Package 'STITCH'

November 14, 2016

Type Package

Title STITCH - Sequencing To Imputation Through Constructing Haplotypes

Version 1.2.1

Date 2016-11-14

Author Robert William Davies

Maintainer Robert William Davies <robertwilliamdavies@gmail.com>

Description STITCH performs imputation of individuals sequenced to low coverage in a read aware fashion without a reference panel.

Installation To install, first install dependencies, then run the install.packages command, pointing to the downloaded tarball (STITCH.tar.gz)

Getting started A minimum run requires the following options to be set: the chromosome being run (chr); a path to a file with a set of bi-allelic SNP sites (posfile); a choice of K, the number of internally modelled haplotypes (K); a path to an output directory (outputdir); a path to a temporary directory, ideally on fast disks or a RAM disk (tempdir); a list of bam files (bamlist); and the number of generations since founding (nGen), which can be approximated from a choice of K for wild populations from 4 * Ne / K. Additional useful options relate to what region to impute (regionStart, regionEnd, buffer), whether to use validation data to benchmark imputation (genfile), the number of cores to use (nCores), whether imputation is run on a server or cluster (environment), the number of EM iterations (niterations), whether to run in diploid or pseudoHaploid mode (method), and if run in pseudoHaploid mode, what iteration to switch from pseudoHaploid to diploid (switchModelIteration).

Depends parallel, Rsamtools

Imports Rcpp

LinkingTo Rcpp, RcppArmadillo

RoxygenNote 5.0.1

License See attached LICENSE file

NeedsCompilation yes

R topics documented:

STITCH *Sequencing To Imputation Through Constructing Haplotypes*

Usage

```
STITCH(chr, nGen = "", posfile, K, outputdir, tempdir, bamlist = "",
  cramlist = "", reference = "", genfile = "", method = "diploid",
  outputInputInVCFFormat = FALSE, downsampleToCov = 50,
  downsampleFraction = 1, readAware = TRUE, chrStart = NA, chrEnd = NA,
  regionStart = NA, regionEnd = NA, buffer = NA,
  maxDifferenceBetweenReads = 1000, alphaMatThreshold = 1e-04,
  emissionThreshold = 1e-04, iSizeUpperLimit = as.integer(600),
  bqFilter = as.integer(17), niterations = 40,
  shuffleHaplotypeIterations = c(4, 8, 12, 16), splitReadIterations = 25,
  nCores = 1, expRate = 0.5, maxRate = 100, minRate = 0.1, Jmax = 100,
  regenerateInput = TRUE, originalRegionName = NA,
  keepInterimFiles = FALSE, keepTempDir = FALSE, environment = "server",
  pseudoHaploidModel = 9, diploidModel = 2,
  outputHaplotypeProbabilities = FALSE, switchModelIteration = NA,
  generateInputOnly = FALSE, restartIterations = NA,
  refillIterations = c(6, 10, 14, 18), downsampleSamples = 1,
  downsampleSamplesKeepList = NA, subsetSNPsfile = NA,
  useSoftClippedBases = FALSE, outputBlockSize = 1000,
  inputBundleBlockSize = NA, reference_haplotype_file = "",
  reference_legend_file = "", reference_sample_file = "",
  reference_populations = NA, reference_phred = 20,
  reference_iterations = 10)
```

Arguments

chr	What chromosome to run. Should match BAM headers
posfile	Where to find file with positions to run. File is tab seperated with no header, one row per SNP, with col 1 = chromosome, col 2 = physical position (sorted from smallest to largest), col 3 = reference base, col 4 = alternate base. Bases are capitalized. Example first row: 1<tab>1000<tab>A<tab>G<tab>
K	How many founder / mosaic haplotypes to use
outputdir	What output directory to use
tempdir	What directory to use as temporary directory. If possible, use ramdisk, like /dev/shm/
bamlist	Path to file with bam file locations. File is one row per entry, path to bam files. Bam index files should exist in same directory as for each bam, suffixed either .bam.bai or .bai
cramlist	Path to file with cram file locations. File is one row per entry, path to cram files. cram files are converted to bam files on the fly for parsing into STITCH

`reference` Path to reference fasta used for making cram files. Only required if cramlist is defined

`genfile` Path to gen file with high coverage results. Empty for no genfile. File has a header row with a name for each sample, matching what is found in the bam file. Each subject is then a tab seperated column, with 0 = hom ref, 1 = het, 2 = hom alt and NA indicating missing genotype, with rows corresponding to rows of the posfile. Note therefore this file has one more row than posfile which has no header

`method` How to run imputation - either diploid or pseudoHaploid, the former being the original method quadratic in K, the later being linear in K

`outputInputInVCFFormat`
Whether to output the input in vcf format

`downsampleToCov`
What coverage to downsample individual sites to. This ensures no floating point errors at sites with really high coverage

`downsampleFraction`
Downsample BAMs by choosing a fraction of reads to retain. Must be value 0<downsampleFraction<1

