User Manual for SplitsTree4 V4.14.6

Daniel H. Huson and David Bryant

September 26, 2017

Contents

1 Introduction

License: Copyright (c) 2015, Daniel H. Huson and David J. Bryant

This program is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program. If not, see `http://www.gnu.org/licenses`.

Type-setting conventions: In this manual we use e.g. `Network`→NeighborNet to indicate the `NeighborNet` menu item in the `Network` menu. We use e.g. `Main:Source` to indicate the `Source` tab in the `Main` window.

How to cite: If you publish results obtained in part by using SplitsTree4 , then we require that you acknowledge this by citing the program as follows:

- D.H. Huson and D. Bryant, *Application of Phylogenetic Networks in Evolutionary Studies*, Molecular Biology and Evolution, 23(2):254-267, 2006. software available from `www.splitstree.org`,

Evolutionary relationships are usually represented using phylogenetic trees, based on a model of evolution dominated by mutations and speciation events. More realistic models must also account for gene genesis, loss and duplication events, hybridization, horizontal gene transfer or recombination. Here, phylogenetic networks have a role to play.

Moreover, network methods also provide a value tool for phylogenetic inference even when reticulation events do not play an important role. The combined effect of *sampling error* and *systematic error* makes phylogenetics an uncertain science, and network methods provide tools for representing and quantifying this uncertainty.

The aim of SplitsTree4 is to provide a framework for evolutionary analysis using both trees and networks. The program takes as input a set of taxa represented by characters (that is, aligned sequences), distances, quartets, trees or splits and produces as output trees or networks using a number of different methods.

This document provides both an introduction and a reference manual for SplitsTree4 .

2 Getting Started

This section describes how to get started.

First, download an installer for the program from from www.splitstree.org, see Section 3 for details.

Use the File→Open menu item to open a file containing data such as character sequences, distances or trees. The file must be in one of the following formats: Nexus, ClustalW, *PhylipParsimony*, FastA or *Newick*. Alternatively, select the File→Enter Data menu item and enter data in one of these formats by hand.

Example files are provided with the program. They are contained in the `examples` sub directory of the installation directory. The precise location of the installation directory depends upon your operating system.

Use the different menu items to determine which methods are applied to the input data. The Distances, Trees and Network menus contain items that determine how to compute distances, trees or a network from the given data. Some of the methods provided have parameters that can be set using the Analysis→Configure Pipeline or Analysis→Configure Recent Methods items.

The computed data can be viewed in text form in the Data tab and can be saved using the File→Save As item.

The computed network or tree is displayed in the Network tab. Attributes of the network can be changed by selecting nodes or branches (also called edges) and using the Format Nodes and Edges dialog which is reachable using the View→Format Nodes and Edges item.

Individual blocks of data can be saved in a number of different formats using the File→Export item. The network displayed in the Network tab can be saved in a graphics format using the File→Export Image item.

3 Obtaining and Installing the Program

SplitsTree4 is written in Java and requires a Java runtime environment version 1.7 or newer, freely available from www.java.org.

SplitsTree4 is installed using an installer program that is freely available from www.splitstree.org. There are four different installers, targeting different operating systems:

- `splitstree_windows-x64_4.14.6.exe` provides an installer for Windows 64-bit (newer systems).
- `splitstree_windows_4.14.6.exe` provides an installer for Windows 32-bit (older systems).
- `splitstree_macos_4.14.6.dmg` provides an installer for Mac OS.
- `splitstree_unix_4.14.6.sh` provides a shell installer for Linux.

4 Program Overview

In this section, we give an overview over the main design goals and features of this program. Basic knowledge of the underlying design of the program should make it easier to use the program.

SplitsTree4 is written in the programming language Java. The advantages of this is that we can provide versions that run under the Linux, MacOS, Windows and Linux operating systems. Additionally, this makes it possible for the program to support plug-ins that can add new functionality to the program, such as new methods and import or export modules. A potential draw-back is that an algorithm implemented in Java will generally run slower than the same algorithm implemented in C or $C++$.

Earlier versions $1-3$ of the program [20] were written in $C++$ and only contain a small part of what is now available with SplitsTree4 .

SplitsTree4 uses multi-threading and supports multiple documents. This means that that you can work on more than one data set simultaneously, in different windows, and run multiple calculations simultaneously, making use of multiple processors, when available.

A *Document* consists of an individual data set and possesses its own `Main` window. The document is discarded, when its window is closed. The program is based on the Nexus format, as introduced in [27] and the data associated with a document is organized into *blocks*, and each such block of data is represented by a corresponding "block" in the Nexus format. The blocks are:

- `Taxa`: the names of all taxa.
- `Unaligned`: unaligned sequences.
- `Characters`: aligned character sequences.
- `Distances`: pairwise distances between taxa.
- `Quartets`: (possibly weighted) quartet topologies.
- `Trees`: list of (possibly partial) trees.
- `Splits`: (possibly weighted) splits.
- `Network`: phylogenetic tree or network.
- `ST-Assumptions`: contains all methods and options used to compute data.
- `ST-Bootstrap`: bootstrap support of splits.

The first eight blocks `Taxa`, `Unaligned`, `Characters`, `Distances`, `Quartets`, `Trees`, `Splits` and `Network` are organized as a *pipeline* and data is processed from left-to-right along this pipeline. Any non-empty document must contain a `Taxa` block and will usually contain an `ST-Assumptions` block. All other blocks are optional and the presence or absence of some block depends on the set of computations that the user has selected.

We will use the term *source block* to denote the left-most block in the pipeline (excluding the `Taxa` block). The source block contains the original data that is provided to the program. Any

computations performed by the program will update blocks from left to right along the pipeline, starting after the source block. Note that some types of computations do not fit into this pipeline design, for example SplitsTree4 cannot provide any method that takes a `Splits` block and produces a `Trees` block, because the latter occurs before the former in the pipeline.

Typically, the user will provide a source-block and will then use different menu items to determine how the program will compute data along the pipeline until a `Network` block has been computed and an image of the network has been drawn in the `Main` window.

The program is designed to keep all blocks in the pipeline *synchronized* by enforcing that the different blocks only contain data that has been computed via the pipeline. For example, if you attempt to load data from a file in Nexus format that contains a `Taxa` and two or more other blocks, e.g. a `Characters` block or `Distances` block, then the program will request you to choose which of the two latter blocks you want to keep. (This does not apply if the file was created by SplitsTree4 using the `File→Save` or `File→Save As` command, in which case the blocks in the file are synchronized, and so none must be discarded and no computations are necessary).

Once a source block has been provided, the program will proceed to perform a chain of *default calculations*, which differ, depending on the type of the source block. In the case of a `Characters` block, the program will use `UncorrectedP`, `NeighborNet` and then `EqualAngle` to compute a network for the data. In the case of a `Distances` block, `NeighborNet` and `EqualAngle` are used. In the case of a `Trees` block, the first tree in the block is displayed using the `EqualAngle` algorithm.

The data contained in the different blocks of the pipeline is displayed in the `Data` tab.

The program is designed to operate in two differents modes: in a GUI mode, the program provides a GUI for the user to interact with the program. In command-line mode, the program reads commands from a file or from standard input and writes output to files or to standard output.

5 Splits, Trees and Networks

Here we give a brief introduction to some of the concepts from phylogenetics that the program is based on.

Evolutionary relationships between taxa are usually represented using a phylogenetic *tree*. Such trees are often computed from molecular sequence data, either directly, using methods such as parsimony, or indirectly, by first computing a distance matrix and then applying a method such as Neighbor-Joining.

Such approaches implicitly or explicitly model the evolution of a single gene under the assumption that the process is dominated by two types of events, mutations and speciation events. Under more complex models of evolution, i.e. involving gene loss and duplication, or hybridization, horizontal gene transfer or recombination, a single phylogenetic tree will often not be an appropriate representation of the phylogentic history or of the different incompatible phylogenetic signals. Also, the presence of noise in a data set, or uncertainty due to inadequacies of the underlying model up on which a tree inference is based, may also make it necessary to use a more general graph, that is, a *network*, to represent the data.

The aim of SplitsTree4 is to provide a frame-work for computing phylogenetic networks. As the name of the program suggests, it is based on the fundamental mathematical concept of a *split* . For example, if we are given an alignment of binary sequences

```
a 010011010110
b 100001011110
c 011001101110
d 010001101111
```

then each non-constant column in the alignment defines a split of the taxon set consisting of those taxa with the value 0 and those with the value 1. For example, the first column partitions the taxa into two sets $\{a, c, d\}$ and $\{b\}$, and thus gives rise to the split $\frac{\{a,c,d\}}{\{b\}} (= \frac{\{b\}}{\{a,c,d\}})$, while the fourth column does not define a split because the characters are constant.

Mathematically, for a set of X taxa any phylogenetic tree T defines a set of such splits called the *split encoding* $\Sigma(T)$ of T, as follows: deletion of any *branch* (also called *edge*) e in the tree produces two subtrees, T_A and T_B, say, and they give rise to a split $S = \frac{A}{B} = \frac{B}{A}$, consisting of the set A of all taxa contained in T_A and the set B of all taxa contained in T_B. In the literature, a split $\frac{A}{B}$ is sometimes denoted by $A|B$ or $\{A, B\}$.

For example, consider the tree T displayed in Figure 1. Its split encoding $\Sigma(T)$ contains 5 *trivial splits* that separate a single taxon from all other taxa, and 2 *non-trivial splits* that contain at least two taxa in both parts. The trivial splits are:

$$\frac{\{a\}}{\{b,c,d,e\}}, \frac{\{b\}}{\{a,c,d,e\}}, \frac{\{c\}}{\{a,b,d,e\}}, \frac{\{d\}}{\{a,b,c,e\}} \text{ and } \frac{\{e\}}{\{a,b,c,d\}},$$

and the non-trivial ones are:

$$\frac{\{a,b\}}{\{c,d,e\}} \text{ and } \frac{\{a,b,e\}}{\{c,d\}}.$$

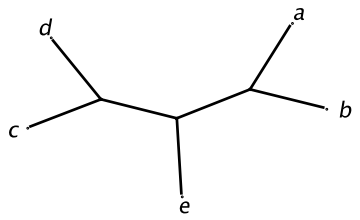

Figure 1: An unrooted phylogenetic tree.

A fundamental result in mathematical phylogeny states that any given set of splits Σ corresponds to some phylogenetic tree T, with $\Sigma(T) = \Sigma$, if and only if Σ is *compatible* [8]. Thus, any phylogenetic tree may be viewed as a graph whose task it is to give a visual representation of a given set of compatible splits.

A phylogenetic tree can be thought of as an idealized representation of the historical relationships among a set of taxa; and tree building methods are attempts to find a the set of compatible splits

that are most consistent with the data according to some algorithm. Often there are multiple tree that are equally consistent with the data, i.e. multiple sets of compatible splits. In general that collection of splits will be incompatible. Moreover, there exist inference methods such as split decomposition and Neighbor-Net that compute a set of incompatible splits from the data in the form of a given distance matrix. Note that split decomposition produces a set of splits that are called *weakly compatible*, while Neighbor-Net produces a set of splits that is *circular*, as originally defined in [2].

