RFMix v1.0.2 Manual

Bustamante Lab, Stanford University

August 28, 2013

1. Installation

1.1 Instructions

- Download RFMix_v1.0.2.zip and unzip it
- In the PopPhased folder, compile using this command:
  - `g++ -Wall -O3 -ftree-vectorize -fopenmp main.cpp getdata.cpp randomforest.cpp crfviterbi.cpp windowtosnp.cpp -o RFMix_PopPhased`
- In the TrioPhased folder, compile using this command:
  - `g++ -Wall -O3 -ftree-vectorize -fopenmp main.cpp getdata.cpp randomforest.cpp crfviterbi.cpp windowtosnp.cpp -o RFMix_TrioPhased`
- Note: binaries compiled on Mac OS X 10.8.2 are provided

1.2 Testing your installation

- In the RFMix_v1.0.2 folder, run the command:
  - `python RunRFMix.py ./TestData/alleles1.txt ./TestData/classes.txt ./TestData/markerLocationsChr1.txt -o outputTrioPhased`
  - Note: requires Python v2.7 (will not work with v3.x)
- You should see the following output (may take a couple minutes):

```
Checking input...
Done checking input
./TrioPhased/RFMix_TrioPhased -a ./TestData/alleles1.txt -p
./TestData/classes.txt -m ./TestData/markerLocationsChr1.txt -o
outputTrioPhased
Giddyup
Reading files
Number of processors available: <computer/node dependent>
Number of SNPs Read: 51231
Number of SNPs to Exclude: 0
Number left after pruning: 51231
Number of windows: 1183
Growing forests
Applying Conditional Random Fields
Converting Window Calls to SNP Calls and Writing to File
Cleaning up
```

- In the TestData folder, run the following command (requires R to be installed)
  - `Rscript getDiploidAccuraciesTrioPhased.R`
- The Mean Accuracy should be approximately 97.5%
- In the RFMix_v1.0.2 folder, run the command:
  - `python RunRFMix.py ./TestData/alleles1.txt ./TestData/classes.txt ./TestData/markerLocationsChr1.txt --correct-phase -o outputPopPhased`
- You should see the following output (may take a couple minutes):

```
Checking input...
Done checking input
./PopPhased/RFMix_PopPhased -a ./TestData/alleles1.txt -p
./TestData/classes.txt -m ./TestData/markerLocationsChr1.txt -o
outputPopPhased
Giddyup
Reading files
Number of processors available: <computer/node dependent>
Number of SNPs Read: 51231
Number of SNPs to Exclude: 0
Number left after pruning: 51231
Number of windows: 1183
Creating phasings
Done creating phasings
Growing forests
Applying Conditional Random Fields
CRF on individual: 0
CRF on individual: 1
CRF on individual: 2
CRF on individual: 3
CRF on individual: 4
CRF on individual: 5
CRF on individual: 6
CRF on individual: 7
CRF on individual: 8
CRF on individual: 9
Converting Window Calls to SNP Calls and Writing to File
Cleaning up
```

- In the TestData folder, run the following command (requires R to be installed)
  - `Rscript getDiploidAccuraciesPopPhased.R`
- The Mean Accuracy should be approximately 97.5%

1.3 Troubleshooting

- If you are running on Mac OS X and you are unable to compile RFMix_PopPhased and RFMix_TrioPhased, it may be because the version of GCC on your computer is an Apple version that does not work with OpenMP. To get a newer version, make sure you have MacPorts installed and enter the following in Terminal:
  - `sudo port install gcc47`
  - `sudo port select --set gcc mp-gcc47`

2. Running

2.1 General Idea

RFMix uses designated reference haplotypes to infer local ancestry in designated admixed haplotypes. Fundamentally, RFMix takes in a matrix, with one row per bi-allelic SNP and one column per haplotype (both admixed and reference). The (i,j)th element of this matrix is the allele present at the ith SNP of the jth haplotype. RFMix returns the most likely (i.e. Viterbi) sequence of local ancestry along each haplotype, along with possibly other files depending on options selected.

2.2 Input Data Requirements

RFMix requires that input haplotypes are phased and that all SNPs are bi-allelic with no missing values. Programs such as BEAGLE (Browning & Browning 2009) are able to both phase and infer missing values. If doing phase correction, organism must be diploid with both copies of each chromosome adjacent to each other in the input files. At least two reference populations should also be provided in addition to the admixed sample.

