GBS SNP Calling Reference Optional Pipeline (GBS-SNP-CROP)

User Manual v1.0

Arthur T. O. Melo, Radhika Bartaula, and Iago Hale

Department of Biological Sciences, College of Life Science and Agriculture, University of New Hampshire, Durham, NH, USA.

December 2015

Table of Contents

INTRODUCTION

The GBS SNP Calling Reference Optional Pipeline (GBS-SNP-CROP) is executed via a sequence of seven Perl scripts which integrate custom parsing and filtering procedures with well-known, vetted bioinformatic tools, giving the user full access to all intermediate files. By employing a novel strategy of SNP calling based on the correspondence of within-individual to across-population patterns of polymorphism, the pipeline is able to identify and distinguish high-confidence SNPs from both sequencing and PCR errors. The pipeline adopts a clustering strategy to build a population-tailored "Mock Reference" using the same GBS data for downstream SNP calling and genotyping. Designed for libraries of paired-end (PE) read of arbitrary lengths, GBS-SNP-CROP maximizes data usage by eliminating unnecessary data culling due to imposed length uniformity requirements. GBS-SNP-CROP is a complete bioinformatics pipeline developed primarily to support curation, research, and breeding programs wishing to utilize GBS for the cost-effective genome-wide characterization of plant genetic resources, mainly in the absence of a reference genome. The pipeline, however, can also be used when a reference genome is available, either as a standalone analysis or as a complement to reference-based analyses via alternative pipelines (e.g. TASSEL-GBS) or indeed its own reference-independent analysis.

HOW TO CITE

Melo et al. (2015) GBS-SNP-CROP: A reference-optional pipeline for SNP discovery and plant germplasm characterization using genotyping-by-sequencing data. BMC Bioinformatics. DOI XX.

PIPELINE WORKFLOW

GBS-SNP-CROP is accomplished via the following series of seven steps (i.e. Perl scripts), functionally grouped into four main stages of analysis:

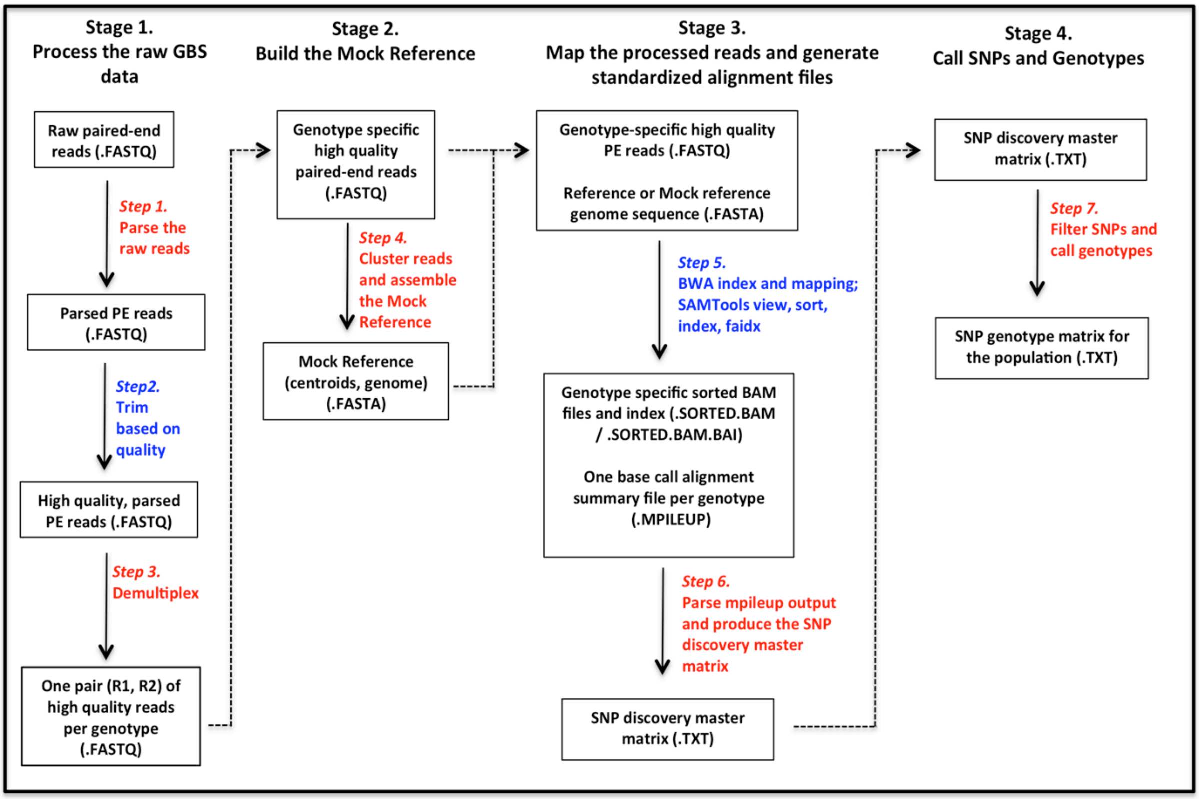
Stage 1. Process the raw GBS data
Step 1: Parse the raw reads
Step 2: Trim based on quality
Step 3: Demultiplex

Stage 2. Build the Mock Reference
Step 4: Cluster reads and assemble the Mock Reference

Stage 3. Map the processed reads and generate standardized alignment files
Step 5: Align with BWA-mem and process with SAMtools
Step 6: Parse mpileup output and produce the SNP discovery master matrix

Stage 4. Call SNPs and Genotypes
Step 7: Filter SNPs and call genotypes

On the following page is a schematic of the workflow, with inputs and outputs indicated for each step (arrow). Red steps (1, 3, 4, and 7) are executed via custom Perl scripts, while blue steps (2 and 5) call upon familiar bioinformatic tools.


