PLUMAGE v. 2.0 – pipeline for GBS error correction in Python3
DOCUMENTATION (DOC version 9/26/2013)

System requirements:
Python 3
Pandas and all dependencies (http://pandas.pydata.org/)

All Plumage scripts can be executed from the command line within a working directory containing PLUMAGE, using the following general command:

Python3 Plumage_[script name] parameters

See specific scripts for example calls. Note, at any time you can obtain a list of parameters and their descriptions by calling the script with the "-h" option, for example, the command <Python3 Plumage_1_filter_impuation_results.py –h> will return the parameters and their descriptions for Plumage_1.

1. Plumage_1_filter_imputation_results.py

Purpose: This script takes the outfile from Plaid Impute as input and removes all SNPs with accuracy scores lower than the "strin" parameter.

Parameters:

-i infile -- the absolute path/name of the input file, should be the output from plaid impute. Default is stdin.

-o outfile – the absolute path/name of the file you want your filtered imputation results written to. Default is stdout.

-s filter stringency – enter the confidence score below which imputable SNPs will be dropped. Scores range between 0 and 1 – 0 will keep all imputable SNPs, 1 will discard all SNPs with confidence scores lower than 100%. Default = 0.95.

example call:

python3 Plumage_1_filter_impuation_results.py –i
/Users/Bob/my_GBS_project/plaid_impute_output.txt –o
/Users/Bob/my_GBS_project/Plumage_1_output.txt

2. Plumage_2_sequence_error_correction.py

Purpose: This script takes the output from Plumage 1 and changes likely sequence errors to "NA". The script works by first identifying recombination breakpoints. For each chromosome, each breakpoint call is tested for whether or not it is likely to be an error. If the first three calls following a breakpoint call that are not on the same tag as the

breakpoint call are the same as the breakpoint call, then the call is left as it is. If any of the first three calls on different tags do not agree with the breakpoint call, then it is changed to "NA".

Parameters:

-i infile: absolute path/name of the file output from Plumage 1. Default is stdin.

-o outfile: the absolute file name you want the results of the script written to. Defaults to std.out

-p1 parent 1: this is the exact name of the data column containing your parent 1 calls. Required.

-p2 parent 2: this is the exact name of the data column containing your parent 2 calls. Required.

-e optional file path/name you want a report of the changes written to, defaults to no report written

example call:

python3 Plumage2_sequence_error_correction.py –i /Users/Bob/my_GBS_project/Plumage_1_output.txt -o /Users/Bob/my_GBS_project/Plumage_2_output.txt –p1 IR64 –p2 Azucena

3. Plumage_3_remove_failed_markers_and_individuals

Purpose: This script calculates the call rate for markers and individuals and drops all markers and/or individuals with call rates less than the respective thresholds – see parameters for details.

Parameters:

-i infile – the absolute path/name of the input file, usually the output of Plumage_2. Optional, defaults to std.in

--drop_m marker threshold -- marker call rate below which a marker is dropped from the dataset. Should be between 0 and 1, i.e., if you want all markers with 50% or more missing data dropped, then set drop_m to .5. Optional, defaults to .75

--drop_i individual threshold -- Proportion of missing data at and above which individuals should be dropped from the dataset. Enter a value between 0 and 1, optional, defaults to 1 (no individuals dropped). Script will print a list of dropped individuals.

-o outfile – the path/name of file you want results written to. Optional, defaults to std.out.

example call:

python3 Plumage_3_remove_failed_markers_and_indivudals.py –i /Users/Bob/my_GBS_project/Plumage_2_output.csv –o /Users/Bob/my_GBS_project/Plumage_3_output.csv

Example call, piping output from Plumage_2 into Plumage_3 (only one call necessary where two were before):

python3 Plumage_2_ sequence_error_correction.py –i /Users/Bob/my_GBS_Project/Plumage_1_output.txt –p1 IR64 –p2 Azucena | python3 Plumage_3_remove_failed_markers_and_indivudals.py –o /Users/Bob/my_GBS_Project/Plumage_3_output.txt

4. Plumage_4_add_reads_to_Panati_hmp

Purpose: This script takes the .vcf file generated by PANATI and uses the data in it to add a column to a PANATI hapmap containing the number of sequence reads covering each SNP. This information is useful for quality control and selecting high quality SNP subsets.

Parameters:

-v vcf file: the full file path/name for the .vcf file produced when running your data through PANATI. Required.

-i infile: the full file path/name of the PANATI hapmap to which you want to add a column for the number of reads/SNP. Optional, defaults to sys.stdin.

-o outfile: The file path/name you want the results written to. Optional, defaults to stdout

example call:
python3 Plumage_4_add_reads_to_Panati_hmp.py -v /Users/Bob/my_GBS_project/PANATI_results/IR64_Azucena.vcf -i /Users/Bob/my_GBS_project/Plumage_3_output.txt –o /Users/Bob/Plumage_4_output.txt

5. Plumage_5_transform_for_rqtl

Purpose: This script takes the standard PANATI hapmap and transforms it so it is formatted for use with r/qtl (of course, to do QTL mapping, you will have to manually insert your phenotype data).

Parameters:

-i infile – full path/name for data file you want to transform. Optional, defaults to stdin

-o outfile – full path/name for file you want results written to. Required.

