gz) with some SNPs and/or taxa filtered out

Arguments:

GBSHapMapFiltersPlugin	
-hmp	Input HapMap file(s) (.hmp.txt or .hmp.txt.gz). Use a plus sign (+) as a wild card character to specify multiple chromosome numbers (each chromosome in a separate file).
-o	Output HapMap file(s) (.hmp.txt or .hmp.txt.gz). Use a plus sign (+) as a wildcard character to specify multiple chromosome numbers (each chromosome in a separate file). If you use a ".gz" suffix at very end of the filename, the output genotype files will be gzip compressed.
-mnTCov	Minimum taxon coverage. The minimum SNP call rate for a taxon to be included in the output, where call rate is the proportion of the SNP genotypes for a taxon that are not "N" (where N = missing). Default: no filter.
-mnScov	Minimum site coverage. The minimum taxon call rate for a SNP to be included in the output, where taxon call rate is the proportion of the taxa with genotypes that are not "N" for that SNP (where N = missing). Default: no filter
-mnF	Minimum value of F (inbreeding coefficient). Not tested by default.
-p	Optional pedigree file containing full sample names & expected inbreeding coefficient (F) for each. Only taxa (samples) with expected F >= mnF used to calculate F (= 1-Ho/He) when applying the -mnF filter. See Appendix 2 for an example pedigree file. Default: use ALL taxa to calculate F.
-mnMAF	Minimum minor allele frequency Default: 0.0 (no filtering).
-mxMAF	Maximum minor allele frequency. Default: 1.0 (no filtering).
-hLD	Specifies that SNPs should be filtered for those in statistically significant LD with at least one neighboring SNP. Default: Off.
-mnR2	Minimum R-square value for the LD filter. Default: 0.01
-mnBonP	Minimum Bonferroni-corrected p-value for the LD filter. Default: 0.01
-sC	Start chromosome. Must be an integer.
-eC	End chromosome. Must be an integer.

Example command:
```
/programs/tassel/run_pipeline.pl -fork1 -GBSHapMapFiltersPlugin
-hmp hapmap/mergedSNPs/myGBSGenos_mergedSNPs_chr+.hmp.txt -o
hapmap/filt/myGBSGenos_mergedSNPsFilt_chr+.hmp.txt -mnTCov 0.01 -mnSCov 0.2 -
```

```
mnMAF 0.01 -hLD -mnR2 0.2 -mnBonP 0.005 -sC 1 -eC 10 -endPlugin -runfork1

Rajneeshs-MacBook-Pro:GBSpractice RajneeshPaliwal$ perl
./tassel4.0_standalone/run_pipeline.pl  -Xmx6g -fork1 -
GBSHapMapFiltersPlugin -hmp
./08_hapmapMergedSNPs/ILRI_mergedSNP_chr+.hmp.txt -o
./09_hapMapfilter/ILRI_hapmapfilterSNP_chr+.hmp.txt -mnTCov 0.01 -mnSCov 0.2
-mnMAF 0.01 -hLD -mnR2 0.2 -mnBonP 0.005 -sC 9 -eC 10 -endPlugin -runfork1 |
tee -a ./Logfile/HapmapfilterSNPcallerPlugin
```

Gory Details:

The **-mnTCov** and **-mnSCov options** allow you to filter out taxa and/or SNPs, respectively, with call rates lower than the specified cutoffs. These filters are off by default. If the -mnTCov (taxon filter) is invoked, it is applied first, so that taxa with very low call rates (*i.e.*, blanks and/or failed samples) are removed prior to applying any of the other filters. Taxa with low call rates are identified based only on the starting chromosome (specified by the -sC option), and then this same set of low call rate taxa is filtered from the output of all the chromosome. This is done to avoid the possibility of output genotype files for different chromosomes containing different sets of taxa, which could happen if some taxa hover above or below the -mnTCov cutoff on different chromosomes.

Filtering based on the **-mnF (minimum F) option**, and the optional use of the **-p (pedigree file) option** to specify which samples should be included in the calculation of F, are the same as described above for the DiscoverySNPCallerPlugin.

The **-mnMAF and -mxMAF options** allow you to select for those SNPs whose minor allele frequencies fall into an expected range. For example, if you are working in a backcross (or psuedo-testcross) family, with an expected minor allele frequency (MAF) of 0.25, you might set the -mnMAF at 0.15 and the -mxMAF at 0.35.

If your study samples are from a single, biparental cross (or from another type of population in which LD is fairly extensive along a chromosome), then the **-hLD (high LD) filter** (off by default) can be very useful to filter out bad SNPs with high genotyping error or incorrect physical genomic positions. If you invoke the -hLD filter, the cutoff minimum R^2 and Bonferroni-corrected *p*-value can be adjusted using the **-mnR2 and -mnBonP options** (both of these default to 0.01). To pass through the LD filter, a SNP must be in statistically significant LD (Bonferroni corrected *p*-value less than that specified by the -mnBonP option) with at least one SNP that is a minimum of 128 bp away (*i.e.*, not from the same TagLocus or cut site) but within a window of 50 SNPs on either side. In Tassel3, **the LD filter only works properly for inbred lines** (*e.g.*, RILs). This will be fixed in Tassel4, so that the LD filter can be applied to outbred (highly heterozygous) populations as well.