Stage 1.
Process the raw GBS data
Raw paired-end reads (.FASTQ)
Step 1.
Parse the raw reads
Parsed PE reads (.FASTQ)
Step2.
Trim based on quality
High quality, parsed PE reads (.FASTQ)
Step 3.
Demultiplex
One pair (R1, R2) of high quality reads per genotype (.FASTQ)
Stage 2.
Build the Mock Reference
Genotype specific high quality paired-end reads (.FASTQ)
Step 4.
Cluster reads and assemble the Mock Reference
Mock Reference (centroids, genome) (.FASTA)
Stage 3.
Map the processed reads and generate standardized alignment files
Genotype-specific high quality PE reads (.FASTQ)
Reference or Mock reference genome sequence (.FASTA)
Step 5.
BWA index and mapping; SAMTools view, sort, index, faidx
Genotype specific sorted BAM files and index (.SORTED.BAM / .SORTED.BAM.BAI)
One base call alignment summary file per genotype (.MPILEUP)
Step 6.
Parse mpileup output and produce the SNP discovery master matrix
SNP discovery master matrix (.TXT)
Stage 4.
Call SNPs and Genotypes
SNP discovery master matrix (.TXT)
Step 7.
Filter SNPs and call genotypes
SNP genotype matrix for the population (.TXT)

REPOSITORY AND DISCUSSION GROUP/FORUM

The current version of GBS-SNP-CROP is deposited on Github at:
https://github.com/halelab/GBS-SNP-CROP.git

and we have a Google group discussion forum for any suggestions and questions at:
https://groups.google.com/forum/#!forum/gbs-snp-crop

STAGE 1. PROCESS THE RAW GBS DATA

Step 1: Parsing the raw reads

Perl script:
GBS-SNP-CROP-1.pl

Arguments (all required):

Flag	Explanation
-b	Barcode-ID file name (.txt file). See Appendix A
-fq	FASTQ file name seed (string). Ex: *FileNameSeed*_R1_001.fastq.gz
-s	Start number of FASTQ files (numeric). Ex: FileNameSeed_R1_00*1*.fastq.gz
-e	End number of FASTQ files (numeric). Ex: FileNameSeed_R1_0*34*.fastq.gz
-enz1	Enzyme 1 restriction site residue sequence (string). Ex: *TGCA* (PstI)
-enz2	Enzyme 2 restriction site residue sequence (string). Ex: *CGG* (MspI)

UNIX command example:

perl /*path to GBS_SNP_CROP*/GBS-SNP-CROP-1.pl –b barcodesID.txt –fq *FileNameSeed* –s 1 –e 34 –enz1 TGCA –enz2 CGG

FASTQ file naming convention:

*FileNameSeed*_R1_001.fastq.gz	*FileNameSeed*_R2_001.fastq.gz
*FileNameSeed*_R1_002.fastq.gz	*FileNameSeed*_R2_002.fastq.gz
*FileNameSeed*_R1_003.fastq.gz	*FileNameSeed*_R2_003.fastq.gz

etc.

Inputs:

- CASAVA-generated paired-end (R1, R2) files (.fastq.tar.gz)
- Barcode-ID file (.txt)

Outputs:

Output files are placed in the following four directories:

./parsed:	Parsed paired-end fastq files, with barcode-annotated headers
./singles:	Parsed, unpaired R1 reads, with barcode-annotated headers
./summaries:	Text files containing parsing summary information
./distribs:	Text files containing read length distribution summaries

Details:

The code associated with Step 1 is compatible with Illumina1.8+ sequencing data, where the input files are assumed to be CASAVA-processed, paired-end (i.e. R1 and R2), compressed (*.gz) FASTQ files. As per the protocol developed by Poland et al. (2012), these FASTQ files are assumed to contain multiplexed reads from a barcoded library of genotypes. To execute this stage of the pipeline, an auxiliary text file is required that associates each barcode with its corresponding genotype ID (see example "Barcode-ID" file in Appendix A). The script for Step 1 processes the raw reads in a relatively standard manner, beginning by searching the R1 read for a high-confidence barcode sequence (i.e. no more than one mismatch, relative to the provided list of barcodes) immediately preceding the expected cut site remnant of the less frequent cutter. If both barcode and cut site are found, they are trimmed from the read, the barcode is appended to the header of both the R1 and R2 reads, and the pair is retained for further processing. This first parsing script then searches for the 3'-ends of the GBS fragment, indicated by the in-line presence of the Illumina common adapter, coupled with the appropriate cut site residue. If found, the reads are truncated appropriately. Finally, all reads consisting of a majority of uncalled bases (i.e. N's) are discarded.

Step 2: Trim based on quality

Perl script:
GBS-SNP-CROP-2.pl

Arguments (all required):

Flag	Explanation
-fq	FASTQ file name seed (string). Ex: *FileNameSeed*_R1_001.fastq.gz
-t	Number of threads used by Trimmomatic (numeric)
-ph	Trimmomatic Phred scale: 33 or 64 (numeric)
-l	Trimmomatic LEADING parameter value (numeric)
-sl	Trimmomatic SLIDINGWINDOW parameter values (colon-separated numeric)
-tr	Trimmomatic TRAILING parameter value (numeric)
-m	Trimmomatic MINLEN parameter value (numeric)

UNIX command example:

```
perl /path to GBS_SNP_CROP/GBS-SNP-CROP-2.pl –fq L001 –t 10 –ph 33 –l 30 –sl 4:30 –tr 30 –m 32
```

Inputs:
- Parsed PE read file pair (.fastq), created in Step 1.