A *split network* is a more general type of phylogenetic graph that can represent any collection of splits, whether incompatible or not [2]. For a compatible set of splits, it is always possible to represent each split by a single branch, and thus the resulting graph is a tree. In general, however, this will not be possible and in a split network usually a whole band of *parallel branches* (also called *parallel edges*) is required to represent a single split. A phylogenetic tree is therefore a special case of a split network. In SplitsTree4, if you click on any branch in the representation of a split network, then all branches corresponding to the same split will be highlighted. If one were to delete all branches corresponding to a given split S, then the remaining graph will consist of precisely two components, G_A and G_B, and, as above, the we have $S = \frac{A}{B}$, where A is the set of all taxa contained in G_A and B is the set of all taxa contained in G_B.

A more detailed discussion of the fundamental concepts can be found in [22] and [21].

6 Opening, Reading and Writing Files

To open a file, select the `File→Open` menu item and then browse to the desired file. Alternatively, if the file was recently opened by the program, then it may be contained in the `File→Open Recent` submenu.

The native file format of SplitsTree4 is based on the Nexus format, see [27]. However, the program can also parse a number of other formats, including ClustalW format for aligned sequences, Phylip format for sequences and distances, FastA format for aligned distances and Newick format for trees. Earlier versions had separate Open and Import menu items; these have now been combined.

If a file is opened while the `Network` tab or `Data` tab is open, then the program will attempt to parse and execute the file. If the file is a complete Nexus file previously generated by SplitsTree4, then the network described in the file will be displayed. If the file is in some other format, then, depending on the type of content, the program will perform a chain of default calculations.

To save the complete data associated with a given window in SplitsTree4 's native Nexus format, use the `File→Save` or `File→Save As` menu items. To save selected blocks of data, or to export data in a different file format, use the `File→Export`. A picture of the computed network can be saved using the `File→Export Image` menu item.

7 Estimating Distances

Many methods in phylogenetics begin by estimating distances between the taxa. This is done by taking sequences two at a time and estimating the average number of mutations that occurred on

the paths from them to their most recent common ancestor. If the rate of mutation was constant, then this will be proportional to their divergence time. SplitsTree4 provides a large number of methods for estimating distances, for sequences and other types of data.

The UncorrectedP method computes the proportion of positions at which two sequences differ. For DNA or RNA sequences, there are choices over how ambiguous state codes (such as W, M, V) are handled. Ignore, means that these states are treated as missing states. 'Average' means that the contribution at a site is averaged over all possible resolutions of the ambiguous codes, with the exception that sites having the same ambiguous code contribute zero. 'Match' looks at each possible state in each sequence, counts one if the state is not a possible resolution of the ambiguous code in the other sequence, and normalises the count by the number of states for each ambiguous code.

The Distances menu lists several standard distance estimation methods. Most of these have parameters that can be changed. When you select the menu item the Pipeline window will open and will display a panel for setting the parameters of the method. If you always use the same parameters for a given method, or if the method has no parameters, then selecting the "Don't show this dialog to configure this method again" will prevent the dialog from appearing again. All methods can be configured using the Pipeline window, accessed by Analysis→Configure Pipeline or Analysis→Configure Recent Methods .

8 Building and Processing Trees

The Trees menu provides a number of methods for constructing trees, both from distance and character data. The following distance-based methods are available: NJ, BioNJ, UPGMA, BunemanTree and RefinedBunemanTree.

The program provides 'front-ends' for several tree based programs, currently PhyML and PhylipParsimony. SplitsTree4 will pass these programs the current sequences, and place any trees produced in the SplitsTree4 Trees block. Our distribution of SplitsTree4 does not provide any external programs, so you must install them separately. In the dialog that is associated with such a method you must specify the location of the external application, e.g. using the PhyML Path text field. Once entered, the program will remember this entry as the default value.

SplitsTree4 provides a number of methods for processing a collection of trees. The Trees→TreeSelector menu contains items to select individual trees or to compute a consensus of trees. If the Trees→TreeSelector is selected, then the Trees→Previous Tree and Trees→Next Tree items can be used to move from one tree to the next in the Trees block. Pressing the shift-key when selecting either of these methods will move to the first tree, or the last tree, respectively. The Trees→ConsensusTree can be used to compute the majority or strict consensus tree.

As with distances, when you select such a menu item the Pipeline window will open and will display a panel for setting the parameters of the method. If you always use the same parameters for a given method, or if the method has no parameters, then selecting the "Don't show this dialog to configure this method again" will prevent the dialog from appearing again. All methods can be configured using the Pipeline window, accessed by Analysis→Configure Pipeline or

Analysis→Configure Recent Methods .

9 Building and Drawing Networks

The Network menu provides methods for computing phylogenetic networks from character sequences, distances and trees.

Methods that compute a split network directly from character sequences provided in the Characters block are ParsimonySplits, MedianNetwork , MedianJoining and SpectralSplits. Note that the Median network method requires binary sequences. However, given DNA or RNA, this program will detect all sites that contain precisely two character-states and will build a Median network from these.

The MedianJoining method computes an unrooted network from binary sequences, DNA and other multi-state sequences. This is an implementation of the algorithm described in [4]. In the case of non-binary sequences, the resulting network will not be a split network.

Two methods for computing split networks from distances provided in the Distances block are SplitDecomposition and NeighborNet.

If a set of phylogenetic trees in the Trees block all contain the full set of taxa listed in the Taxa block, then the ConsensusNetwork method can be applied to compute a consensus network. If, however, the Trees block contains partial trees, that is, trees that do not necessarily all involve identical sets of taxa, then the SuperNetwork or FilteredSuperNetwork method can be used to compute a super network.

The *Draw* menu determines which algorithm is used to construct the final visualization of the tree or network. Existing methods are EqualAngle, RootedEqualAngle and Phylogram. Additionally, the Draw→Hide Selected Splits can be used to remove selected splits from the network and the Draw→Select Trees can be used to highlight different trees contained in a split network.

10 Main Window

With SplitsTree4 , multiple documents can be opened and worked on simultaneously. Each document is represented by a Main window. Additional windows or dialogs are sometimes opened to perform certain tasks.

The Main window is split vertically into two panels:

- the left-hand Data panel is used to display all data associated with a given document in text format, and
- the right-hand Network panel contains the current tree or network.

We now discuss each separately:

10.1 Network Tab

The `Network` tab is used to display the computed tree or network. A status line of text along the bottom of the network pane gives a summary of the data, such as number of taxa, length of sequences, etc, and a summary of the methods used to compute the network. Optionally, a scale bar can be displayed in the upper left corner of the Network-tab to indicate the scale of branches. The scale bar is can be turned on or of using the `Preferences:General` tab. The scale bar can be configured using the `Preferences:Status Line` tab.

10.2 Data Tab

The `Data` tab provides a textual display of the data associated with the given document in the program's native Nexus format, organized in a linear list of items that can be either collapsed or expanded. This view of the data is read-only.

10.3 Tool bar

The tool bar associated with the window provides buttons for quick access to many of the menu items. It can be configured using the The `Preferences:Toolbar` tab.

11 Graphical Interaction with the Network

We now describe how the user can interactively modify the layout and attributes of the displayed network. The `View` menu is used to rotate, move, and zoom in and out. Alternatively, the view can be changed using a wheel mouse, together with the following modifier keys:

none	zoom in and out.
Shift	rotate
ALT/option	move vertically
Shift & ALT/option	move horizontally

Here are additional modifications that can be performed:

- The graph can be dragged around by clicking and dragging a node.
- If all selected nodes lie on one side of a band of parallel branches representing a single split, then clicking and dragging on one of the nodes will change the angle of the branches.
- By default, clicking on an edge will select *all* edges representing the same split, and all nodes on the smaller side of the represented split. To select all nodes on the other side of the split, use the `View→Invert Selection` menu item. To allow selection of individual edges, deselect the `Split-Selection Mode` check box in the `Preferences:General` tab.
- Select a node by clicking on it. Press the mouse button outside of the network and drag a rectangle to select several nodes at once. Hold the shift key and click to add or remove further nodes from the set selected.

- Click on a text label to edit it.
- Double-click on an edge to edit its label.
- By default, node labels are automatically positioned to avoid overlap. Any label that has been interactively reposition by the user is no longer automatically positioned. To apply automatic positioning all node labels, including those that have been moved by the user, deselect and then select the `View→Node Label Layout→No Overlaps` check box.
- Many aspects of the visual representation of nodes and edges can be modified using the `Format Nodes and Edges` window, which is opened using the `View→Format Nodes and Edges` menu item.

12 Main Menus

We now discuss all menus of the `Main` window.

12.1 File Menu

The `File` menu contains the usual file-related items:

- The `File→New` item opens a new `Main` window.
- The `File→Open` item provides an `Open File` dialog to open a file containing input data in one of the supported formats.

  If the file contains character sequences, then the program must know whether the sequences are DNA, RNA, protein or "standard" (0,1) data. In Nexus files, the datatype is explicitly given. In other file formats this not always the case. If the program cannot guess the data type (e.g., if all character states are 'A'), then the program will display a `Choose Datatype` dialog and prompt the user to specify the data type.

  If the current document is non-empty, then the selected file is opened in a new `Main` window.
- The `File→Open Recent` submenu provides access to recently opened or saved files.
- The `File→Replace` item is used to replaced the current data by a new data set.
- The `File→Clone` item clones the current window.
- The `File→Enter Data` item opens the `Data Entry` dialog, which can be used to enter data by hand (or copy-and-paste) in a number of different formats.
- The `File→Close` item closes the current document.
- The `File→Save` item saves the current document to its file, if known.
- The `File→Save As` item provides a `Save As` dialog and saves the current document to selected file.

- The File→Export item opens the Export dialog which is used to save individual data in a number of different formats.
- The File→Export Image item opens the Export Image dialog which is used to save the current network in a number of different formats.
- The File→Tools item provides a submenu of tools. The File→Tools→Load Trees item can be used to merge a set of trees into one document. This item opens a dialog for opening multiple files. The program parses all given files and extracts any trees found in them and opens a new Document containing all trees found. The File→Tools→Load Multi-Labeled Tree item can be used to read a single tree containing multiple labels and to convert it into a single-labeled split network (and will be made available upon publication of the paper describing this new method). The File→Tools→Concatenate Sequences item can be used to concatenate sequences from different files into one document. It requires that each of the input files uses exactly the same set of taxon labels. The File→Tools→Group Identical Haplotypes tools is used to detect multiple identical sequences or taxa. These are recognised from the Characters or, if there is no Characters block, from the distances block. Only selected sites are used to test if two taxa are distinguishable. A new document is created; identical taxa are collapsed into a single taxa with label 'TYPEx' (x ranging from $1, 2, 3, \ldots$, and the original taxa stored in the info field of the Taxa block. Note that files with hundreds or thousands of identical sequences can cause SplitsTree to stall. One solution is to cancel the computations after data is read in, and to then use this tool to create a new smaller document containing only distinct sequences.
- The File→Print item is used to print the current network.
- The File→Quit item quits the program, after asking whether to save unsaved changes.