2.3 Input Files

RFMix requires 3 input files that represent the matrix described in Section 2.1:

- **alleles**: one row per SNP and one column per haplotype. Values are 0 or 1, where each SNP has had its alleles converted to binary format. It doesn't matter which allele is coded as 0 or 1 in each SNP (e.g. dominant/recessive or major/minor doesn't matter). There is NO SPACE between alleles. For an example, see TestData/alleles1.txt
- **classes**: one row with one column per haplotype. The values are the ancestry of each haplotype (0 – admixed, 1 – 1st ancestry, 2 – 2nd ancestry, etc.). Do not skip numbers when assigning indices to ancestries. There are SPACES between columns. For an example, see TestData/classes.txt
- **snp_locations**: one row per SNP and one column. Values are the genetic coordinates in centimorgans for each SNP. Values must be in increasing order (i.e. SNPs must be arranged in order). For an example, see TestData/markerLocationsChr1.txt

2.4 Command

RFMix is actually two separate programs: RFMix_TrioPhased does not attempt to correct phase errors in the admixed haplotypes, while RFMix_PopPhased does. For simplicity, the script RunRFMix.py is provided and calls the appropriate RFMix program depending on a user-specified option (see below). The most basic RFMix command takes the 3 input files described above:

```
python RunRFMix.py alleles classes snp_locations
```

For a list of additional command options and an example, see below. Also, it is possible to get a list of command options and descriptions by running

```
python RunRFMix.py -h
```

2.5 Output Files

- **Viterbi**: Have *.Viterbi.txt suffix. One row per SNP and one column per admixed haplotype in same order as input, with spaces between columns. If `--correct-phase` option was used, then this will be with respect to the CORRECTED haplotype phasing. If EM was used, multiple Viterbi files will be produced and indexed by EM iteration. If `--use-reference-panels-in-EM` option was used, the inferences on the reference haplotypes will be included as well (again in the same order as input file).
- **Forward-Backward**: Have *.ForwardBackward.txt suffix. Not output unless `--forward-backward` option included. Not currently implemented for when `--correct-phase` option is included (however, one can use the corrected phasings from these runs as input to a non-correct-phase run to get forward-backward estimates). Same format as Viterbi, except instead of one column per haplotype, each haplotype gets one column per ancestry. The value is the posterior probability of that ancestry at that SNP in that haplotype. For example, if there are 2 reference populations, the first row will be: <probability of ancestry 1 at SNP 1 in admixed haplotype 1>, <probability of ancestry 2 at SNP 1 in admixed haplotype 1>, <probability of ancestry 1 at SNP 1 in admixed haplotype 2>, etc.
- **Corrected Phasings**: Have *allelesRephased* in name. Only output if `--correct-phase` option is included. Same format as alleles file except only contains admixed haplotypes. If running EM then multiple files will be created, indexed by EM iteration.
- **Excluded SNPs**: Have *toExclude.txt suffix. Created by RunRFMix.py if `--discard-rare-variants` option is used and used as input to the actual RFMix programs. One column with one row per SNP excluded. Values are the (0-based) index of that SNP.

2.6 Options

- `--correct-phase : has RFMix simultaneously attempt to correct phasing errors. Significantly slows analysis and currently does not return Forward-Backward estimates`
- `-w, --window-size : window size in genetic distance. Default=0.2cM`
- `-G, --generations : generations since admixture event. Default=8`
- `-o, --output-name : prefix for output file. Default=ancestry_output`
- `-t, --trees : number of trees to generate per random forest. Default=100. Decreasing will speed up RFMix but may eventually result in more variable estimates due to the stochastic nature of random forests.`