Outputs:
- High quality, parsed files of PE reads (.fastq)
- High quality, parsed files of orphan reads/singletons (.fastq)

Examples:

*FileNameSeed*_PE_R1parsed.fastq   *FileNameSeed*_PE_R2parsed.fastq
*FileNameSeed*_SR_R1parsed.fastq   *FileNameSeed*_SR_R2parsed.fastq

Requirements:

Java 7 or higher
Trimmomatic (Bolger et al., 2014)

Details:

Trimmomatic performs a variety of useful trimming tasks for Illumina paired-end data, of which only a subset are called upon by GBS-SNP-CROP. What follows is a description of the parameters provided to Trimmomatic via the command line:

LEADING: Trim bases from the beginning of a read, if below the specified quality threshold

SLIDINGWINDOW: Scan the read with a sliding window and truncate once the average quality within the window falls below the specified threshold.

TRAILING: Trim bases from the end of a read, if below the specified quality threshold

MINLEN: Cull a trimmed read entirely if its length is below a specified minimum

More information about Trimmomatic, including detailed descriptions of parameters, can be found at http://www.usadellab.org/cms/?page=trimmomatic

Please note that users may use read-cleaning bioinformatics tools other than those specified in the Steps 1 and 2 above. In this case, we encourage users to concatenate all surviving R1 reads into a single FASTQ file per library/lane, and the same for the corresponding R2 reads, because Step 3 (Demultiplex) requires this arrangement.

Step 3: Demultiplex

Perl script:
GBS-SNP-CROP-3.pl

Arguments (all required):

Flag	Explanation
-b	Barcode-ID file name (.txt file). See Appendix A
-fq	FASTQ file name seed (string). Ex: *FileNameSeed*_PE_R1parsed.fastq

UNIX command example:

```
perl /path to GBS_SNP_CROP/GBS-SNP-CROP-3.pl –b barcodesID.txt –fq FileNameSeed
```

Inputs:
- Barcode-ID file (.txt)
- One pair (R1, R2) of high-quality files (.fastq) per library, created in Step 2.

Outputs:
- One pair (R1, R2) of high-quality files (.fastq) per genotype, with all reads associated with that genotype placed inside a daughter directory named ./demultiplexed

Examples:

TaxaName1.R1.fastq	*TaxaName1*.R2.fastq
TaxaName2.R1.fastq	*TaxaName2*.R2.fastq
TaxaName3.R1.fastq	*TaxaName*3.R2.fastq

Details:

This script creates a separate pair of FASTQ files (R1 and R2) for each genotype. Specifically, the script processes each entry of the high quality PE files, searches for the barcode appended to each header, and writes the reads for each index to its own genotype-specific FASTQ file. The final output from this step is a separate pair of FASTQ files (R1 and R2) for each genotype, containing all reads associated with that genotype, and featuring the ID name in the filename.

STAGE 2. BUILD THE MOCK REFERENCE

Step 4: Cluster reads and assemble the Mock Reference

Perl script:
GBS-SNP-CROP-4.pl

Arguments (all required):

Flag	Explanation
-b	Barcode-ID file name (.txt file). See Appendix A
-rl	Raw GBS read lengths (numeric). Ex: *100* bp, *150* bp
-pl	Minimum length required after merging to retain read (numeric)
-p	p-value for PEAR (Zhang et al. 2014) (numeric)
-id	Nucleotide identity value required for USEARCH read clustering (numeric)
-t	Number of threads dedicated to USEARCH clustering (Edgar, 2010) (numeric)
-MR	Mock Reference name (string)

UNIX command example:

```
perl /path to GBS_SNP_CROP/GBS-SNP-CROP-4.pl –b barcodeID.txt –rl 150 –pl 32
–p 0.01 –id 0.93 –t 10 –MR MockRefName
```

Inputs:
- Barcode-ID file (.txt)
- Genotype-specific high quality PE files (.fastq), created in Step 3.

Outputs:
- A daughter directory named ./fastqForRef which contains the PEAR-assembled and manually-stitched FASTQ files used for building the Mock Reference
- Two different Mock Reference FASTA files:
  - *MockRefName*.MockRef_Genome.fasta (used for alignment)
  - *MockRefName*.MockRef_Clusters.fasta (used for optional downstream analyses)

Requirements:

PEAR (Zhang et al., 2014)
USEARCH (Edgar, 2010)

Details:

If a suitable reference genome is available for the target population, one may move directly to Stage 3 of the pipeline, skipping Stage 2 entirely; although it may be worth conducting a Mock Reference analysis in any case, to complement the results of reference-based approaches. If a reference is unavailable, however, Stage 2 is necessary. In it, the parsed and quality-filtered reads from Stage 1 are used to build a GBS-specific, reduced-representation reference (Mock Reference) to enable GBS read mapping and facilitate SNP discovery.