12.2 Edit Menu

The Edit menu contains the usual edit-related items:

- The Edit→Undo item is used to undo text-editing, interactive network manipulation and any item chosen from the View menu.
- The Edit→Redo item is used to redo text-editing, interactive network manipulation and any item chosen from the View menu.
- The Edit→Cut item is used to cut text.
- The Edit→Copy item is used to copy text or to copy the current network as an image.
- The Edit→Paste item is used to paste text.
- The Edit→Select All item is used to select all nodes and edges of a network.
- The Edit→Select Nodes item is used to select all nodes of a network.
- The Edit→Select Labeled Nodes item is used to select all labeled nodes of a network.

- The `Edit→Select Edges` item is used to select all edges of a network.
- The `Edit→Invert Selection` item is used to invert the selection of nodes.
- The `Edit→Deselect All` item is used to deselect all nodes and edges of a network.
- The `Edit→Deselect Nodes` item is used to deselect all nodes of a network.
- The `Edit→Deselect Edges` item is used to deselect all edges of a network.
- The `Edit→Find/Replace` item opens the `Find/Replace` dialog.
- The `Edit→Preferences` opens the `Preferences` window.

12.3 View Menu

The `View` menu contains items that control aspects of the visualization of a network, which are all undo-able and redo-able:

- The `View→Data` item provides a submenu of items that are enabled when the `Data` tab is selected.
  - The `View→Data→Characters`, `View→Data→Distances` and `View→Data→Splits` submenus control the format used to write the `Characters`, `Distances` and `Splits` blocks, respectively.
- The `View→Reset` item resets the layout of the network.
- The `View→Zoom In` item is used to zoom into the network.
- The `View→Zoom Out` item is used to zoom out from the network.
- The `View→Rotate Left` and `View→Rotate Right` items are used to rotate the network.
- The `View→Flip` item changes the layout of the network to it's mirror image.
- The `View→Format Nodes and Edges` item opens the `Format Nodes and Edges` window that is used to modify the graphical attributes of the nodes and edges of the network.
- The `View→Highlight Confidence` item opens the `Confidence` window that can be used to control how confidence values for splits or edges are highlighted in the network.
- The `View→Use Magnifier` item is used to turn the magnifier functionality on and off.
- The `View→Magnify All Mode` item modifiers the magnification process so that the whole network or tree gets mapped into the magnifier.
- The `View→Node Label Layout→No Overlaps` check box item controls whether node labels are automatically placed so as to prevent any overlaps. By default, the program tries computing a label layout in 10 different random orders of the nodes. To change this number to 99, say, type `setprop label-layout-iterations=99` using the `Window→Enter a Command` item.

- The View→Node Label Layout→Radial check box item controls whether node labels are displayed in a radial fashion. At present, there is no way to change the individual angles used in the display.
- The View→Node Label Layout→Simple check box item controls whether a simple node label layout is used.

12.4 Data Menu

The Data menu contains the following items:

- The Data→Keep Only Selected Taxa item removes all taxa from the analysis, whose nodes in the network are unselected. All data associated with the removed taxa is removed from the original source block and all subsequent data is recomputed.
- The Data→Exclude Selected Taxa item removes all taxa from the analysis, whose nodes in the network are selected. All data associated with the removed taxa is removed from the original source block and all subsequent data is recomputed.
- The Data→Restore All Taxa item restores all taxa that where previously removed using either of the previous two menu items, or by the next menu item.
- The Data→Filter Taxa item opens the Pipeline:Taxa:Filter tab that can be used to interactively include or exclude taxa from the analysis.
- The Data→Taxon Sets submenu is used to select all nodes labeled by a given taxon set or to add new taxa sets to the Sets block. The Sets block can be copied to other files to provide a convenient method to quickly label or select taxa in different taxa groups. The Data→TaxonSets→All item selects all taxa. The Data→TaxonSets→New taxa set... item creates a new taxa set with the currently selected taxa and a name provided by the user. Note that the name should be a valid NEXUS name. If a set exists with this name, it is overwritten. The Data→TaxonSets→Clear All Taxa Sets... removes all taxa sets from the Sets block.
- The Data→Exclude Gap Sites item excludes from computation all sites in a Characters block that contain a gap or missing character in any of the sequences.
- The Data→Exclude Parsimony-Uninformative Sites item excludes from computation all sites in a Characters block that are parsimony-uninformative, that is, which are constant across all but one sequence.
- The Data→Exclude Constant Sites item excludes from computation all sites in a Characters block that are constant across all sequences.
- The Data→Restore All Sites item restores all sites that where excluded using the above menu items.
- The Data→Filter Characters item opens the Pipeline:Characters:Filter tab that can be used to interactively include or exclude sites from the analysis.

- The Data→Greedily Make Compatible item uses a greedy approach to makes the splits in the Splits block compatible: in decreasing order of weight, the algorithm adds the next split to the set of kept splits, if it is compatible with all splits that have already been kept.
- The Data→Greedily Make Weakly Compatible item uses a greedy approach to makes the splits in the Splits block weakly compatible: in decreasing order of weight, the algorithm adds the next split to the set of kept splits, if it is weakly-compatible with all splits that have already been kept.
- The Data→Exclude Selected Splits item removes all splits from the analysis, whose edges in the network are selected. The Splits and Network block are modified accordingly. See also: Draw→Hide Selected Splits .
- The Data→Restore All Splits item restores all splits that were excluded using the previous menu item.
- The Data→Filter Splits item opens the Pipeline:Splits:Filter tab that can be used to interactively include or exclude splits from the analysis.
- The Data→Filter Trees item opens the Pipeline:Trees:Filter tab that can be used to interactively include or exclude trees from the analysis.
- The Data→Set Tree Names item opens a dialog that allows one to set the tree names.

12.5 Distances Menu

The Distances menu contains the following items:

- The Distances→UncorrectedP item requests the program to compute distances using the UncorrectedP method.
- The Distances→LogDet item requests the program to compute distances using the LogDet method.
- The Distances→HKY85 item requests the program to compute distances using the HKY85 method.
- The Distances→JukesCantor item requests the program to compute distances using the JukesCantor method.
- The Distances→K2P item requests the program to compute distances using the K2P method.
- The Distances→K3ST item requests the program to compute distances using the K3ST method.
- The Distances→F81 item requests the program to compute distances using the F81 method.

- The `Distances→F84` item requests the program to compute distances using the `F84` method.
- The `Distances→ProteinMLdist` item requests the program to compute distances using the `ProteinMLdist` method.
- The `Distances→NeiMiller` item requests the program to compute distances using the `NeiMiller` method.
- The `Distances→GeneContentDistance` item requests the program to compute distances using the `GeneContentDistance` method.

12.6 Trees Menu

The `Trees` menu contains the following items:

- The `Trees→NJ` item requests the program to compute a tree using the `NJ` method.
- The `Trees→BioNJ` item requests the program to compute a tree using the `BioNJ` method.
- The `Trees→UPGMA` item requests the program to compute a tree using the `UPGMA` method.
- The `Trees→BunemanTree` item requests the program to compute a tree using the `BunemanTree` method.
- The `Trees→RefinedBunemanTree` item requests the program to compute a tree using the `RefinedBunemanTree` method.
- The `Trees→TreeSelector` item requests the program to compute a tree using the `TreeSelector` method.
- If the `Trees→TreeSelector` is selected, then the `Trees→Previous Tree` and `Trees→Next Tree` items are used to move from one tree to the next in the `Trees` block. Pressing the shift-key when selecting either of these methods will move to the first tree, or the last tree, respectively.
- The `Trees→ConsensusTree` requests the program to compute compute the majority or strict consensus tree.
- The `Trees→PhylipParsimony` item requests the program to compute a tree using the `PhylipParsimony` method.
- The `Trees→PhyML` item requests the program to compute a tree using the `PhyML` method.

12.7 Network Menu

The `Network` menu contains the following items:

- The `Networks→NeighborNet` item requests to compute a network using the `NeighborNet` method.

- The `Networks→SplitDecomposition` item requests to compute a network using the SplitDecomposition method.
- The `Networks→ParsimonySplits` item requests to compute a network using the ParsimonySplits method.
- The `Networks→ConsensusNetwork` item requests to compute a network using the ConsensusNetwork method.
- The `Networks→SuperNetwork` item requests to compute a network using the SuperNetwork method.
- The `Networks→FilteredSuperNetwork` item requests to compute a network using the FilteredSuperNetwork method.
- The `Networks→MedianNetwork` item requests to compute a network using the MedianNetwork method.
- The `Networks→MedianJoining` item requests to compute a network using the MedianJoining method.
- The `Networks→MinSpanningNetwork` item requests to compute a network using the MinSpanningNetwork method.
- The `Networks→SpectralSplits` item requests to compute a network using the SpectralSplits method.

12.8 Analysis Menu

The `Analysis` menu contains the following items:

- The `Analysis→Bootstrap` item opens the Bootstrap dialog.
- The `Analysis→Show Bootstrap Network` item opens a new Main window depicting a network that is based on all splits that occurred in any of the bootstrap replicates. Note that this item is enabled only after bootstrapping has been completed.
- The `Analysis→Show Confidence Network` item opens a new Main window containing a network that represents a 95% confidence set for the trees or networks estimated. Note that this item is enabled only after bootstrapping has been completed.
- The `Analysis→Estimate Invariable Sites` uses the capture-recapture method of [26].
- The `Analysis→Compute Phylogenetic Diversity` is enabled when a set of taxa are selected. It computes the sum of the weights for all splits that separate these taxa into two non-empty groups. On a tree, this is equivalent to the phylogenetic diversity measure of [12], as these splits will be exactly those lying on a path between two taxa.

- The `Analysis→Compute delta scores` is enabled when a set of taxa are selected and there is a valid distances block. It computes the average of the delta scores for all quartets selected from that set of taxa, see [19]. The averages for each individual taxon are printed in the message menu. The delta score is computed using the distances block.
- The `Analysis→Conduct Phi test for Recombination` tests for recombination using the Phi test of [6].
- The `Analysis→Configure Pipeline` item opens the `Pipeline` window which can be used to configure the parameters of a given method.
- The `Analysis→Configure Recent Methods` submenus lists all recently used methods and can be used to open the `Pipeline` window in the appropriate tab to configure the parameters of a given method.

