# Software Manual for SatuTe

Christiane Elgert

August 28, 2024

## Contents

# 1 Introduction

## 1.1 Overview

Welcome to the SatuTe manual. This document provides comprehensive information on how to install, configure, and use SatuTe effectively.

SatuTe (**Satu**ration **Te**st) is a Python-based tool designed to test for phylogenetic information in phylogenetic analyses. The absence of phylogenetic information can be considered saturation. For two sequences, saturation occurs when multiple substitutions obscure true genetic distances, potentially leading to artifacts and errors. SatuTe provides a new measure that generalizes the concept of saturation between two sequences to a theory of saturation between subtrees. The implemented test quantifies whether the given alignment provides enough phylogenetic information shared between two subtrees connected by a branch in a phylogeny.

This enables the detection of branch saturation and assesses the reliability of inferred phylogenetic trees and the data from which they are derived in phylogenetic reconstruction.

General citation for SatuTe:

- C. Manuel, C. Elgert, E. Sakalli, H.A. Schmidt, A. von Haeseler *When the Past Fades: Detecting phylogenetic signal with SatuTe*, in preparation

## 1.2 Requirements

The minimal input of SatuTe is a multiple sequence alignment, a model of sequence evolution with its parameters, and a phylogenetic tree. SatuTe parses these essential pieces of information from the output of IQ-Tree. While we strongly recommend running IQ-Tree separately with customized options, SatuTe can also use an IQ-Tree executable to generate any missing information using default settings.

**Technical Requirements:**

- Python: 3.6 or higher
- IQ-Tree: 2.2.2.3 or higher

## 1.3 Main Workflow

From the input, SatuTe first calculates the spectral decomposition of the substitution matrix and determines the likelihood vectors for each node in the tree. It then performs the test for phylogenetic information on a user-selected branch or on each branch of the tree. The program outputs the test results and its components in different CSV files and Nexus files (see Section 4).

If an evolutionary model with rate heterogeneity is used, each site is assigned to the rate category with highest posterior probability. Then, for each category, SatuTe employs the test for phylogenetic information on the rescaled phylogenetic tree and the subalignment of the considered category.