First, the pipeline calls upon the PEAR software package to merge the processed paired-end reads into single reads spanning the full GBS fragment lengths, wherever sequence overlap is sufficient (≥ 10 bp) between the paired reads to justify merging. For each genotype designated in the Barcode-ID file to contribute to the Mock Reference assembly (see Appendix A), this step generates four different FASTQ files: An “assembled” file, containing successfully merged reads; two “unassembled” files (R1 and R2), comprised of sequentially paired R1 and R2 reads which could not be merged; and a “discarded” file, containing assembled reads that failed to pass PEAR's user-specified minimum sequence length requirement.

Second, the pipeline stitches together all unmerged reads by joining pairs of sufficiently long "unassembled" R1 and R2 sequences together with an intermediate run of 20 high-quality A's, thus producing a FASTQ file of "stitched" R1+R2 reads:

```
@header with barcode
R1seq+AAAAAAAAAAAAAAAAAAAA+R2seq
+
R1quality+IIIIIIIIIIIIIIIIIIII+R2 quality
```

Representing the reduced genomic space targeted by the GBS restriction protocol, these PEAR-assembled and manually-stitched reads are then concatenated into a single FASTQ file per genotype.

Third, GBS-SNP-CROP calls upon the USEARCH software package to cluster these "assembled" and "stitched" reads based on a user-specified similarity threshold, thereby producing a reduced list of non-redundant consensus sequences (centroids) that span the GBS fragment space. To accomplish this, the USEARCH clustering procedure is executed first within each selected genotype and subsequently, if more than one genotype is selected to build the Mock Reference, across all selected genotypes. It is this final list of non-redundant clusters that are linked together, end-to-end, to constitute the Mock Reference.

STAGE 3. MAP THE PROCESSED READS AND GENERATE STANDARDIZED ALIGNMENT FILES

Step 5: Align with BWA-mem and process with SAMtools

Perl script
GBS-SNP-CROP-5.pl

Arguments:

Flags	Explanation
-b	Barcode-ID file name (.txt file). See Appendix A
-ref	Reference FASTA file, either Mock Reference or true reference (file)
-Q	Phred score base call quality (numeric)
-q	Alignment quality (numeric)
-f	SAMtools flags controlled by small f (numeric)
-F	SAMtools flags controlled by CAPS F (numeric)
-t	Number of threads dedicated to BWA-mem mapping (numeric)
-Opt	If desired, any additional options for SAMtools view (string within “quotes”)

UNIX command example:
Using both “-f” and “-F” flags from SAMtools:

```
perl /path to GBS_SNP_CROP/GBS-SNP-CROP-5.pl –b barcodeID.txt –ref MockRefName.MockRef_Genome.fasta –Q 30 –q 0 –samf 2 –samF 2308 –t 10 –Opt “–m 1 –s 156”
```

If you do not wish to use some of the above parameters, simply set them to zero. For example, using only the “-Q”, “-f” and “-t” parameters:

```
perl /path to GBS_SNP_CROP/GBS-SNP-CROP-5.pl –b barcodeID.txt –ref MockRefName.MockRef_Genome.fasta –Q 30 –q 0 –samf 2  –samF 2308 –t 10 –Opt 0
```

Inputs:
- Barcode-ID file (.txt)
- Genotype-specific high quality PE files (.fastq), created in Stage 1.
- Reference genome or Mock Reference [genome], created in Stage 2 (.fasta).

Outputs:
- Aligned and filtered reads (.bam)
- Sorted BAM files (.sorted.bam)
- Indexed BAM files (.sorted.bam.bai)
- Indexed reference or Mock Reference (.fasta.idx)
- Base call alignment summary file (.mpileup)

Note: One complete set of the above output files is created for each genotype in the population.

Requirements:
BWA-mem (Li & Durbin 2009)
SAMtools (Li et al., 2009)

Details:

In this step, the BWA-mem algorithm (Li & Durbin 2009) is called upon to align the Stage 1 processed reads, genotype-by-genotype, to the reference/Mock Reference.

SAMtools (Li et al., 2009) is then used to accomplish the following steps: 1) Filter the mapped reads via SAMtools flags, retaining only those which map appropriately as pairs without potentially confounding secondary or supplementary alignments to the reference; 2) Convert the filtered SAM files to BAM files; 3) Index and sort the BAM files; and 4) Index the FASTA reference sequence. These five steps (BWA-mem alignment and the four SAMtools procedures) are carried out individually for each genotype.

After alignment, we encourage users to advance only those reads for further analysis that map in proper pairs (-f 2) with no supplementary alignments (-F 2308), criteria applied via SAMtools view flags. For more detailed explanation about SAMtools flags, please refer to https://broadinstitute.github.io/picard/explain-flags.html.

Finally, the SAMtools mpileup algorithm is used to make a liberal identification of all potential SNPs based on the individual alignments of each genotype considered separately, producing a base call alignment summary (mpileup file) for each genotype. In addition, this script calls upon SAMtools mpileup flags –B and –C50 to reduce false SNP calling caused by misalignment and delete reads with excessive mismatches, respectively.

Step 6: Parse mpileup output and produce the SNP discovery master matrix

Perl script:
GBS-SNP-CROP-6.pl

Arguments:

Flag	Explanation
-b	Barcode-ID (file). See Appendix A
-out	SNP discovery master matrix (tab-delimited .txt file).