12.9 Draw Menu

The `Draw` menu contains the following items:

- The `Draw→EqualAngle` item requests to draw a network using the `EqualAngle` method.
- The `Draw→RootedEqualAngle` item requests to draw a rooted network using the `RootedEqualAngle` method.
- The `Draw→ConvexHull` item requests to draw a network using the `ConvexHull` method.
- The `Draw→Phylogram` item requests to draw a tree using the `Phylogram` method.
- The `Draw→NoGraph` item requests that no network be computed.
- The `Draw→Reroot` item requests that the selected edge becomes the new root of a tree in a rooted tree display.
- The `Draw→Select Characters` item opens the `Pipeline:Characters:Select` tab that can be used to select individual characters (that is, sites in the given sequence alignment) to be displayed on the nodes of the network. set of selected input trees. If one or more splits are selected in the main viewer, then pressing on the `Select Supporting Characters` button will make the program select all characters that support the selected split or splits, that is, all characters for which the character states are different on both sides of the split, and constant one at least one side of the splits.
- The `Draw→Hide Selected Splits` item can be used to remove all selected splits directly from the depicted network. This does not change the `Splits` block. See also: `Data→Exclude Selected Splits` .
- The `Draw→Hide Non-Selected Splits` item can be used to remove all non-selected splits directly from the depicted network. This does not change the `Splits` block. See also: `Data→Exclude Selected Splits` .

- The `Draw→Hide Incompatible Splits` item can be used to remove all splits that are incompatible with the set of currently selected splits directly from the depicted network. This does not change the `Splits` block.
- The `Draw→Redraw All Splits` item redraws the network using all splits, including those excluded using the previous menu item.
- The `Draw→Select Trees` item opens the `Pipeline:Trees:Select` tab that can be used to select all splits in a given network that are contained in a given set of selected input trees.

12.10 Window Menu

The `Window` menu contains the following items:

- The `Window→Set Window Size` item allows one to set the width and height of the window.
- The `Window→Enter a Command` item allows entry of a single command.
- The `Window→Message Window` item opens the message window.

The bottom of the `Window` menu contains a list of all open windows.

12.11 Help Menu

The `Help` menu contains the following items:

- The `Help→About` item shows information about the version of SplitsTree4 . Under MacOS, this item is found in the SplitsTree menu.
- The `Help→How to Cite` item gives instructions on how to cite the program.
- The `Help→Nexus Syntax` submenu provides the syntax of all Nexus blocks that SplitsTree4 can process.
- The `Help→Command Syntax` item summarizes the syntax of the commands that can be used with the command-line mode version of the program, or which can be present in an input file.
- The `Help→Website...` item opens the program's website in a web browser.
- The `Help→Reference Manual...` item opens the program's reference manual in a web browser.
- The `Help→Check For Updates...` item checks for a new update to the program.

12.12 Configuring Methods

By default, selecting a menu item that applies some method to the given data, such as e.g. the Trees→NJ , will open up a dialog in the Pipeline window which one can use to configure the method by selecting appropriate options. For methods that have no options, or that are always used with the same options, selecting the *Don't show this dialog for this method again* checkbox will tell the program not to open the dialog in the future, but rather to immediately apply the named method. Note that configuration dialogs hidden in this way can still be accessed using the Analysis→Configure Recent Methods submenu.

13 Popup Menus

Right-clicking on the Main window will open a popup menu. There are three different menus that will appear, depending upon what is hit by the mouse click.

If the mouse is clicked on a node of a network, then this opens the Node popup menu, which has the following items:

- The Copy Label copies the node label to the system clipboard.
- The Edit Label opens a dialog to edit the current node label.
- The Exclude Selected Taxa excludes the selected taxa from all computations.
- The Show Name labels the selected node by the names of any correspond taxa.
- The Show Id labels the selected node by the ids of any correspond taxa.
- The Hide Label hides the label of the selected node.
- The Format opens the Format Nodes and Edges window.

If the mouse is clicked on a edge of a network, then this opens the Edge popup menu, which has the following items:

- The Copy Label copies the edge label to the system clipboard.
- The Edit Label opens a dialog to edit the current edge label.
- The Show Id labels the selected edge by the id of the corresponding split.
- The Show Weight labels the selected edge by the weight of the corresponding split.
- The Show Confidence labels the selected edge by the confidence of the corresponding split.
- The Show Interval labels the selected edge by the confidence interval of the corresponding split.
- The Hide Label hides the label of the selected edge.

- The `Format` opens the `Format Nodes and Edges` window.

If the mouse is clicked on the drawable region of the window, but not on any node or edge, then the `Window` popup menu will open, which has the following items:

- The `Zoom In` item is a short-cut to the `View→Zoom In` menu item.
- The `Zoom Out` item is a short-cut to the `View→Zoom Out` menu item.
- The `Reset` item is a short-cut to the `View→Reset` menu item.
- The `Reset Label Positions` item resets all node labels to their default positions.
- The `Restore All Taxa` item restores all excluded taxa.
- The `Restore All Splits` item restores all excluded splits.
- The `Set Window Size` item can be used to set the size of the `Main` window.

14 Tool Bar

For easier access to frequently used options and methods, SplitsTree4 provides a *tool bar* at the top of the `Main` window. The tool bar can be configured using the `Preferences:Toolbar` tab. By default, the tool bar contains the following items: `File→Open`, `File→Clone`, `File→Save As`, `File→Print`, `File→Export Image`, `Edit→Find/Replace`, `View→Reset`, `View→Zoom In`, `View→Zoom Out`, `View→Rotate Right`, `View→Rotate Left`, `Edit→Preferences`, `Window→Message Window`, `Draw→EqualAngle`, `Draw→RootedEqualAngle`, `Draw→Phylogram`, `Trees→Previous Tree` and `Trees→Next Tree`.

15 Data Entry Dialog

The `Data Entry` dialog can be used to enter data by hand or by copy-and-paste, in any of the file formats that the program supports, see Section 21.

16 Pipeline Window

The `Pipeline` window is accessed using the `Analysis→Configure Pipeline` item. It controls all aspects of the computational pipeline that SplitsTree4 uses. It is organized into nine tabs, reflecting the order of the blocks in the pipeline: `Pipeline:Taxa`, `Pipeline:Unaligned`, `Pipeline:Characters`, `Pipeline:Distances`, `Pipeline:Quartets`, `Pipeline:Trees` and `Pipeline:Splits`.

This is a potential source of confusion, because in the main menus of the `Main` window we organize the methods by the type of their *output*, rather than by the type of their *input*, as is done here.

Each tab in controls how a block of the given type is processed to produce a block of a subsequent type. Each of the tabs contains upto three sub-tabs, as mentioned below. Here we now describe each of the main tabs:

16.1 Taxa Tab

The `Pipeline:Taxa` tab consists of precisely one `Pipeline:Taxa:Filter` sub-tab, which is used to include (show) or exclude (hide) taxa from all calculations. All taxa listed in the *show list* are included in all calculations, where as all taxa listed in the *hide list* are removed from the data set. If a set of taxa are selected in the network displayed in the `Main` window, then these can be added to the show or hide list by pressing the appropriate `From Graph` button. Press `Show All` or `Hide All` to show all hide all known taxa.

Please note that the set of "shown" or "hidden" taxa will change automatically when viewing different partial trees to reflect the set of taxa contained in the current tree.

16.2 Unaligned Tab

The `Pipeline:Unaligned` tab consists of precisely one sub-tab. The `Pipeline:Unaligned:Method` sub-tab is used to choose and configure a method to apply to the current `Unaligned` block. Currently, the program provides three methods:

- The `NoAlign` item simply adds gaps to the ends of sequences to make the all have the same length.
- The `ClustalW` item can be used to call the program ClustalW externally to compute an alignment of the sequences.
- The `Muscle` item can be used to call the program Muscle externally to compute an alignment of the sequences.

16.3 Characters Tab

The `Pipeline:Characters` tab has three sub-tabs. The `Pipeline:Characters:Method` sub-tab is used to choose and configure a method to apply to the current `Characters` block. See Section 22 for a description of all available methods.

The `Pipeline:Characters:Filter` sub-tab is used to exclude certain sites from the analysis. By default, the sites remain present in the `Characters` block and are simply masked during calculations. To remove the sites completely, press the `Delete` button.

The `Pipeline:Characters:Select` sub-tab is used to select certain sites in the `Characters` block for display of sites in the network. If you have selected a split in the main viewer, and if you press `Select Supporting Characters`, then SplitsTree will fill the text box on the left of the dialog with all sites that support the selected split. These are all characters for which the set of character states on the one side of the split is disjoint from the set of character states on the other side of the split. In this calculation, gaps and missing data are ignored. If there are no non-gap and

non-missing data states on one side of a split, then the other side must show only one state and is not allowed to have any gap or missing data state. After determining the supporting characters, then press `Show on All Taxa` or `Show on Selected Taxa` to see the character states in the network view.

16.4 Distances Tab

The `Pipeline:Distances` tab has only one sub-tab. The `Pipeline:Distances:Method` sub-tab is used to choose and configure a method to apply to the current `Distances` block. See Section 22 for a description of all available methods.

16.5 Quartets Tab

The `Pipeline:Quartets` tab has only one sub-tab. The `Pipeline:Quartets:Method` sub-tab is used to choose and configure a method to apply to the current `Quartets` block. See Section 22 for a description of all available methods.

16.6 Trees Tab

The `Pipeline:Trees` tab has three sub-tabs. The `Pipeline:Trees:Method` sub-tab is used to choose and configure a method to apply to the current `Trees` block. See Section 22 for a description of all available methods.

The `Pipeline:Trees:Filter` sub-tab is used to exclude trees sites from the analysis.

The `Pipeline:Trees:Select` sub-tab is used to highlight a set of trees in the displayed network by selecting all splits in the network that are contained in the set of chosen trees.

16.7 Splits Tab

The `Pipeline:Splits` tab has two sub-tabs. The `Pipeline:Splits:Method` sub-tab is used to choose and configure a method to apply to the current `Splits` block. See Section 22 for a description of all available methods.

The `Pipeline:Splits:Filter` sub-tab is used to modify the weights of splits using a *least squares* optimization, or to exclude certain splits from the analysis. Possible filters are:

`Greedy Compatible`	uses a greedy approach to makes the splits compatible: in decreasing order of weight, the algorithm adds the next split to the set of kept splits, if it is compatible with all splits that have already been kept.
`Closest Tree`	makes the splits compatible by computing the "closest tree"
`Greedy Weakly Compatible`	uses a greedy approach to makes the splits weakly compatible: in decreasing order of weight, the algorithm adds the next split to the set of kept splits, if it is weakly-compatible with all splits that have already been kept.
`Weight Threshold`	removes any split whose weight does not exceed the given threshold. Pressing the `Set` button will open a histrogram and slider to set the threshold.
`Confidence Threshold`	removes all splits whose confidence does not exceed the given threshold. A confidence value usually lies in the range $[0, 1]$ and can be obtained by bootstrapping, for example. Pressing the `Set` button will open a histrogram and slider to set the threshold.
`Set Maximum Dimension`	greedily removes a subset of splits that cause boxes in the network of dimension higher than the given threshold, as described in [23].
`None`	applies no filter.

Use the `Exclude Selected Splits` to remove all splits that are currently selected in the displayed network.

17 Export and Export Images Dialogs

17.1 Export

The `Export` dialog is opened using the `File→Export` item. It is used to save individual blocks of data in any of the formats described in Section 21. The dialog show two lists. On the left, the set of available Nexus blocks. Is listed. Depending on the selection of blocks made by the user, on the right, the set of available export formats is listed.

