Moves on real values (e.g. branch lengths, proportion of invariant sites)

"mvScale"

A simple "multiplication" move: the scaling proposal draws a random uniform number u ~ unif(-0.5,0.5) and scales the current value by a scaling factor

sf = exp(lambda * u)

where lambda (rate of decay) is the tuning parameter of the proposal to influence the size of the proposals. Values that end up outside the interval of the variable are ignored, i.e., the move is simply aborted.

"mvSlide"

An additive move using a sliding window: the slider proposal draws a random uniform number u ~ unif(-0.5,0.5), multiplies it by the tuning parameter delta, and adds it to the current parameter value.

newVal = val + (delta * u)

"mvScalerUpDown"

This scaler proposal moves two values at the same time. It obtains a scaling factor in the same way as the simple scaler move, and then multiplies one of the values with the factor while it divides the other.

Moves on integer values (e.g. non-continuous morphological traits, number of local clocks)

"mvRandomIntegerWalk"

This is a very simple move on integer numbers that proposes with probability p = 0.5 to increase the current value by 1 and with probability p = 0.5 to decrease the current value by 1. Thus, it is a random walk, and there is no tuning parameter (i.e., no auto-tuning possible).
This is again done without reflection, i.e., the move is simply aborted if the proposed value is outside the interval for the parameter.

"mvRandomGeometricWalk"

This move on integer numbers proposes with probability p = 0.5 to increase the current value and with probability p = 0.5 to decrease the current value. The number of steps taken by this move (value by which the current value is increased or decrease) is geometrically distributed. The probability parameter of the geometric distribution ("x") is used as tuning parameter. As this version allows taking several steps at a time, this move is useful if an integer has a wide range. Average step size can be auto-tuned.

Moves on simplices (e.g. base frequencies, exchangeability rates for GTR)

"mvSimplex"

This proposal updates simplex values by proposing new values from a Dirichlet distribution centred on the current values. The i-th parameter of the Dirichlet is the i-th value in the simplex multiplied by a parameter (alpha, the tuning parameter) that controls the variance of the Dirichlet. So, if a very high value of the tuning parameter is chosen, then the newly proposed values of the simplex will be very close to the previous values. A low tuning parameter leads to more dissimilar values and thus larger moves.
All values of the simplex are here chosen at the same time.

"mvSimplexElementScale"

This simplex move chooses one of the elements at random. It then draws a new value from a beta distribution centred on the current value, again with the variance of the beta distribution determined by the tuning parameter alpha. You can make bolder proposals by increasing alpha. All elements of the simplex are then scaled up or down so that they again sum to 1.

Moves on vectors of real values

"mvVectorScale"

This move obtains the scaling factor in the same way as the other scaling moves (involving the tuning parameter lambda), and scales each element of the vector by this factor.

"mvVectorSingleElementScale"

This move takes a single element of a vector of values at random, and changes it with a classic scale move. The other elements remain unchanged.

"mvVectorSingleElementSliding"

This move takes a single element of a vector of values at random, and changes it with a classic sliding window move. The other elements remain unchanged.

"mvRlcRateScale"

This move is used in conjunction with Random Local Clocks. It scales the substitution rate if a jump is occurring on the branch in question.

"mvSwitchRateJump"

This proposal adds or removes a rate jump, e.g. in the context of a random local molecular clock. If a jump is added, then the new substitution rate is simply drawn from the prior.

Tree proposals: Topology (& branch lengths)

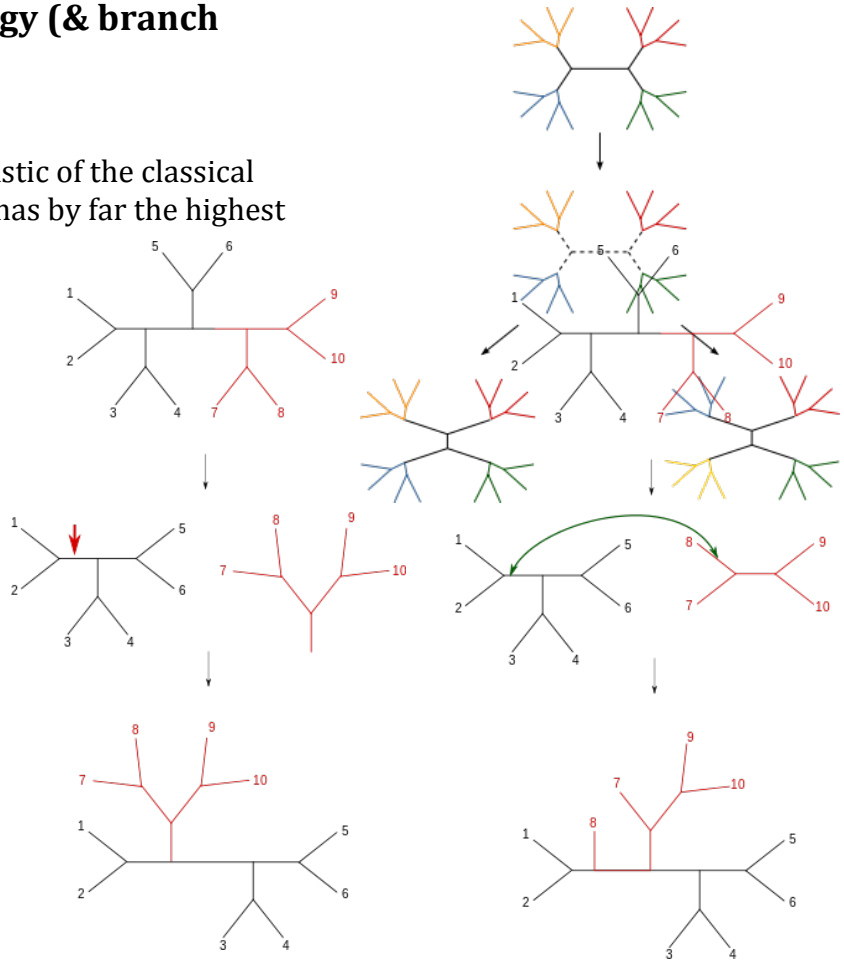
"mvNNI"

The NNI move is the least drastic of the classical topology moves, and usually has by far the highest acceptance rate (even though it is far below the acceptance rates for scalar parameters).

"mvSPR" = Subtree pruning regrafting

This move changes the topology by cutting a subtree and re-attaching it to another branch. It is only implemented for nonclock trees.

"mvTBR" =
Tree bisection reconnection
The TBR move, which re-attaches not with the previous attachment point of the subtree, but with any branch, is not yet implemented.



Examples of SPR (left) and TBR (right) moves on unrooted topologyes

Tree proposals: branch lengths only

"mvNodeTimeSlideUniform"

In the clock case, slides a single node uniformly between its parent and older descendent.

"mvOriginTimeSlide"

Slides a whole subtree (?).

"mvRateAgeBetaShift"

Changes the age of a random node according to a beta distribution, and then also adapts the branch rates around this node.

"mvRootTimeSlide"

Additive move on the root age, taking care that it does not become younger than any of its children.

"mvSubtreeScale"

Rescales a whole subtree, chosen randomly.

"mvTreeScale"

Scales the whole tree by a factor determined by the tuning parameter and a random number.