UNIX command example:

perl /*path to GBS_SNP_CROP*/GBS-SNP-CROP-6.pl –b barcodeID.txt –out SNPs_master_matrix.txt

Inputs:

- Barcode_ID file (.txt)
- One base call alignment summary file (.mpileup) per genotype, created in Step 5

Outputs:

- One reduced base call alignment summary count file (.txt) per genotype
- The SNP discovery master matrix (.txt)

Details:

The Step 6 script scans the genotype-specific mpileup files from Step 5 and extracts the necessary information to create distilled "count" files, text files containing four essential tab-delimited columns: (1) Reference genome/chromosome identifier; (2) SNP position; (3) Reference base at that position; and (4) A comma-delimited string containing alignment information at that position (i.e. depths of A, C, G, and T reads). Example:

```
chr01        71019        A        15,0,2,0
chr01        135677       C        0,89,0,0
chr01        135714       T        20,0,0,115
```

During this parsing, all putative indels are rigorously detected and removed from the analysis; thus the resultant count files are strictly SNP-specific. Each genotype-specific count file is then parsed, such that only those rows containing reads polymorphic to the reference sequence are retained (e.g. the first and third rows in the example above). Once this is completed for each genotype separately, Step 6 proceeds by mining the full set of reduced genotype-specific count files to generate a single, non-redundant master list of all potential SNP positions throughout the target population. Comprehensive alignment information is then extracted from the original count files for each genotype for all potential SNP positions in the master list and the data organized into a "master matrix" for the population.

STAGE 4. CALLING SNPS AND GENOTYPES

Step 7: Filter SNPs and call genotypes

Perl script:
GBS-SNP-CROP-7.pl

Arguments:

Flag	Explanation
-in	SNP discovery master matrix input file (file). The output from last step
-out	SNP genotyping matrix for the population output file (file)
-mnHoDepth0	Minimum depth required for calling a homozygote when the alternative allele depth = 0 (numeric)
-mnHoDepth1	Minimum depth required for calling a homozygote when the alternative allele depth = 1 (numeric)
-mnHetDepth	Minimum depth required for each allele when calling a heterozygote (numeric)
-altStrength	Across the population for a given putative bi-allelic SNP, this alternate allele strength parameter is the minimum proportion of non-primary allele reads that are the secondary allele (numeric)
-mnAlleleRatio	Minimum required ratio of less frequent allele depth to more frequent allele depth (numeric)
-mnCall	Minimum acceptable proportion of genotyped individuals to retain a SNP (numeric)
-mnAvgDepth	Minimum average depth of an acceptable SNP (numeric)
-mxAvgDepth	Maximum average depth of an acceptable SNP (numeric)

UNIX command example:

```
perl /path to GBS_SNP_CROP/GBS-SNP-CROP-7.pl –in SNPs_master_matrix.txt
–out SNPs_genotyping_matrix.txt –mnHoDepth0 11 –mnHoDepth1 48 –mnHetDepth 3 –
altStrength 0.9 –mnAlleleRatio 0.1 –mnCall 0.75 –mnAvgDepth 4 –mxAvgDepth 200
```

Inputs:
- SNP discovery master matrix (.txt), created in Step 6

Outputs:
- Final SNP genotyping matrix (.txt)

Details:

In this step, the master matrix created in Step 6 is systematically pared down based on a series of SNP-culling filters to arrive at a final "SNP genotyping matrix" containing only high-confidence SNPs and genotypes. First, the master list of potential SNPs is parsed based upon a flat criteria of independence, namely that a SNP is retained for further consideration if and only if there exist independent reads of the putative secondary allele, at a specified minimum depth (-mnHetDepth), across at least three genotypes. This simple requirement for independent instances of the secondary allele is an essential strategy for minimizing false SNP declarations due to random sequencing and PCR errors, including strand bias errors.

GBS-SNP-CROP considers only potential bi-allelic SNPs (i.e. it excludes multi-allelic variants) by imposing a population-level allele frequency filter via a user-defined Alternative Allele Strength parameter (-altStrength). For each potential SNP position, this parameter considers the total read depth, across the whole population, of all four bases. A potential SNP is retained for further downstream analysis if and only if it is strongly bi-allelic, that is if:

$$\frac{2^{\circ}\,Allele\ Depth}{2^{\circ}\,Depth + 3^{\circ}\,Depth + \ 4^{\circ}\,Depth} > altStrength$$

For a tetraploid species, we suggest a minimum value of 0.90 for this parameter, though higher values may be imposed in the interest of stricter error control (see Appendix B).

For calling heterozygotes, a given genotype must have a user-specified minimum read depth for each allele (-mnHetDepth); and the depth ratio of the lower-depth to the higher-depth allele must exceed a specified ploidy-appropriate threshold (-mnAlleleRatio; see Appendix B). The GBS-SNP-CROP genotyping criterion for homozygotes is more stringent, requiring a relatively high, user-specified minimum depth secondary allele count is zero (-mnHoDepth0) and an even higher depth when the secondary allele count is one (-mnHoDepth1) in an effort to reduce the rate of erroneous calls (i.e. true heterozygotes called as homozygous due to sampling bias; see Appendix B).

Next, in an effort to retain only broadly informative SNPs, the matrix is further reduced such that all SNPs (i.e. rows) are discarded for which less than some minimum proportion of population (-mnCall) have genotype calls. Finally, users are able to retain only those SNPs with an average depth (D) within a specified range, as a means of deleting all markers with low coverage as well as those that are over-represented. In our analysis, we accept only those SNPs such that 4 < D < 200. The lower bound of this range is in agreement with our -mnHetDepth of 3, a bare minimum depth for genotyping with any confidence. The upper bound is a value commensurate with what "over-representation" means in our particular dataset, namely that roughly 5% of the detected SNPs had an average read depth > 200.