Most of the dependences between blocks and formats should be apparent. One interesting feature is the following. If one has enabled graph editing using the Allow Graph Editing option in the `Preferences:General` tab and has interactively constructed a phylogenetic tree, then this tree can be saved in Newick format. Hence, SplitsTree4 can be used for editing trees.

17.2 Export Image

The `Export Image` dialog is opened using the `File→Export Image` item. This dialog is used to save a picture of the current network in a number of different formats. The following graphics formats are supported:

- JPEG, "Joint Photographic Experts Group".
- GIF, "Graphics Interchange Format".
- EPS, "Encapsulated PostScript".
- SVG, "Scalable Vector Graphics".
- PNG, "Portable Network Graphics".
- BMP, "Bitmap".
- PDF, "Portable Document Format".

There are two radio buttons, the `Save whole image` to save the whole image, and the `Save visible image` to save only the part of the image that is currently visible in the main viewer. If the chosen format is EPS, then selecting the `Convert text to graphics` check box will request the program to render all text as graphics, rather than fonts.

Pressing the apply button will open a standard file save dialog to determine where to save the graphics file.

18 Preferences Window

The `Preferences` window is opened using the `Edit→Preferences` item.

This window controls program *preferences*. Many of the preferences are persistent, that is, they remain effective even after closing and re-opening the program. To "remember" these choices, SplitsTree4 creates and maintains a *properties file*, which is placed in the users home directory and is called something like *.SplitsTree.def*. It has six tabs.

The `Preferences:General` tab controls general aspects of the program:

Allow Graph Editing	if set, the user can add or delete nodes and edges from displayed network. To create a node, control-double click on the canvas. To create an edge between two nodes, control-click on a node and then drag create an edge. If dragging ends on a second node, then this is connected to the first, otherwise, a new node is created. To delete a node or delete an edge select it and press the delete key.
`Lock Edge Lengths`	Usually, when interacting with a displayed network, we want to lock edge lengths so that they cannot be changed interactively..
`Show Scale Bar`	The program uses a small *scale bar* in the top left hand corner of the `Network` tab to indicate the scale of the network.
`use Split-Selection Mode`	When this feature is on, clicking on an edge will select all edges that correspond to the same split.

Any choices made here can either be applied to the current document or can be made the default for all subsequently opened documents.

The `Preferences:Defaults` tab controls the default sizes, shapes, colors and fonts for nodes and edges, see also `Format Nodes and Edges`.

The `Preferences:Layout` tab controls aspects of the layout of trees and split networks:

`Recompute`	The taxon layout is recomputed each time the `Splits` block is modified.
`Stabilize`	Use the union of the set of current splits and of the previously computed splits to compute a taxon layout. This is useful when one wants to compare two different networks computed on the same taxon set: by switching back and forth between the two computations, the program will find a joint taxon layout for both graphs, if one exists. In this case, both graphs will be laid out in such a way that in both graphs the taxa appear in approximately the same region of the canvas.
`Snowball`	Similar to the previous method, this method uses all splits ever computed in a given dataset to produce a taxon layout. This doesn't work very well.
`Use this layout`	Use the provided taxon layout to draw the network.

The `Preferences:Toolbar` tab allows the user to interactively configure the tool bar.

The `Preferences:Status Line` tab allows the user to configure the *status line* displayed along the bottom of the `Network` tab.

Here, two items are of particular interest. If the displayed graph was computed either by the `BunemanTree` or `SplitDecomposition` method, then the pair-wise distances in the graph may under estimate the true distances in the given `Distances` block. The difference between the two can be expressed in terms of the *fit* value, which is activated using the `Fit` check box, which is based on the sum of all pairwise absolute differences between given distances and split distances,

divided by the total sum of given distances. This is given by

$$\left(1 - \frac{\sum_{ij}(d_{ij} - p_{ij})}{\sum_{ij} d_{ij}}\right) \times 100,$$

where the split distances are p_{ij} and the given distances are d_{ij}.

As an alternative, you can also use the `LSFit` item, which computes the *least squares fit* between the pairwise distances in the graph and the pairwise distances in the matrix.

19 Additional Windows and Dialog

Here we list all other windows.

19.1 Open File

The `Open File` dialog is opened using the `File→Open` item. Use it to open any file containing phylogenetic data in one of the formats described in Section 21.

If an error is encountered, then the file is opened in the `Main`Source tab. If possible, the offending line is highlighted.

19.2 Choose Datatype

When opening a file containing character sequences, or importing sequences from the `Data Entry` dialog, the program must know whether the sequences are to be interpreted as DNA, RNA, protein or "standard" (0,1) data. In Nexus files, the datatype is explicitly given. In other file formats this is usually not the case. The program employs a simple heuristic to guess the datatype. If this fails (e.g., if all character states are 'A'), then the program will display a `Choose Datatype` dialog and prompt the user to specify the datatype.

The choices are:

- `dna`,
- `rna`,
- `protein`,
- `standard`, which means $0, 1$ data, and
- `unknown`, which the program cannot deal with.

19.3 Save As

The `Save As` dialog is opened using the `File→Save As` item. Its purpose is to save the complete state of a document in Nexus format. To save parts of the document in Nexus format, or in some other supported format listed in Section 21, use the `Export` dialog.

19.4 Find/Replace Window

The `Find/Replace` window can be opened using the `Edit→Find/Replace` item. It's purpose is to find strings in a displayed network, in the source tab or in the message window. It can also be used to replace text.

Enter a query in the top text region. Optionally, enter a replacement string below it. Use the following check boxes to parameterize the search:

- If the `Whole words only` item is selected, then only taxa or reads matching the complete query string will be returned.
- If the `Case sensitive` item is selected, then the case of letters is distinguished in comparisons.
- If the `Regular Expression` item is selected, then the query is interpreted as a Java regular expression.

The scope can be `global` or restricted to items that are already selected `selected`. The direction in which the next match is searched for can be selected using the `Forward` and `Backward` buttons.

Press the `Close`, `Find First` or `Find Next` buttons to close the dialog, or find the first, or next occurrence of the query, respectively. Press the `Find All` button to find all occurrences of the query.

Press the `Replace` or `Replace All` button to replace the next or all occurrences of the query.

The search can be applied to different targets:

- `Nodes` - search all node labels
- `Edges` - search among edge labels
- `Source` - search in the source tab
- `Messages` - search among text in the Messages window.

Press the `From File` button to load a set of queries, one per line, from a file.

19.5 Format Nodes and Edges Window

The `Format Nodes and Edges` window is opened using the `View→Format Nodes and Edges` item. Its purpose is to modify the appearance of nodes and their labels, and edges and their labels, in the displayed network.

The first row of items is used to set the font , choosing the font famly, italics and bold. The second row of items is used to set the color of nodes and edges. The check boxes Line Color, Fill Color, Label Color and Label Fill Color determine which color features are set. The Color Chooser can be used to set a specific color. Use the Random Colors or Invisible button to use random colors or to set no color.

The third row of items is used to set the Node Size and Node Shape.

The fourth row of items is used to set the Edge Width and Edge Style.

The fifth row is used to set the node labels to Names or IDs, and also to rotate the node labels, using the buttons Rotate Left and Rotate Right.

The final row is used to set the edge labels to Weights, IDs, Confidence values or confidence Intervals.

Please note that any changes made *only apply* to the currently selected nodes or edges. If there are no edges or nodes selected, then changes will apply to *all* nodes and edges. Any change made in this dialog box can be reversed using Edit→Undo .

19.6 Highlight Confidence Window

The Highlight Confidence Window is opened using the View→Highlight Confidence window item.

Use this window to request that edges and/or edge shading is done to reflect the confidence values associated with each split.

19.7 Bootstrap Window

The Bootstrap window is opened using the Analysis→Bootstrap item. Enter the Number of Replicates and then press Run to execute.

19.8 Message Window

The Message window is opened using the Window→Message Window item. The program writes all messages to this window.

19.9 About Window

The About Window is opened using the Help→About item. It reports the version of the program.

20 Nexus Blocks

In this section we describe the *Nexus* format, as implemented in SplitsTree4 , based on the definition provided in [27]. Unfortunately, there exist two variants of the Nexus format, which we will call *old Nexus* and *new Nexus* . SplitsTree4 is based on the latter, as this is what is defined in [27]. Given a file formatted in old Nexus, the program will often be able to parse it as it contains code to automatically convert from old to new Nexus format. However, the algorithm that does this does not provide a full implementation of the old Nexus format and thus sometimes it will be necessary to reformat a file by hand.

It is easy to tell the difference between old Nexus and new Nexus: if a file in Nexus format contains a `Taxa` block, then it is new Nexus. Most blocks in both formats have the same name and similar syntax. One main difference is that the `Characters` block in new Nexus is called a `Data` block in old Nexus.

In the following syntax descriptions, we used upper case letters for keywords, square brackets for optional statements and curly brackets to indicate a list of choices.

20.1 Taxa Block

The `Taxa` block is the only mandatory *block* in a Nexus file. Its purpose is to list the names of all taxa. It has the following syntax:

```
BEGIN TAXA;
DIMENSIONS NTAX=number-of-taxa;
[TAXLABELS taxon_1 taxon_2 ... taxon_ntax;]
[TAXINFO info_1 info_2 ... info_ntax;]
END;
```

The `TAXLABELS` statement is optional, if it is followed by a *source* block that contains all taxa labels.