`readAware` Whether to run the algorithm is read aware mode. If false, then reads are split into new reads, one per SNP per read

`chrStart` When loading from BAM, some start position, before SNPs occur. Default NA will infer this from either regionStart, regionEnd and buffer, or posfile

`chrEnd` When loading from BAM, some end position, after SNPs occur. Default NA will infer this from either regionStart, regionEnd and buffer, or posfile

`regionStart` When running imputation, where to start from. The 1-based position x is kept if regionStart <= x <= regionEnd

`regionEnd` When running imputation, where to stop

`buffer` Buffer of region to perform imputation over. So imputation is run form regionStart-buffer to regionEnd+buffer, and reported for regionStart to regionEnd, including the bases of regionStart and regionEnd

`maxDifferenceBetweenReads`
How much of a difference to allow the reads to make in the forward backward probability calculation. For example, if P(read | state 1)=1 and P(read | state 2)=1e-6, re-scale so that their ratio is this value. This helps prevent any individual read as having too much of an influence on state changes, helping prevent against influence by false positive SNPs

`alphaMatThreshold`
Minimum (maximum is 1 minus this) state switching into probabilities

`emissionThreshold`
Emission probability bounds. emissionThreshold < P(alt read | state k) < (1-emissionThreshold)

`iSizeUpperLimit`
Do not use reads with an insert size of more than this value

`bqFilter` Minimum BQ for a SNP in a read. Also, the algorithm uses bq<=mq, so if mapping quality is less than this, the read isnt used

`niterations` Number of EM iterations.

`shuffleHaplotypeIterations`
Iterations on which to perform heuristic attempt to shuffle founder haplotypes for better fit. To disable set to NA.

`splitReadIterations`
: Iterations to try and split reads which may span recombination breakpoints for a better fit

`nCores`
: How many cores to use

`expRate`
: Expected recombination rate in cM/Mb

`maxRate`
: Maximum recomb rate cM/Mb

`minRate`
: Minimum recomb rate cM/Mb

`Jmax`
: Maximum number of SNPs on a read

`regenerateInput`
: Whether to regenerate input files

`originalRegionName`
: If regenerateInput is FALSE (i.e. using existing data), this is the name of the original region name (chr.regionStart.regionEnd). This is necessary to load past variables

`keepInterimFiles`
: Whether to keep interim parameter estimates

`keepTempDir`
: Whether to keep files in temporary directory

`environment`
: Whether to use server or cluster multicore options

`pseudoHaploidModel`
: How to model read probabilities in pseudo diploid model (shouldn't be changed)

`diploidModel`
: How to model read probabilities in diploid model (shouldn't be changed)

`outputHaplotypeProbabilities`
: Whether to output haplotype probabilities in files

`switchModelIteration`
: Whether to switch from pseudoHaploid to diploid and at what iteration (NA for no switching)

`generateInputOnly`
: Whether to just generate input data then quit

`restartIterations`
: In pseudoHaploid method, which iterations to look for collapsed haplotype prnobabilities to resolve

`refillIterations`
: When to try and refill some of the less frequently used haplotypes

`downsampleSamples`
: What fraction of samples to retain. Useful for checking effect of N on imputation. Not meant for general use

`downsampleSamplesKeepList`
: When downsampling samples, specify a numeric list of samples to keep

`subsetSNPsfile`
: If input data has already been made for a region, then subset down to a new set of SNPs, as given by this file. Not meant for general use

`useSoftClippedBases`
: Whether to use (TRUE) or not use (FALSE) bases in soft clipped portions of reads

`outputBlockSize`
: How many samples to write out to disk at the same time when making temporary VCFs that are later pasted together at the end to make the final VCF. Smaller means lower RAM footprint, larger means faster write.

`inputBundleBlockSize`
: If NA, disable bundling of input files. If not NA, bundle together input files in sets of <= inputBundleBlockSize together

`reference_haplotype_file`
: Path to reference haplotype file in IMPUTE format (file with no header and no rownames, one row per SNP, one column per reference haplotype, space separated, values must be 0 or 1)

`reference_legend_file`
: Path to reference haplotype legend file in IMPUTE format (file with one row per SNP, and a header including position for the physical position in 1 based coordinates, a0 for the reference allele, and a1 for the alternate allele)

`reference_sample_file`
: Path to reference sample file (file with header, one must be POP, corresponding to populations that can be specified using reference_populations)

`reference_populations`
: Vector with character populations to include from reference_sample_file e.g. CHB, CHS

`reference_phred`
: Phred scaled likelihood or an error of reference haplotype. Higher means more confidence in reference haplotype genotypes, lower means less confidence

`reference_iterations`
: When using reference haplotypes, how many iterations to use to train the starting data

`Number` of generations since founding or mixing. Note that the algorithm is relatively robust to this. Use nGen = 4 * Ne / K if unsure

Value

Results in properly formatted version

Author(s)

Robert Davies

Index