To facilitate the characterization of those high-confidence SNPs that satisfy the above filters and criteria, the final SNP genotyping matrix contains both summary statistics as well as complete genotype-specific alignment data for each retained SNP. The first ten columns of the matrix feature the following information: 1) Genome/chromosome identifier; 2) SNP position; 3) Reference base; 4) Average read depth at that SNP position across the population; 5) Primary allele (i.e. the most frequent allele at that position, based on read depth across the population); 6) Secondary allele (i.e. the less frequent allele at that position); 7) Percentage of individuals genotyped at that position; 8) Total number of potential homozygotes for the primary allele; 9) Total number of potential heterozygotes; and 10) Total number of potential homozygotes for the secondary allele. Columns 11 and higher contain the complete alignment data for each individual genotype for all possible SNP positions. The structure of the final SNP genotyping matrix is shown in the figure below:

Cells in columns 11 and higher contain genotype calls and genotype-specific alignment data, separated by a pipe ("|"). A failure to genotype is represented by a dash ("-").

Genome/Chromosome identifier	SNP position	Reference base	Average read depth	Primary allele	Secondary allele	% of individuals genotyped	Total number of primary homozygotes	Total number of heterozygotes	Total number of secondary homozygotes				
MR01	71019	A	8.66	A	G	100	33	17	0	A/A|8/0	A/A|32/0	A/G|13/49	...
MR01	135676	C	20.48	C	T	94	25	19	3	C/C|11/0	T/T|1/49	C/C|7/0	
MR01	135714	T	20.63	C	T	98	27	14	8	-|0/4	C/T|45/106	T/T|0/98	
MR01	307247	G	64.16	A	G	88	18	21	10	A/A|23/0	-|132/5	G/G|0/150	
...													...

DOWNSTREAM TOOL

Perl script:
GBS-SNP-CROP-8.pl

Function:
To transform the final SNP genotyping matrix into forms appropriate for downstream analyses via familiar software packages, specifically R (input matrix), TASSEL GUI (HapMap file), and/or PLINK (.tped file).

Arguments:

Flag	Explanation
-in	Final SNP genotyping matrix input file (file)
-out	Output label name without extension (string)
-b	Barcode-ID file name (.txt file). See Appendix A
-formats	The name(s) of the software packages (R, Tassel, Plink) for which an input-compatible file format should be created. If more than one format is desired, the names should be separated by commas without any space, as in the examples shown below. (string)

UNIX command example:
Creating input files for R, Tassel GUI, and PLINK

```
perl /path to GBS_SNP_CROP/GBS-SNP-CROP-8.pl –in SNPs_genotyping_matrix.txt  –out SNP_genotyping –b barcodeID.txt –formats R,Tassel,Plink
```

Creating an input file only for Tassel GUI

```
perl /path to GBS_SNP_CROP/GBS-SNP-CROP-8.pl –in SNPs_genotyping_matrix.txt  –out SNP_genotyping –b barcodeID.txt –formats Tassel
```

Inputs:
- Final SNP genotyping matrix (.txt), created in Step 7

Outputs:
- R-compatible final SNP genotyping matrix ("*GBS-SNP-CROP*_R_in.txt")
- Tassel GUI-compatible final SNP genotyping matrix ("*GBS-SNP-CROP*_Tassel_in.hmp.txt")
- PLINK-compatible final SNP genotyping matrix ("*GBS-SNP-CROP*_Plink_in.tped")

Details:

Based on the output formats specified by the user, this script converts the final SNP genotyping matrix from Step 7 into:

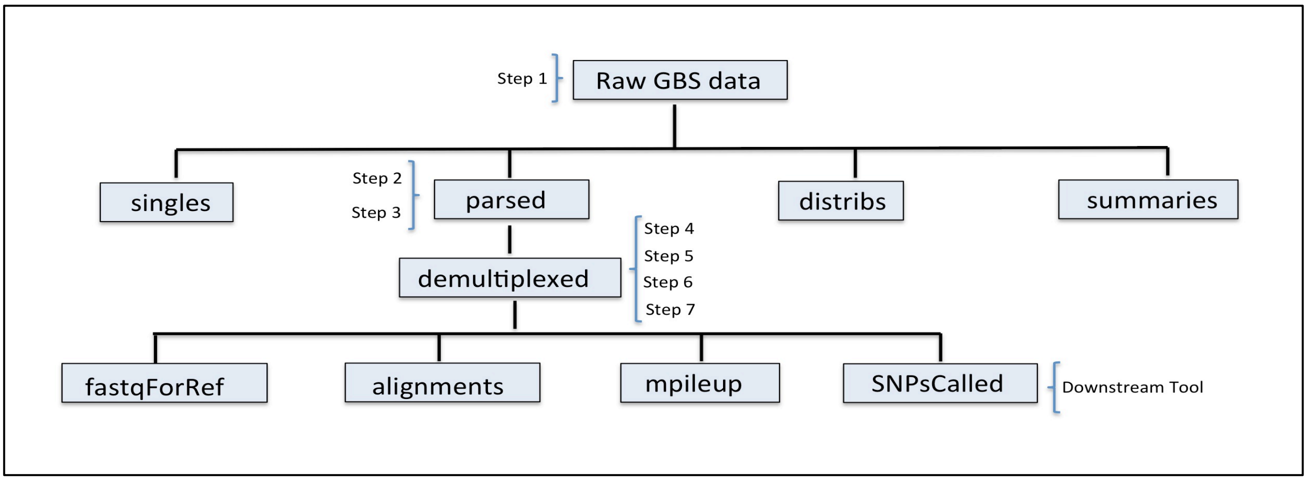
- A version appropriate for diversity analyses within R (R Development Core Team, 2013) (e.g. calculating distance metrics, generating cladograms, etc.), achieved by replacing primary homozygotes with 0, heterozygotes with 0.5, secondary homozygotes with 1, and unassignable genotypes with "NA";
- A HapMap file for use as input into Tassel GUI (Glaubitz et al., 2014), allowing users to easily access the functionality of that software package for forward analysis; and/or
- The transposed .tped file required by the whole genome association analysis toolset PLINK (Purcell et al., 2007).