20.2 Unaligned Block

The `Unaligned` block contains unaligned sequences. It has the following syntax:

```
BEGIN UNALIGNED;
        [DIMENSIONS NTAX=number-of-taxa;]
        [FORMAT
            [DATATYPE={STANDARD|DNA|RNA|NUCLEOTIDE|PROTEIN}]
            [RESPECTCASE]
            [MISSING=symbol]
            [SYMBOLS="symbol symbol ..."]
            [LABELS={LEFT|NO}]
        ;]
        MATRIX
            [taxonlabel1] sequence ,
                [taxonlabel2] sequence ,
                  ...
                [taxonlabelN] sequence
        ;
END;
```

20.3 Characters Block

The `Characters` block contains aligned character sequences. It has the following syntax:

```
BEGIN CHARACTERS;
        DIMENSIONS [NTAX=number-of-taxa] NCHAR=number-of-characters;
        [FORMAT
            [DATATYPE={STANDARD|DNA|RNA|PROTEIN}]
            [RESPECTCASE]
            [MISSING=symbol]
            [GAP=symbol]
            [SYMBOLS="symbol symbol ..."]
            [LABELS={NO|LEFT}]
            [TRANSPOSE={NO|YES}]
            [INTERLEAVE={NO|YES}]
            [TOKENS=NO]
        ;]
        [CHARWEIGHTS wgt_1 wgt_2 ... wgt_nchar;]
        [CHARSTATELABELS character-number [character-name]
                [ /state-name [ state-name... ] ], ...;]
        MATRIX
            sequence data in specified format
        ;
END;
```

20.4 Distances Block

The `Distances` block contains a matrix of pairwise distances. It has the following syntax:

```
BEGIN DISTANCES;
        [DIMENSIONS [NTAX=number-of-taxa];]
        [FORMAT
            [TRIANGLE={LOWER|UPPER|BOTH}]
            [[NO] DIAGONAL]
            [LABELS={LEFT|NO}]
        ;]
        MATRIX
            distance data in specified format
        ;
END;
```

20.5 Quartets Block

The `Quartets` block contains a list of quartets. It has the following syntax:

```
BEGIN Quartets;
        DIMENSIONS [NTAX=number-of-taxa] NQUARTETS=number-of-quartets;
        [FORMAT
```

```
            [LABELS={LEFT|NO}]
            [WEIGHTS={YES|NO}]
        ;]
        MATRIX
        [label1] [weight1] a1 b1 : c1 d1,
...
        [labeln] [weightn] an bn : cn dn,
        ;
END;
```

20.6 Trees Block

The `Trees` block contains a list of phylogenetic trees. It has the following syntax:

```
BEGIN Trees
[PROPERTIES PARTIALTREES={YES|NO};]
[TRANSLATE
    nodeLabel1 taxon1,
    nodeLabel2 taxon2,
    ...
    nodeLabelN taxonN
;]
[TREE name1 = tree1-in-Newick-format;]
[TREE name2 = tree2-in-Newick-format;]
...
[TREE nameM = treeM-in-Newick-format;]
END;
```

20.7 Splits Block

The `Splits` block contains a list of splits. It has the following syntax:

```
BEGIN Splits;
        [DIMENSIONS [NTAX=number-of-taxa] [NSPLITS=number-of-splits];]
        [FORMAT
            [LABELS={LEFT|NO}]
            [WEIGHTS={YES|NO}]
            [CONFIDENCES={YES|NO}]
            [INTERVALS={YES|NO}]
        ;]
        [THRESHOLD=non-negative-number;]
        [PROPERTIES
            [FIT=non-negative-number]
            [leastsquares]
            [{COMPATIBLE|CYCLIC|WEAKLY COMPATIBLE|INCOMPATIBLE}]
```

```
        ;]
        [CYCLE [taxon_i_1 taxon_i_2 ... taxon_i_ntax];]
        [SPLITSLABELS label_1 label_2 ... label_nsplits;]
        MATRIX
            [label_1] [weight_1] [confidence_1] split_1,
            [label_2] [weight_2] [confidence_2] split_2,
            ....
            [label_nsplits] [weight_nsplits] [confidence_nsplits] split_nsplits,
        ;
END;
```

20.8 Network Block

The `Network` block contains the definition of a phylogenetic network. It has the following syntax:

```
BEGIN NETWORK;
        DIMENSIONS NTAX=number-taxa NVERTICES=number-vertices NEDGES=number-edges;
        [DRAW
            [ROTATE=rotation]
        ;]
        [TRANSLATE
            [vertex_1 taxon_1,
            vertex_2 taxon_2,
            ...
            vertex_ntax taxon_ntax,]
        ;]
        VERTICES
             1 x_1 y_1 [WIDTH=n] [HEIGHT=n] SHAPE=[RECT|OVAL][FGC=color]
                        [BGC=color] [LINE=n],
             2 x_2 y_2 [WIDTH=n] [HEIGHT=n] SHAPE=[rect|OVAL][FGC=color]
                        [BGC=color] [LINE=n],
             ...
             nvertices x_nvertices y_nvertices [WIDTH=n] [HEIGHT=n]
                       SHAPE=[RECT|OVAL][FGC=color] [BGC=color] [LINE=n],
        ;
        VLABELS
            vertex_id label [X= xoffset Y=yoffset] [FGC=color] [FONT=font],
            ...
            vertex_id label [X= xoffset Y=yoffset] [FGC=color] [FONT=font],
        EDGES
            1 vertex_id vertex_id [ECLASS=n] [FGC=color] [LINE=n],
            2 vertex_id vertex_id [ECLASS=n] [FGC=color] [LINE=n],
            ...
            nedges vertex_id vertex_id [FGC=color] [LINE=n],
        [ELABELS
```

```
            edge_id label [X= xoffset Y=yoffset] [FGC=color] [FONT=font],
            ...
            edge_id label [X= xoffset Y=yoffset] [FGC=color] [FONT=font],
        ;]
        [INTERNAL
            edge_id [x y] [x y] ...,
            ...
            edge_id [x y] [x y] ...,
        ;]
END;
```

20.9 Bootstrap Block

The `ST-Bootstrap` block contains the results of a bootstrap analysis. It has the following syntax:

```
BEGIN ST_BOOTSTRAP;
        [DIMENSIONS [NTAX=number-of-taxa] [NCHAR=number-of-characters]
                [NSPLITS=number-of-splits];]
        [FORMAT
            [LABELS={LEFT|NO}]
            [SPLITS={NO|YES}]
            [ALL={YES|NO}]
        ;]
        [RUNS=the-number-of-runs;]
        [LENGTH={sample-length | SAME}];
        [SEED=random-number-seed;]
        [SAVEWEIGHTS={yes|no};]
        [FIXSPLITS={yes|no};]
        [COMPUTEDIST={yes|no};]
        [OUTPUTFILE=file-name;]
        [MATRIX
            [label_1]  value_1  [split_1,]
            [label_2]  value_2  [split_2,]
            ....
            [label_nsplits]    value_nsplits  [split_nsplits,]
            [label_nsplits+1]  value_(nsplits+1)  [splits_(nsplits+1),]
            ....
            [label_n]  value_n  [splits_n,]
        ;]
END;
```

20.10 Sets Block

The `Sets` block can be used to define sets of taxa or characters. It has the following syntax:

```
BEGIN Sets;
        [TAXSET taxset-name    = taxon-list;]
        [CHARSET charset-name  = character-list;]
        [CHARPARTITION charpart-name = 1:charset-name|character-list [,...];
        ...
END;
```

20.11 ST Assumptions Block

The `ST-Assumptions` block controls the processing of the data along the pipeline. It contains all choices made by the user that affect computations. It has the following syntax:

```
BEGIN ST_ASSUMPTIONS;
        [UNALIGNTRANSFORM=name [parameters];]
        [CHARTRANSFORM=name [parameters];]
        [DISTTRANSFORM=name [parameters];]
        [SPLITSTRANSFORM=name [parameters];]
        [SPLITSPOSTPROCESS
                [[NO] LEASTSQUARES]
                [FILTER={GREEDYCOMPATIBLE|WEAKLYCOMPATIBLE|WEIGHT VALUE=value
                                |CONFIDENCE VALUE=valueDIMENSION VALUE=value|NONE};]
        [EXTAXA={NONE|list-of-original-taxa-labels};]
        [EXCHAR={NONE|list-of-original-char-positions};]
        [EXCLUDE [[NO] GAPS] [[NO] NONPARSIMONY]
                [{NO CONSTANT|CONSTANT [number]}]
                [[NO] CODON1] [[NO] CODON2] [[NO] CODON3];]
                [EXTREES={NONE|list-of-original-tree-labels};]
                [LAYOUTSTRATEGY={STABILIZE|SNOWBALL|KEEP};]
                [[NO] AUTOLAYOUTNODELABELS;]
        [UPTODATE;]
END;
```

21 File Formats

By default, SplitsTree4 reads and writes data in Nexus format. The program can read and export the following additional formats: FastA, Phylip and ClustalW:

Name	file suffix	data type	
new Nexus	`.nex`, `.nxs`	taxa, unaligned sequences, aligned sequences (characters), distances, quartets, trees, splits, networks	
old Nexus	`.nex`, `.nxs`	aligned characters, distances, trees	[27]
FastA	`.fa`, `.fasta`	unaligned sequences, or aligned characters	
Phylip	`.phy`, `.dst`, `.dist`	aligned characters or distances	[13]
ClustalW	`.aln`	aligned characters	[34]

22 All Methods

Here we list all methods supported by SplitsTree4 . We describe the usage and list the input and output Nexus blocks.

`Binary2Splits` : This method converts binary characters to splits.

```
Usage:  Binary2Splits AddAllTrivial=<boolean> MinSplitWeight=<int>
Input:  Characters
Output: Splits
```

`BioNJ` : This method computes the Bio-NJ tree [15].

```
Usage:  BioNJ
Input:  Distances
Output: Trees
```

`BunemanQuartets` : This method computes all quartets with positive Buneman index [8].

```
Usage:  BunemanQuartets threshold=<double>
Input:  Distances
Output: Quartets
```

`BunemanTree` : This method computes the Buneman tree [8].

```
Usage:  BunemanTree
Input:  Distances
Output: Splits
```

`ClustalW` : This method externally runs the ClustalW *sequence alignment* program [34].

```
Usage:  ClustalW GapOpen=<int> GapExtension=<double>
```

```
        WeightMatrix=<java.lang.String> OptionalParameter=<java.lang.String>
        PathToCommand=<java.lang.String>
Input:  Unaligned
Output: Characters
```

`Coalescent` : This method transforms a set of quartets to a set of splits representing a binary phylogenetic tree by applying the *coalescent method* described in [28].

```
Usage:  Coalescent
Input:  Quartets
Output: Splits
```

`ConsensusNetwork` : This method computes the consensus splits of trees [3, 18] to produce a *consensus network*.

```
Usage:  ConsensusNetwork Threshold=<double> EdgeWeights=<String>
Input:  Trees
Output: Splits
```

`ConsensusTree` : This method computes different types of *consensus trees*.

```
Usage:  ConsensusTree EdgeWeights=<String> Method=<String>
Input:  Trees
Output: Splits
```

`ConvexHull` : This method computes a splits graph using the convex hull extension algorithm [9].

```
Usage:  ConvexHull Weights=<boolean> ScaleNodesMaxSize=<int>
Input:  Splits
Output: Network
```

`DNA2Splits` : This method converts DNA characters to splits by setting the majority state against all other states.

```
Usage:  DNA2Splits AddAllTrivial=<boolean> MinSplitWeight=<int>
Input:  Characters
Output: Splits
```

`DQuartets` : This method compute all quartets with positive isolation index [2].

```
Usage:  DQuartets threshold=<double>
Input:  Distances
Output: Quartets
```

`EqualAngle` : This method computes a planar split network for a circular (sub-)set of splits [9]. If the `RunConvexHull` option is chosen, then the convex hull algorithm is subsequently applied

to obtain a graph for the complete set of splits. This method provides a number of heuristics for obtaining a better layout [14]: set `DaylighIterations` to ≈ 5 to apply the *equal daylight* heuristic, set `OptimizeBoxesIterations` to ≈ 10 to apply the *box opening* heuristic and set `SpringEmbedderIterations` to ≈ 500 to apply a modified spring embedder.

```
Usage:  EqualAngle UseWeights=<boolean> RunConvexHull=<boolean>
        DaylightIterations=<int> OptimizeBoxesIterations=<int>
        SpringEmbedderIterations=<int>
Input:  Splits
Output: Network
```