- `--disable-parallel : turn off OpenMP (shared memory) parallelization`
- `--num-threads : number of threads to use if doing parallelization. Default is number of processors on computer/node. Increasing this number beyond the number of available nodes/processors will not result in additional speedup.`
- `-b, --bootstrap-sample-size : number of bootstrap samples used per tree. Default=<number of reference haplotypes>`
- `-s, --bootstrap-type : kind of bootstrap sampling. 0-sample with equal prob from each haplotype, 1-(Default)-sample with equal probability from each class, 2-stratified by class`
- `-e, --em-iterations : number of EM iterations. Default=0`
- `--use-reference-panels-in-EM : not using this flag means that reference panels are discarded after the initial inference step`
- `-x, --discard-rare-variants : ignore SNPs with minor allele occurrence less than the given value e.g. an argument of 2 will ignore SNPs with minor alleles that occur 1 or less times across both the admixed and reference panels. Outputs file with suffic toExclude containing 0-based indices of removed SNPs. Default=0`
- `-f, --mtry-factor : mtry is the number of SNPs to consider when creating a tree node in a random forest. The standard is to use the square root of the number of SNPs in the window. This multipiplies that number`
- `--forward-backward : output the forward-backward probabilities. Defaults to not doing so because of potentially large file sizes`
- `--skip-check-input-format : don't do initial check of input files within RunRFMix.py script`

2.7 Example Run

Say we've sample 10 individuals from an admixed population with ancestries from Europe, Africa and Asia. We've got HapMap reference populations (CEU, YRI, JPT+CHB) that we want to use as reference.

- We begin by taking the usual precautions when combining data sets (e.g. looking out for strand flips) and won't go into detail on those here.
- We phase and impute missing values in the admixed samples using Beagle, giving us the files in TestData/Pre-Processed
- We then create the alleles and classes files by running ConvertToRFMixFormat.R
- The snp_locations file is created separately.
- We think there may be a decent number of phase errors in the admixed samples, so we run RFMix with the phase correcting option (note: the example file actually has perfect phasing). Since the number of admixed samples is small compared to the reference sample sizes we don't bother with EM. We keep the remaining options as defaults because these defaults work well in most situations.

- `python RunRFMix.py ./TestData/alleles1.txt ./TestData/classes.txt ./TestData/markerLocationsChr1.txt --correct-phase -o outputPopPhased`

- We see the following output. RunRFMix.py checks the input formatting and passes the command output in the third line to the appropriate RFMix binary. The remaining output is from this binary.

```
Checking input...
Done checking input
./PopPhased/RFMix_PopPhased -a ./TestData/alleles1.txt -p
./TestData/classes.txt -m ./TestData/markerLocationsChr1.txt -o
outputPopPhased
Giddyup
Reading files
Number of processors available: <computer/node dependent>
Number of SNPs Read: 51231
Number of SNPs to Exclude: 0
Number left after pruning: 51231
Number of windows: 1183
Creating phasings
Done creating phasings
Growing forests
Applying Conditional Random Fields
CRF on individual: 0
CRF on individual: 1
CRF on individual: 2
CRF on individual: 3
CRF on individual: 4
CRF on individual: 5
CRF on individual: 6
CRF on individual: 7
CRF on individual: 8
CRF on individual: 9
Converting Window Calls to SNP Calls and Writing to File
Cleaning up
```

- Note: the conditional random field inference step takes longer when doing phase correction, and its progress is monitored with the "CRF on individual" output
- We can then look in the outputPopPhased.0.Viterbi.txt file for the called local ancestries.
- To see how well we did, we can run getDiploidAccuraciesPopPhased.R to compare with the solutions to this example stored in admSolutionChr1.bgl.phased (unfortunately not available in real life).
- We see we get an accuracy of ~97.5%. This changes a small amount every time we run RFMix because of the stochastic nature of Random Forests.

2.8 Troubleshooting

- If a very large run is performed (e.g. 20,000 haplotypes) and the process runs out of memory, there are several possible solutions. One is to increase the stack size, as it is likely that the recursive tree construction is causing a stack overflow. Another is to reduce the window size (for the same reason). A third is to simply divide the haplotypes into separate runs.
- If running on shared cluster and you want to use parallelization, determine the number of processors you want, request that number in your submission and also make sure to include the `--num-threads` RFMix option with that

number as the argument. Otherwise RFMix will try to use all available nodes, and this upsets cluster admins.

3. Citing

To cite RFMix, please reference the following publication:

Maples BK, Gravel S, Kenny EE, and Bustamante CD. (2013). RFMix: A Discriminative Modeling Approach for Rapid and Robust Local-Ancestry Inference. Am. J. Hum. Genet. *93*, 278-288