DIRECTORY STRUCTURE CREATED BY GBS-SNP-CROP

The figure below shows the structure of all directories created by GBS-SNP-CROP over the course of the analysis and is intended to help users navigate efficiently during pipeline execution. Starting with the raw reads (fastq.gz files) in directory "Raw GBS data", users, Step 1 creates and populates several subdirectories: 1) The "parsed" directory will contain the parsed paired-end FASTQ files with barcode-annotated headers; 2) "singles" will contain the parsed, unpaired R1 reads, with barcode-annotated headers; 2) "distribs," will contain text files of read length distribution summaries; and 4) "summaries" will contain text files of parsing summary information from Step 1. After Step 1 is completed, users should move to the "parsed" directory to execute Steps 2 and 3.

Inside the "demultiplexed" directory, created in Step 3, users will run all subsequent Steps (4, 5, 6 and 7). Step 4 creates the FASTQ files (both PEAR-assembled + manually-stitched) used to assemble the Mock Reference and save them in the "fastqForRef" directory. Step 4 also creates two Mock References (*Cluster.fasta and *Genome.fasta) and stores them in the "demultiplexed" directory. Finally, Step 4 generates a log file (Pear.log) that contains all summary results from both PEAR (Zhang et al., 2014) assembly and manual read stitching; this files is also saved to the "demultiplexed" directory.

After mapping all genotype-specific high quality reads to the chosen reference sequence, Step 5 writes all alignment files (.BAM and .BAM.BAI) to the "alignments" directory. At the end of Step 6, after all mpileup files have been parsed and the SNP discovery master matrix created, the alignment summary (.mpileup) and count (*count.txt) files for each genotype can be found in the "mpileup" directory. Finally, in the end (Step 7), both the SNP master matrix and the SNP genotyping matrix can be found inside the "SNPsCalled" directory, within which the three optional downstream scripts may be executed.



REFERENCES

Bolger AM, Lohse M, Usadel B. Trimmomatic: A flexible trimmer for Illumina Sequence Data. Bioinformatics. 2014;30(15): 2114-2120.

Edgar RC. Search and clustering orders of magnitude faster than BLAST. Bioinformatics. 2010;26(19):2460-2461.

Glaubitz JC, Casstevens TM, Lu F, Harriman J, Elshire RJ, Sun Q, Buckler ES. TASSEL-GBS: A High Capacity Genotyping by Sequencing Analysis Pipeline. PLoS One. 2014; DOI: 10.1371/journal.pone.0090346.

Li H, Durbin R. Fast and accurate short read alignment with Burrows-Wheeler Transform. Bioinformatics. 2009; 25:1754-1760.

Li H, Handsaker B, Wysoker A, Fennell T, Ruan J, Homer J, et al. The Sequence Alignment/Map format and SAMtools. Bioinformatics. 2009;25 (16): 2078-2079.

Melo ATO, Bartaula R, Hale I. GBS-SNP-CROP: A reference-optional pipeline for SNP discovery and plant germplasm characterization using variable length, paired-end genotyping-by-sequencing data. BMC Bioinformatics. DOI XX.

Poland JA, Brown PJ, Sorrells ME, Jannink JL. Development of high-density genetic maps for barley and wheat using a novel two-enzyme Genotyping-by- Sequencing approach. PLoS One. 2012; doi:10.1371/journal.pone.0032253.

Purcell S, Neale B, Todd-Brown K, Thomas L, Ferreira MAR, Bender D, et al. PLINK: a toolset for whole-genome association and population-based linkage analysis. American Journal of Human Genetics. 2007;81(3):559-575.

R Development Core Team. R: a language and environment for statistical computing. R Foundation for Statistical Computing. 2013.

Zhang J, Kobert K, Flouri T, Stamatakis A. PEAR: a fast and accurate Illumina Paired-End reAd mergeR. Bioinformatics. 2014;30(5):614-20.

APPENDIX A: BARCODE-ID file example

The Barcode-ID file is a simple tab-delimited text file required by GBS-SNP-CROP for the purpose of associating barcodes with genotype names. This file also allows users to specify which individual(s) from the target population are to be used for building the Mock Reference.