`FilteredSuperNetwork` : This method computes a *super-network* from *partial trees* using the *Z-closure* algorithm [23] and a *distortion filter* [25].

```
Usage:  FilteredSuperNetwork MinNumberTrees=<int> MaxDistortionScore=<int> EdgeWeights=<String
        AllTrivial=<boolean> UseTotalScore=<boolean>
Input:  Trees
Output: Splits
```

`F81` : This method calculates distances using the *Felsenstein-81* model [33].

```
Usage:  F81 Maximum_Likelihood=<boolean> Estimate_Base_Frequencies=<boolean>
        Base_Freqs=<N double1 double2 ... doubleN> Normalize=<boolean>
Input:  Characters
Output: Distances
```

`F84` : This method Calculates distances using the *Felsenstein-84* model [33].

```
Usage:  F84 Maximum_Likelihood=<boolean> Estimate_Base_Frequencies=<boolean>
        Normalize=<boolean> TRatio=<double> A=<double> C=<double> G=<double>
        T_U=<double>
Input:  Characters
Output: Distances
```

`GapDist` : This method calculates the "gap distance" from a set of sequences.

```
Usage:  GapDist
Input:  Characters
Output: Distances
```

`GeneContentDistance` : This method compute distances based on *gene content* [24].

```
Usage:  GeneContentDistance UseMLDistance=<boolean>
Input:  Characters
Output: Distances
```

`Hamming` : This method calculates distances using the Hamming distance. This is identical to the `UncorrectedP` method.

```
Usage:  Hamming ignoregaps=<boolean>
Input:  Characters
Output: Distances
```

`HKY85` : This method calculates distances using the *Hasegawa, Kishino and Yano model*.

```
Usage:  HKY85 Estimate_Base_Frequencies=<boolean> Normalize=<boolean>
        TRatio=<double> A=<double> C=<double> G=<double> T_U=<double>
        P_Invar=<double> Gamma=<double>
Input:  Characters
Output: Distances
```

`JukesCantor` : This method computes distances using the *Jukes Cantor model* [33].

```
Usage:  JukesCantor Maximum_Likelihood=<boolean>
Input:  Characters
Output: Distances
```

`K2P` : This method calculates distances using the *Kimura-2P* model [33].

```
Usage:  K2P Maximum_Likelihood=<boolean> TRatio=<double>
Input:  Characters
Output: Distances
```

`K3ST` : This method calculates distances using the *Kimura-3ST* model [33].

```
Usage:  K3ST Maximum_Likelihood=<boolean> TRatio=<double> AC_vs_ATRatio=<double>
Input:  Characters
Output: Distances
```

`LogDet` : This method Calculates the LogDet-distance [32].

```
Usage:  LogDet Impute_Gaps=<boolean>
Input:  Characters
Output: Distances
```

`LogHamming` : This method calculates distances using the log-Hamming distance.

```
Usage:  LogHamming ignoregaps=<boolean>
Input:  Characters
Output: Distances
```

`MedianNetwork` : This method computes an (unreduced) *median network* [3]. It uses all sites in the character alignment that contain exactly two different states, and no gaps or missing states. If `UseRYAlphabet` is selected, then DNA and RNA sequences are translated using $R = \{A, G\}$ and $Y = \{C, T, U\}$. If `UseRelaxedSupport` is selected, a character need only be constant on one side of a split to count toward the support of the split.

```
Usage:  MedianNetwork AddAllTrivial=<boolean> MinimalSupport=<int>
        UseRYAlphabet=<boolean> LabelEdges=<boolean> UseRelaxedSupport=<boolean>
Input:  Characters
Output: Splits
```

`MedianJoining` : This method computes a *median joining network* [4]. It uses all sites in the character alignment and treats gaps or missing data as additional sites. The parameter *epsilon* is used to control the level of homoplasy considered in the analysis.

```
Usage:  MedianJoining Epsilon=<int> SpringEmbedderIterations=<int> LabelEdges=<boolean>
        ShowHaplotypes=<boolean> SubdivideEdges=<boolean> ScaleNodesByTaxa=<boolean>
Input:  Characters
Output: Network
```

`PrunedQuasiMedian` : This method computes a *geodescically-pruned quasi-median network* [1]. It uses all sites in the character alignment and treats gaps or missing data as additional sites.

```
Usage:  PrunedQuasiMedian SpringEmbedderIterations=<int> LabelEdges=<boolean>
        ShowHaplotypes=<boolean> SubdivideEdges=<boolean> ScaleNodesByTaxa=<boolean>
Input:  Characters
Output: Network
```

`MinSpanningNetwork` : This method computes a *minimum spanning network* [11]. It computes the unnormalized Hamming distance between every pair of sequences. A spring embedder is used to compute a layout of the graph. The number of iterations required will vary by the size and complexity of the graph, and can be set using the `Spring Embedder Iterations` item. If `Subdivide Edges` is selected, then edges are divided into sub-edges, one for each change along the edge.

```
Usage:  MinSpanningNetwork Epsilon=<int> SpringEmbedderIterations=<int> LabelEdges=<boolean>
        ShowHaplotypes=<boolean> SubdivideEdges=<boolean> ScaleNodesByTaxa=<boolean>
Input:  Characters
Output: Network
```

`Muscle` : This method externally runs the Muscle sequence alignment program [10].

```
Usage:   Muscle Maxiters=<int> ClusterMethod_1=<String>
         ClusterMethod_2=<String> DistanceMeasure_1=<String>
         DistanceMeasure_2=<String> ObjectiveScore=<String>
         LogFile=<boolean> LogFileName=<String>
```

```
         OptionalParameter=<String> PathToCommand=<String>
Input:  Unaligned
Output: Characters
```

`NeiMiller` : This method computes distances from restriction-sites using the Nei and Miller method [29].

```
Usage:  NeiMiller
Input:  Characters
Output: Distances
```

`NJ` : This method computes the Neighbour-Joining tree [30].

```
Usage:  NJ
Input:  Distances
Output: Trees
```

`NeighborNet` : This method computes the *Neighbor-Net* splits [7] to produce a Neighbor-Net network.

```
Usage:  NeighborNet Variance=<String> Minimize_AIC=<boolean>
Input:  Distances
Output: Splits
```

`NoAlign` : This method obtains a trivial *sequence alignment* by adding gaps to the end of each sequence to make all sequences have the same length.

```
Usage:  Noalign
Input:  Unaligned
Output: Characters
```

`NoGraph` : This method prevents the program from constructing a final network.

```
Usage:  NoGraph
Input:  Splits
Output: Network
```

`NoSplits` : This method prevents the program from constructing splits from a set of trees.

```
Usage:  NoSplits
Input:  Trees
Output: Splits
```

`ParsimonySplits` : This method computes the set parsimony splits [2].

```
Usage:  ParsimonySplits
Input:  Characters
Output: Splits
```

`PhylipParsimony` : This method computes the *maximum parsimony* tree from DNA sequences using an external call to the Phylip package [13].

```
Usage:  PhylipParsimony PhylipPath=<String> SearchMode=<String>
        NumberOfTreesToSave=<int> InputOrderSeed=<int> InputOrderJumbles=<int>
        OutgroupRoot=<String> ThresholdParsimony=<double>
        TranversionParsimony=<boolean> WeightsFile=<String>
Input:  Characters
Output: Trees
```

`Phylogram` : This method computes a traditional phylogenetic tree.

```
Usage:  Phylogram Weights=<boolean> Cladogram=<boolean> Outgroup=<String>
        UseOutgroup=<boolean>
Input:  Splits
Output: Network
```

`PhyML` : This method calculates *maximum likelihood* trees from DNA sequences using PHYML [16].

```
Usage:  PhyML PHYMLPath=<String> TreePath=<String> Bootstrap=<boolean>
        NumberOfBootstrapReplicates=<int> PrintBootstrap=<boolean>
        SubstitutionModel=<java.lang.String>
        OptimiseEquilibriumFrequencies=<boolean>
        EquilibriumFrequenciesEmpirical=<boolean> FrequencyA=<double>
        FrequencyC=<double> FrequencyG=<double> FrequencyT=<double>
        SubstitutionParameterAC=<int> SubstitutionParameterAG=<int>
        SubstitutionParameterAT=<int> SubstitutionParameterCG=<int>
        SubstitutionParameterCT=<int> SubstitutionParameterGT=<int>
        EmpiricalBaseFrequencyEstimates=<boolean>
        GammaDistributionParameter=<double>
        GammaDistributionParameterFixed=<boolean>
        InvariableSitesProportion=<double>
        InvariableSitesProportionFixed=<boolean>
        NumberOfSubstitutionCategories=<int> OneSubstitutionCategory=<boolean>
        OptimiseStart=<boolean> OptimiseStartKeepingTopology=<boolean>
        TstvRatio=<double> TstvRatioFixed=<boolean> UseBioNJstart=<boolean>
        OptimiseRelativeRateParameters=<boolean>
Input:  Characters
Output: Trees
```

`ProteinMLdist` : This method calculates maximum likelihood protein distance estimates using the following models: *cpREV45* , *Dayhoff* , *JTT* , *mtMAM* , *mtREV24* , *pmb* , *Rhodopsin* and *WAG* [33].

```
Usage:  ProteinMLdist Gamma=<double> Model=<String> PInvar=<double>
        Estimate_variances=<boolean>
Input:  Characters
Output: Distances
```

`PTreeSplits` : This method computes the parsimony splits tree [2].

```
Usage:  PTreeSplits
Input:  Characters
Output: Splits
```

`RefinedBunemanTree` : This method computes the Refined Buneman Tree [5]. This module was implemented by Lasse Westh-Nielsen and Christian N. S. Pedersen.

```
Usage:  RefinedBunemanTree
Input:  Distances
Output: Splits
```

`RootedEqualAngle` : This method computes a rooted split network using the rooted equal angle algorithm [14].

```
Usage:  RootedEqualAngle OptimizeDaylight=<boolean> UseWeights=<boolean>
        RunConvexHull=<boolean> DaylightIterations=<int> OutGroup=<String>
        MaxAngle=<int> SpecialSwitch=<boolean>
Input:  Splits
Output: Network
```

`RYSplits` : This method computes all RY splits.

```
Usage:  RYSplits
Input:  Characters
Output: Splits
```

`SpectralSplits` : This method computes all splits arising using spectral analysis [17].

```
Usage:  SpectralSplits Threshold=<double> Method=<String> Weight_ATvsGC=<double>
        Weight_AGvsCT=<double> Weight_ACvsGT=<double>
Input:  Characters
Output: Splits
```

`SplitDecomposition` : This method computes the *split decomposition* [2].

```
Usage:  SplitDecomposition
Input:  Distances
Output: Splits
```

`SuperNetwork` : This method computes a *super-network* from *partial trees* using the *Z-closure* algorithm [23].

```
Usage:  SuperNetwork EdgeWeights=<String> ZRule=<boolean> NumberOfRuns=<int>
        SuperTree=<boolean> ApplyRefineHeuristic=<boolean>
Input:  Trees
Output: Splits
```

`TreeSelector` : This method is used to select a single tree from a set of trees.

```
Usage:  TreeSelector Which=<int>
Input:  Trees
Output: Splits
```

`UncorrectedP` : This method calculates uncorrected (observed, "P") distances. This is identical to the `Hamming` method.

```
Usage:  Uncorrected_P ignoregaps=<boolean>
Input:  Characters
Output: Distances
```

`UPGMA` : This method computes UPGMA tree [31].

```
Usage:  UPGMA
Input:  Distances
Output: Trees
```

23 Command-Line Options and Mode

SplitsTree4 has the following *command-line options* :

```
-i, --input [string]                Input file.
-g, --commandLineMode               Run SplitsTree in commandline mode. Default value: false.
-c, --commandFile                   File of commands to execute in command-line mode
-x, --initCommand [string]          Execute this command at startup.
-m, --hideMessageWindow             Hide the message window. Default value: false.
-p, --propertiesFile [string]       Properties file. Default: OS specific default location.
-V, --version                       Show version string. Default value: false.
-S, --silentMode                    Silent mode. Default value: false.
-d, --debug                         Debug mode. Default value: false.
-s, --hideSplash                    Hide startup splash screen. Default value: false.
-v, --verbose                       Echo commandline options and be verbose. Default value: fa
-h, --help                          Show program usage and quit.
```

Launching the program with option `-g` will make the program run in non-GUI *command-line mode*, first reading commands from a file supplied with the `-i` option, if any, then executing any command given with the `-x` option, and then finally reading additional commands from standard input.

