1 Installing the software

1.1 Java compiler and necessary class libraries

The DualBrothers package is distributed as a Java JAR file (DualBrothers.jar). In order to use the package, a Java virtual machine (JVM) and the Colt class library must be installed. The JVM must support the Java 1.2 language specification. We currently use either or Sun's Java 1.4.2 with HotSpot technology . The Colt class library is an open source library for high performance scientific and technical computing in Java. Version 1.0.3 is currently available for download at `http://tilde-hoschek.home.cern.ch/\~{}hoschek/colt/`.

1.2 Setting the classpath

Once the JVM has been installed, the environment variable CLASSPATH must be set such that it includes both DualBrothers.jar and colt.jar. An example command appropriate under the Linux bash shell is shown below. The command may either be issued from the command prompt or in the login script `.bashrc`.

```
[davinci]$ export CLASSPATH=$CLASSPATH:path_to_file/DualBrothers.jar:path_to_file/colt.jar
```

2 Preparing the data and command files

2.1 Sequence data

The DualBrothers package reads in Phylip 4.0 formatted sequence files. There is no maximum line length, so the entire sequence for one taxon may be entered on a single line. Sequences may also be wrapped by interleaving them. **The DualBrothers package assumes that the putative recombinant sequence is the LAST sequence in the Phylip data file.** Erroneous results will obtain if the putative recombinant is positioned elsewhere. Below is a truncated example of a Phylip 4.0 formatted sequence file with putative recombinant sequence X positioned last:

```
 9 1480
A             NNNNNNNCGA GAGCGTCAGT ATTAAGNGGG GGAAAATTAG ATGCATGGGA
B             ATGGGTGCGA GAGCGTCAGT ATTAAGCGGG GGAGAATTAG ATAGATGGGA
C             ATGGGTGCGA GAGCGTCAAT ATTAAGAGGG GNAAAATTAG ATAAATGGGA
D             ATGGGTGCGA GAGCGTCAGT ATTAAGCGGG GGAAAATTGG ATGCATGGGA
F             ATGGGTGCGA GAGCGTCAGT ATTAAGCGGG GGAAAATTAG ATGCATGGGA
G             ATGGGTGCGA GAGCGTCAGT ATTAAGCGGG GGAAAATTAG ATGCTTGGGA
```

```
H             ATGGGTGCGA GAGCGTCAGT ATTAAGCGGG GGAAAATTAG ATGCTTGGGA
J             ATGGGTGCGA GAGCGTCAAT ATTAAGTGGG GGAAAATTAG ANGATTGGGA
X             ATGGGTGCGA GAGCGTCAGT ATTAAGCGGG GGAAAATTAG ATGCATGGGA

              GAGTCAAGTA CAACAGACA- --NNNAACAT
              GAGCCAAGTA ACAAATTCAN NNGCTACCAT
              GAGCCAAGCA AACAGTNCC- -----NNNAT
              GAGCCAAGCA ACAAGTTCAG CTGCTGCAGT
              GAGCCAAGCA ACAAATACAN NNNNNNCCAT
              GAGCCAGGCA TCAGGTGCAG CAGCAGCCAT
              GAGCCAAGTA ACAAATGCAA ATGCAGCCAT
              GAGCCAAGTG ACCAATACN- -----AACAT
              GAGCCAAGTA ---AATACAA ATGCAGTTAT
```

2.2 Command file

The posterior sampler class within the DualBrothers package requires a text-based **command file** to run. This file specifies all sampler parameters. The most important parameters to adjust are the tree of putative parental taxa, the hyperprior parameters on the change-point processeses and the posterior sample length. There are also many adjustable parameters to tune the sampler transition kernels. Efficient sampling depends on reasonable choices for these parameters.

In the **command file**, parameters are specified one per line in the form:

```
<parameter name>: <value>
```

Below is a list of parameters that may appear in the **command file** along with a default value and a brief description of each parameter's function. Please read the descriptions for important information about running DualBrothers.

Parameter	Default Value	Description
length:	210000	Total number of samples to generate
burnin:	10000	Number of initial samples to discard
subsample:	20	Frequency at which samples are saved to the **output file** after the burn-in
		The above configuration generates 210,000 posterior samples, discards the first 10,000 as burn-in and then records every 20th in the **output file**. The final posterior sample size is 10,000.

start_tree:	no default	Specifies the fixed topology relating the N potential parentals. The current version is not user-friendly. In the current version, taxon labels in the tree must be 0, 1, ..., $N-1$ representing the 1st, 2nd, ... , Nth parental sequence listed in the textbfdata file, respectively. The putative recombinant must also be listed in the tree with label N, although it's location does not matter, as the sampler will permute the putative recombinant's location all over the tree. The remaining parentals are fixed in place by this tree. Finally, the tree must be rooted with taxon 0 as the "outgroup." The sampler uses a reversible model, so specifying the true rooting among the parentals is not necessary.
tree_file:	no default	Name of the file with a list of trees in Newick format. Each tree must start on a new line. This option allows user to include arbitrary topologies into the tree space. At least 3 trees are required in the list!
		CAUTION!!! You must supply either "start_tree" or "tree_file" but NOT BOTH. Very little checking is done for trees in the list. Make sure you are not including the same topology more than one time.
par_lambda:	5	Hyperprior parameter representing the prior mean number of substitution process change-points (See Equation (3) Minin et al. [2005]). The paper also describes how to choose par_lambda.
top_lambda:	0.693	Hyperprior parameter representing the prior mean number of topology break-points (See Equation (3) in Bioinformatics paper). We recommend setting top_lambda to $\log 2 = 0.693$ when no prior information about recombination is available.