-- has_rn specify this flag if your hapmap contains a column for read number

--no_cr specify this flag if your hapmap does NOT contain a column for call rate

-p1 parent 1 -- this is the exact name of the data column containing your parent 1 calls. Required.

-p2 parent 2 – this is the exact name of the data column containing your parent 2 calls. Required.

example call:

python3 Plumage_5_transform_for_rqtl.py –i /Users/Bob/my_GBS_project/Plumage_4_output.txt –o /Users/Bob/Plumage_4_output_rqtl_format.csv –has_rn –p1 IR64 –p2 Azucena

6. Plumage_6_choose_binned_set_from_reads

Purpose: This script chooses a subset of SNPs for QTL mapping based on a binning parameter. The SNP with the highest number of reads will be chosen from each bin to make up the subset.

Parameters

-i infile – the path/name of the file containing a column for read number from which you want to select a subset of SNPs. Optional, defaults to stdin.

-o outfile – the path/name of the file you want the results written to. Required.

-b bin size – number of base pairs you want between selected SNPs. Optional, defaults to 240000 (= 240Kb, or 1cM in rice)

example call:

python3 Plumage_6_choose_binned_set_from_reads.py –i /Users/Bob/my_GBS_project/Plumage_4_output.txt –o /Users/Bob/Plumage_4_output_subset_240kb_binned.txt

7. Plumage_7_enrich_specific_locations

Purpose: This script can be used to pull out all SNPs that are in one or more specific location ranges from a larger hapmap.

Parameters:

-i infile: path/name of file containing full SNP dataset. Optional, defaults to stdin.

-k location key: a tab delimited file containing three columns that describe your region(s) of interest: "chrom", "pos_start", and "pos_end" – note, you MUST use these header names! For each region that you want to pull SNPs for, fill in the chromosome in the "chrom" column, the start of the position range in base pairs in the "pos_start" col, and position range end in base pairs in the "pos_end" column. Required.

-o outfile: path/name of file you want results written to. Optional, defaults to stdout.

example call:

python3 Plumage_7_enrich_specific_locations.py –i /
Users/Bob/my_GBS_project/Plumage_4_output.txt
–k /Users/Bob/my_GBS_project/my_regions_of_interest.txt –o
/Users/Bob/my_GBS_project/Plumage_4_output_regions_of_interest.txt

8. Plumage_8_get_genome_percentages

Purpose: For each individual in a dataset, this script lists the percent of the SNPs in that individual that match parent 1, the percent that match parent 2, the percent that are heterozygous, and the percent that are missing data. Note, that missing data calls are excluded from the calculation of the other three percents, so these four columns will NOT sum to one.

Parameters:

-i infile – file path/name for which you want to calculate the genome-wide percentages. Optional, defaults to stdin

-o outfile -- the absolute file name you want the results of the script written to. Optional, defaults to std.out

-p1 parent 1 -- this is the exact name of the data column containing your parent 1 calls. Required.

-p2 parent 2 -- this is the exact name of the data column containing your parent 2 calls. Required.

example call:

python3 Plumage_8_get_genome_percentages.py –i /
Users/Bob/my_GBS_project/Plumage_4_output.txt
–o /Users/Bob/Plumage_4_output_genome_percentages.txt –p1 IR64 –p2 Azucena

EXTRAS (for ricediversity.org users)

1. Plumage2_for_F2.py

Purpose: Use this script if you have an F2 population already in R/qtl format and you want to run the sequence error correction. Note that for this script only, your data may consist of both dominant and co-dominant markers, as longs as they are coded A, B, C, D, where A = homozygous AA, B= homozygous BB, C= Not AA (i.e. BB or H), and D= Not BB (i.e. AA or H).

Parameters:

-i infile: absolute path/name of the file, should be .csv in R/qtl format, only instead of leaving the column names for "chrom" and "position" blank, label them "chrom" and "pos". Default is stdin.

-o outfile: the absolute file name you want the results of the script written to. Defaults to std.out

-ct breakpoint cutoff: the number of matching calls that should follow a putative breakpoint in order for the breakpoint to be considered "good". Defaults to 5.

-e optional file path/name you want a report of the changes written to, defaults to no report written

example call:

python3 Plumage2_sequence_error_correction.py –i
/Users/Bob/my_GBS_project/my_F2_Rqtl.csv -o
/Users/Bob/my_GBS_project/F2_Plumage_2_output.csv

2. Plumage5_for_F2.py

Purpose: Use this script to transform and F2 population in nucleotide format (single codes as in TASSEL output) into R/Qtl format. Data can only contain codominant markers and any number of information columns.

Parameters:

-i	infile – full path/name for data file you want to transform. Optional, defaults to stdin
-o	outfile – full path/name for file you want results written to. Required.
-p1	parent 1 -- this is the exact name of the data column containing your parent 1 calls. Required.
-p2	parent 2 – this is the exact name of the data column containing your parent 2 calls. Required.

example call:

python3 Plumage_5_for_F2.py –i /Users/Bob/my_GBS_project/Plumage_4_output.txt –o /Users/Bob/Plumage_4_output_rqtl_format.csv –p1 IR64 –p2 Azucena