Please be aware that the command-line version of the program uses the same properties file as the interactive version. So, any a preferences set using the interactive version of the program will also apply to the command-line version of the program. It this is not desired, then please use the `-p` option to supply a different properties file.

Commands provided to the program from within a file must be grouped together in a `SplitsTree` block so:

```
begin SplitsTree;
commands...
end;
```

Here we list all commands known to SplitsTree4 :

```
EXECUTE FILE=file - open and execute a file in Nexus-format
OPEN FILE=file - open (but don't execute) a file in Nexus-format
IMPORT FILE=file [DATATYPE={PROTEIN|RNA|DNA|STANDARD|UNKNOWN}] - open (but don't execute)
  a file in non-Nexus or old-Nexus format
LOAD FILE=file - open or import a file
LOAD TREEFILES=file1 ... filen - load trees from one or more files
LOAD CHARFILES=file1 ... filen - concatenate sequences from one or more files
LOAD FILE=file - open or import a file
SAVE FILE=file [REPLACE={YES|NO}] [APPEND={YES|NO}] [DATA={ALL|list-of-blocks}]
        - save all data or named blocks to a file in Nexus format
EXPORT FILE=file FORMAT=format [REPLACE={YES|NO}] [APPEND={YES|NO}]
            [DATA=list-of-blocks]
         - export data in the named formatAA
EXPORTGRAPHICS [format={EPS|PNG|GIF|JPG|SVG}] file=file [REPLACE={YES|NO}]
               [TEXTASSHAPES={YES|NO}] [TITLE=title]
         - export graphics in specified format (default is EPS)
UPDATE - rerun computations to bring data up-to-date
BOOTSTRAP RUNS=number-of-runs - perform bootstrapping on character data
DELETEEXCLUDED; - delete all sites from characters block that are
            currently excluded
ASSUME assumption - set an assumption using any statement defined
            in the ST_ASSUMPTIONS block
SHOW [DATA=list-of-blocks] - show the named data blocks
CYCLE {KEEP|cycle} - set the graph layout cycle to KEEP or to a given cycle
HELP - show this info
HELP DATA=list-of-blocks - show syntax of named blocks
HELP TRANSFORM=transform - show usage of a specific data transformation
VERSION - report version
```

```
ABOUT - show info on version and authors
QUIT - exit program

To begin or end a multi-line input, enter a backslash '\'
```

Launching the program with option `-g` will make the program run in a command-line mode, first excuting any command given with the `-x` option and then reading commands from the file specified using the `-c` command. If no such file is given, additional commands are read from standard input.

Please note that windows will still open when in command-line mode, but should not be used interactively. To prevent windows from opening, or to use the command-line mode on a server, please use the linux *virtual frame buffer command*, as shown here:

```
xvfb-run --auto-servernum --server-num=1 SplitsTree -g
```

24 Examples

Example files are provided with the program. They are contained in the `examples` sub directory of the installation directory. The precise location of the installation directory depends upon your operating system.

25 Acknowledgments

We would like to thank Barry G. Hall, Pete Lockhart, David Morrison and Mike Steel for many helpful comments.

This product includes software developed by the Apache Software Foundation (http://www.apache.org/), namely the *batik* library for generating image files. It also contains *Jama*, a Java matrix package (http://math.nist.gov/javanumerics/jama/).

References

[1] Sarah C. Ayling and Terence A. Brown. Novel methodology for construction and pruning of quasi-median networks. *BMC Bioinformatics*, 9:115, 2008.

[2] H.-J. Bandelt and A. W. M. Dress. A canonical decomposition theory for metrics on a finite set. *Advances in Mathematics*, 92:47–105, 1992.

[3] H.-J. Bandelt, P. Forster, B. C. Sykes, and M. B. Richards. Mitochondrial portraits of human population using median networks. *Genetics*, 141:743–753, 1995.

[4] Hans-Jürgen Bandelt, Peter Forster, and Arne Röhl. Median-joining networks for inferring intraspecific phylogenies. *Molecular Biology and Evolution*, 16:37–48, 1999.

[5] G.S. Brodal, R. Fagerberg, A. Östlin, C.N.S.Pedersen, and S.S. Rao. Computing refined buneman trees in cubic time. *Lecture Notes in Computer Science*, 2812:259–270, 2003. Springer Verlag.

[6] T. Bruen, H. Philippe, and D. Bryant. A quick and robust statistical test to detect the presence of recombination. *Genetics*, (in press), 2005.

[7] D. Bryant and V. Moulton. NeighborNet: An agglomerative method for the construction of planar phylogenetic networks. In R. Guigó and D. Gusfield, editors, *Algorithms in Bioinformatics, WABI 2002*, volume LNCS 2452, pages 375–391, 2002.

[8] P. Buneman. The recovery of trees from measures of dissimilarity. In F. R. Hodson, D. G. Kendall, and P. Tautu, editors, *Mathematics in the Archaeological and Historical Sciences*, pages 387–395. Edinburgh University Press, 1971.

[9] A. W. M. Dress and D.H. Huson. Constructing splits graphs. *IEEE/ACM Transactions in Computational Biology and Bioinformatics*, 1(3):109–115, 2004.

[10] R.C. Edgar. MUSCLE: multiple sequence alignment with high accuracy and high throughput. *Nucleic Acids Research*, 32(5):1792–97, 2004.

[11] L Excoffier and P Smouse. Using allele frequencies and geographic subdivision to reconstruct gene trees within a species: .... *Genetics*, Jan 1994.

[12] D.P. Faith. Conservation evaluation and phylogenetic diversity. *Biol. Conserv.*, 61:1–10, 1992.

[13] J. Felsenstein. PHYLIP – phylogeny inference package (version 3.2). *Cladistics*, 5:164–166, 1989.

[14] P. Gambette and D.H. Huson. Improved layout of phylogenetic networks. To appear in: TCBB, 2008.

[15] O. Gascuel. BIONJ: An improved version of the NJ algorithm based on a simple model of sequence data. *Mol. Biol. Evol.*, 14:685–695, 1997.

[16] S. Guindon and O. Gascuel. A simple, fast, and accurate algorithm to estimate large phylogenies by maximum likelihood. *Syst Biol.*, 52(5):696–704, 2003.

[17] M.D. Hendy and D. Penny. Spectral analysis of phylogentic data. *Journal of Classification*, 10:5–24, 1993.

[18] B. Holland and V. Moulton. Consensus networks: A method for visualizing incompatibilities in collections of trees. In G. Benson and R. Page, editors, *Proceedings of "Workshop on Algorithms in Bioinformatics"*, volume 2812 of *LNBI*, pages 165–176. Springer, 2003.

[19] B. R. Holland, K. T. Huber, A. Dress, and V. Moulton. delta plots: A tool for analyzing phylogenetic distance data. *Mol Biol Evol*, 19(12):2051–2059, December 2002.

[20] D.H. Huson. SplitsTree: A program for analyzing and visualizing evolutionary data. *Bioinformatics*, 14(10):68–73, 1998.

[21] D.H. Huson. Introduction to phylogenetic networks. Tutorial presented at ISMB, 2005.

[22] D.H. Huson and D. Bryant. Application of phylogenetic networks in evolutionary studies. *Molecular Biology and Evolution*, 23:254–267, 2006. Software available from `www.splitstree.org`.

[23] D.H. Huson, T. Dezulian, T. Kloepper, and M. A. Steel. Phylogenetic super-networks from partial trees. *IEEE/ACM Transactions in Computational Biology and Bioinformatics*, 1(4):151–158, 2004.

[24] D.H. Huson and M. Steel. Phylogenetic trees based on gene content. *Bioinformatics*, 20(13):2044–9, 2004.

[25] D.H. Huson, M.A. Steel, and J. Whitfield. Reducing distortion in phylogenetic networks. In P. Bücher and B.M.E. Moret, editors, *Algorithms in Bioinformatics*, LNBI 4175, pages 150–161, 2006.

[26] P. J. Lockhart, M. A. Steel, and D.H. Huson. Invariable site models and their uses in phylogeny reconstruction. *Syst. Biol.*, 49(2):225–232, 2000.

[27] D.R. Maddison, D.L. Swofford, and W.P. Maddison. NEXUS: an extendible file format for systematic information. *System. Bio.*, 46(4):590–621, 1997.

[28] E. Mossel and M. Steel. A phase transition for a random cluster model on phylogenetic trees. Submitted, 2003.

[29] M. Nei and J.C. Miller. A simple method for estimating average number of nucleotide substitutions within and between populations from restriction data. *Genetics*, 125:873–879, 1990.

[30] N. Saitou and M. Nei. The Neighbor-Joining method: a new method for reconstructing phylogenetic trees. *Molecular Biology and Evolution*, 4:406–425, 1987.

[31] R. R. Sokal and C. D. Michener. A statistical method for evaluating systematic relationships. *University of Kansas Scientific Bulletin*, 28:1409–1438, 1958.

[32] M.A. Steel. Recovering a tree from the leaf colorations it generates under a Markov model. *Appl. Math. Lett.*, 7(2):19–24, 1994.

[33] D. L. Swofford, G. J. Olsen, P. J. Waddell, and D. M. Hillis. Chapter 11: Phylogenetic inference. In D. M. Hillis, C. Moritz, and B. K. Mable, editors, *Molecular Systematics*, pages 407–514. Sinauer Associates, Inc., 2nd edition, 1996.

[34] J.D. Thompson, D.G. Higgins, and T.J. Gibson. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucl. Acids Res.*, 22:4673–4680, 1994.

Index