		The above parameters must be set for the MCMC sampler to run. DO NOT USE THE DEFAULT VALUES. Adjust these parameters to match the specific data and problem addressed.
window_length:	10	Size of window around which existing change-points are randomly moved during update
C:	0.30	Peter Green's constant. Adjust this constant to change the proportion of time that the sampler spends updating parameters when the number of dimensions is fixed. (see [Suchard et al., 2003])
		The above parameters modify the transition kernels. Their values can be adjusted later to optimize the sampler's acceptance rates and to speed up convergence of the chain. They are described in detail in the technical report.
sigma_alpha:	0.75	Spread of αs when proposing new segments (see [Suchard et al., 2003])
sigma_mu:	0.75	Spread of μs when proposing new segments (see [Suchard et al., 2003])
subst_hyper_mean:	no default	prior mean of $\log \kappa$
subst_hyper_variance:	no default	prior variance of $\log \kappa$
diver_hyper_mean:	no default	prior mean of $\log \mu$
diver_hyper_variance:	no default	prior variance of $\log \mu$
		If you want to fix the four above parameters, you must provide values for all of them. If some of the values are missing the program will terminate. If ALL values for hyperparameters are missing they will be estimated simultaneously with other model parameters. (See Minin et al. [2005])
top_breaks:	1	1 = include topology break-points into the model, 0 = do not include
par_breaks:	1	1 = include substitution change-points into the model, 0 = do not include

3 Running the sampler

To generate a posterior sample from the change-point model, use the following command:

```
java DualBrothers <seed> <command file> <data file> <output file>,
```

where <seed> is any integer and is used to seed the package's pseudo-random number generator. Here is an example,

```
java DualBrothers 123  KAL153.cmdfile KAL153.phy KAL153.post
```

uses the random number seed 123, reads `KAL153.cmdile` as the **command file**, uses `KAL153.phy` as the **data file** and saves the posterior sampler output to the **output file** `KAL153.post`.

The **output file** is difficult to interpret directly since the change-point model has a variable number of dimensions. To ease interpretation of the **output files**, use the two following commands:

```
java TopologyProfile <output file> <topology profile file> <sequence length>
java EPProfile <output file> <kappa/mu profile file> <sequence length>
```

These programs read the **output file** and compute the marginal posterior values for the topology, κ and μ at each of the ¡sequence length¿ sites in the data. These values are saved in the ¡topology profile file¿ and ¡kappa/mu profile file¿. The format of these files are described in the next section.

4 Interpreting the posterior sample:

The DualBrothers class saves the posterior sample in a text file. Each row in this textfile corresponds to a different posterior sample, where the number in the first column of each row reports the chain-step number for that sample. Because the multiple change-point model has a variable number of parameters, the number of columns in each row also varies. As a results, this **output file** is difficult for standard statistical packages to read and interpret. To overcome this difficult, the DualBrothers packages contains two classes that interpret the posterior sample and generate easy to read summaries of the posterior.

These two classes can be invoked by:

```
java TopologyProfile <output file> <topology profile file> <sequence length>
java EPProfile <output file> <kappa/mu profile file> <sequence length>
```

and generate two text files ¡topology profile file¿ and ¡kappa/mu profile file¿. Both of these files can easily read into a statistical or plotting program, such as R.

The file ¡topology profile file¿ is formatted as follows:

```
<site> <posterior probability of tree 1> <posterior probability of tree 2> ....
```

where <site> corresponds to the sequence alignment site in the order the alignment is given in the **data file** and trees 1, 2 etc refer to the trees outputted to the STDERR when "java TopologyProfile" is run. The trees are sorted in order of frequency observed marginalized over the entire alignment length. For example, the command

```
java TopologyProfile KAL153.post KAL153.tree 8588
```

generates STDERR output

```
Read in 10001 lines.
Found 3 unique trees.
Tree list: (0,(1,(2,3))) (0,((1,3),2)) (0,((1,2),3))
```

that gives the order of the trees. The posterior probabilities are marginal probabilities for each listed site.

The file <kappa/mu profile file> gives the marginal posterior estimates of the evolutionary parameters for each list site. The format is as follows:

```
<site> <kappa-mean> <mu-mean> <kappa-q2.5%> <kappa-q50%> <kappa-q97.5%> <mu-q2.5%> <mu-q50%> <mu-q97.%> <COP>
```

where <site> is the sequence alignment site and remaining columns give the marginal posterior means and quantiles of the scaled transition:transversion rate ratio kappa and expected divergence mu and the marginal posterior probability that a COP exists at that site. For example, the command

```
java EPProfile KAL153.post KAL153.profile 8588
```

records posterior profiles of evolutionary profiles into the file KAL153.profile

References

V.N. Minin, K.S. Dorman, F. Fang, and M.A. Suchard. Dual multiple change-point model leads to more accurate recombination detection. *submitted*, 2005.

M.A. Suchard, R.E. Weiss, K.S. Dorman, and J.S. Sinsheimer. Inferring spatial phylogenetic variation along nucleotide sequences: a multiple change-point model. *Journal of the American Statistical Association*, 98:427–437, 2003.