The file itself contains only three tab-delimited fields per line (i.e. three columns), with one line needed for each unique genotype/barcode combination in the population. The first field (column) contains the 6-10 bp barcode sequences, or index; the second contains the genotype name associated with that barcode; and the third contains a simple flag ("YES" or "NO") indicating whether or not the genotype is to be used to build the Mock Reference. The file requires no header. Example of the appearance of the file in Excel:

Barcode	Genotype name	Used for building the Mock Reference?
TGACGCCA	Lib1_01	YES
CAGATA	Lib1_02	YES
GAAGTG	Lib2_05	YES
TAGCGGAT	Lib3_10	NO
TATTCGCAT	Lib3_11	YES

The final file, saved without a header as a tab-delimited text file:

```
TGACGCCA	Lib1_01		YES
CAGATA		Lib1_02		YES
GAAGTG		Lib2_05		YES
TAGCGGAT	Lib3_10		NO
TATTCGCAT	Lib3_11		YES
```

As discussed in the accompanying manuscript (Melo et al., 2015), users will need to complete Stage 1 of the pipeline to decide, based on the number of recovered reads per genotype, which individual(s) should be used to build the Mock Reference. In this situation, users should modify column 3 of the Barcode-ID file before running Step 4 of GBS-SNP-CROP. In other words, users can initialize column 3 with all "YES" flags in Stage 1 of the pipeline and then simply update the "YES/NO" column before Stage 2, using the result of Stage 1 to decide how many and which genotypes to use for building the Mock Reference.

APPENDIX B: Suggested parameter values for Script 7, based on ploidy scenarios and confidence considerations

Various user-defined parameters within Script 7 of the GBS-SNP-CROP workflow should be adjusted depending on the ploidy of the species under study (i.e. the number of independent copies of a SNP locus in the genome) and the desired level of error control of the user. The following rationale and the summary table at the end of this Additional file are intended to guide users in their selection of these values:

–mnHoDepth0 and –mnHoDepth1

Assuming random (i.e. non-biased) allele sampling during GBS library preparation and sequencing, the maximum probability for any given heterozygous locus that a sequenced GBS fragment will capture the primary allele is:

$$P(sequencing\ the\ primary\ allele) = \frac{p-1}{p}$$

where p is the ploidy of the species (i.e. p = 2 for a diploid, 4 for a tetraploid, 6 for a hexaploid, etc.). The probability that the primary allele will be sampled d (for *d*epth) independent times is therefore:

$$P(sequencing\ the\ primary\ allele\ d\ times) = \left(\frac{p-1}{p}\right)^d$$

Thus the probability that the alternate allele will be sampled at least *once* over d independent samples is:

$$P(sequencing\ the\ alternate\ allele\ at\ least\ once\ in\ d\ samples) = 1 - \left(\frac{p-1}{p}\right)^d$$

For any single genotype call (i.e. for any single SNP-accession combination), it reasonable that a user would want this probability of detecting the alternate allele, if it exists, to be at least 95%. For the tetraploid case (p = 4), if one wishes to call a homozygote with at 95% confidence, this requirement dictates a minimum read depth of at least:

$$d_{min} = \frac{\ln(1-0.95)}{\ln(\frac{p-1}{p})} = \frac{\ln(1-0.95)}{\ln(0.75)} = 10.4 \to 11$$

Calculated in this way, this minimum depth required for calling homozygotes should be considered an absolute minimum because it controls error only in the case of a single genotypic call, whereas actual GBS datasets may require >10^6 such calls, if not more. To control error across the entire set of such calls (i.e. to permit, say, only 1 erroneous homozygote declaration in 10^6 calls), the minimum required depth increases significantly:

$$d_{min} = \frac{\ln(10^{-6})}{\ln(0.75)} = 48$$

Which values are chosen for homozygote depth thresholds depends both on ploidy as well as on the user's attitude toward error control. For this manuscript (48 accessions of 4x kiwiberry), we set *–mnHoDepth* = 11 to call homozygotes in the absence of any reads of the alternative (i.e. secondary) allele but used the higher threshold (*-mnHoDepth1* = 48), commensurate with the size of our dataset, when calling a homozygote in the presence of a single read of the alternate allele, such higher error control being needed to dismiss this single read as an error.

-mnAlleleRatio

At low read depth (e.g. the d_{min} = 11 case above), the minimum acceptable ratio of the depth of the secondary (2°) allele to that of the primary (1°) allele (*-mnAlleleRatio*) can be inferred directly, namely:

$$2°{:}\,1°_{min} = \frac{1}{\mathrm{d}_{min} - 1} = \frac{1}{10} = 0.1$$

Simulations indicate that this cutoff ratio increases in value as overall read depth increases (e.g. for a tetraploid, this value increases to 0.25 [95% confidence] for a read depth of 200); therefore, the calculated value above should once again be considered an absolute minimum for users of this pipeline.

-altStrength

Another parameter with clear ploidy dependency is –*altStrength*, the minimum required proportion of secondary reads to all non-primary reads [i.e. 2° allele depth / (2° + 3° + 4° depths)]. In the tetraploid case, assuming an acceptable minimum 0.1 allele depth ratio as calculated above and an upper boundary on sequencing error of 10 errors per kbp [39,40], the suggested minimum value of –*altStrength* = 0.1 / (0.1 + 0.01) = 0.9.

With the above rationale and description as a guide, users will hopefully find the following table useful as a rough guide in setting appropriate values for GBS-SNP-CROP Script 7 parameters, based on their study and objectives:

	Error rate	0.05	0.01	0.001	0.0001	0.00001	0.000001	*-mnAlleleRatio (min values)*	*-altStrength (min values)*
	Confidence	95%	99%	99.9%	99.99%	99.999%	99.9999%		
ploidy (p)	2	5	7	10	14	17	20	0.25	0.962
	4	11	17	25	33	41	48	0.10	0.909
	6	17	26	38	51	64	76	0.063	0.862
	8	23	35	52	69	87	104	0.045	0.820

↑ *-mnHoDepth0*
(min values)

↑ *-mnHoDepth1*
(for datasets on order of 10^6 calls)