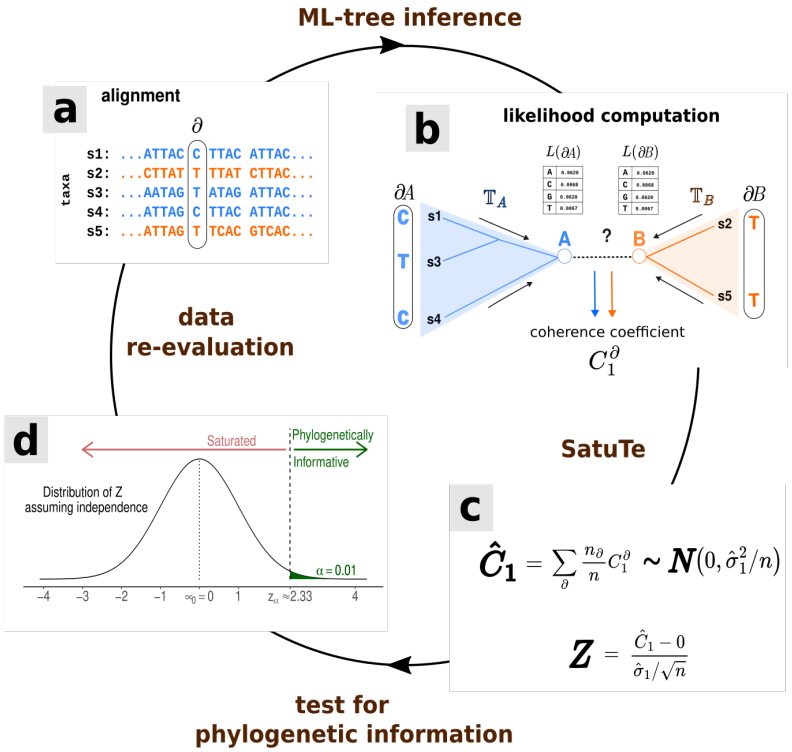


Figure 1: **Theoretical Foundation of SatuTe.** **a**, In an alignment of the sequences from five taxa, each column represents a pattern $\partial$. **b**, The branch $AB$ splits the five-taxon tree into subtrees $\mathbb{T}_A$ with sequences $S1, S3, S4$ and subtree $\mathbb{T}_B$ with sequences $S2, S5$. Additionally, branch $AB$ splits each pattern, such as $\partial = \text{CTTCT}$, into subpatterns $\partial A = \text{CTC}$ and $\partial B = \text{TT}$. These subpatterns are used to compute the likelihood vectors $\boldsymbol{L}(\partial A)$ and $\boldsymbol{L}(\partial B)$, which are then used to compute the coefficient $C_1^{\partial}$. **c**, The average coherence $\hat{C}_1$ is approximately normal distributed. Under the null hypothesis $H_0$ that the alignment was generated independently by subtrees $\mathbb{T}_A$ and $\mathbb{T}_B$, $\mathbb{E}[\hat{C}_1] = 0$, while its variance $\hat{\sigma}_1^2/n$ is easier to estimate. Combining this, we can compute the SatuTe z-score $Z$. **d**, If $Z > z_\alpha$, we reject the null hypothesis $H_0$ at the significance level $\alpha$ and say that the alignment is phylogenetic informative for this branch. Otherwise, the alignment is saturated, meaning that it does not reflect the existence of branch $AB$.

# 2 Installation

SatuTe is available as a Python package from PyPI and can be installed via pip. We recommend using pipx to install SatuTe as a standalone command line tool. Using pipx ensures that SatuTe and its dependencies are installed in an isolated environment, minimizing potential conflicts with other Python packages on your system.

## 2.1 Prerequisites

- Python 3.6 or higher
- pipx (Python package installer)

## 2.2 Install SatuTe using pipx

1. **Install pipx:** If you don't have pipx installed, you can install it using pip:

   ```
   pip install pipx
   ```

   After installation, ensure pipx is set up correctly:

   ```
   pipx ensurepath
   ```

2. **Install SatuTe using pipx:** Once pipx is installed, you can use it to install SatuTe:

   ```
   pipx install satute
   ```

For more detailed instructions and information about pipx, refer to the official pipx documentation.

## 2.3 Verifying the Installation

After the installation is complete, you can verify that SatuTe has been installed correctly by running the following command:

```
satute --version
```

You should see the version number of SatuTe printed on the screen, confirming that the installation was successful.

# 3 Basic Features

## 3.1 Getting Started

To test for branch saturation, you need a multiple sequence alignment, a model of sequence evolution with its parameters, and a phylogenetic tree. SatuTe extracts this essential information from the output of IQ-Tree. The following files are needed:

- a multiple sequence alignment (dna or protein alignment) given in a `.fasta`, `.phy` or `.txt` file.
- the `.iqtree` file to extract the substitution models and parameters and the newick string of the phylogenetic tree.
- the `.siteprob` file to determine the rate category for each site, if a rate heterogeneity model is given.

> **Note**
>
> Note that for rate heterogeneity models you need to run IQ-Tree with the option `-wspr` to get the site probabilities per mixture class and rate category outputted to `.siteprob` file.

If you have a previous output from an IQ-Tree run, such as one from the Webserver, you can specify the directory containing the IQ-Tree files using the `-dir` option.

**SatuTe options**:

| Option | Description |
|---|---|
| `-dir <directory_path>` | Path to the input directory containing IQ-Tree output files. Use this option when you've already run IQ-Tree and want to avoid re-running it. The directory should contain essential IQ-Tree output files including the .iqtree file, tree file(s), and possibly a .siteprob file. |
| `-alpha <significance_level>` | Defines the significance level for the saturation test. The default value is 0.05, indicating a 5% significance level. |

We will present three distinct examples and provide explanations for each to ensure that the user gains a fundamental understanding of the saturation test. The example files used here are on our GitHub repository at `https://github.com/Elli-ellgard/SatuTe/tree/main/examples`.

### 3.1.1 Example: True Tree

All files for this example are provided in the folder `examples/examples_dna/dir_true_tree`.

To evaluate whether the nucleotide alignment `example_dna.fasta` under the JC model provides enough phylogenetic information to justify the branches in the balanced 16-taxon tree on the taxon set A1, ..., A8, B1, ..., B8, provided in `labelled_true_tree.tree` and Figure 2, we run IQ-Tree using the following command:

```
iqtree2 -s example_dna.fasta -te labelled_true_tree.tree -m JC -blfix
```

The `examples/examples_dna/dir_true_tree` folder provides the output files of the IQ-Tree run. Note that this scenario represents the statistically correct situation.

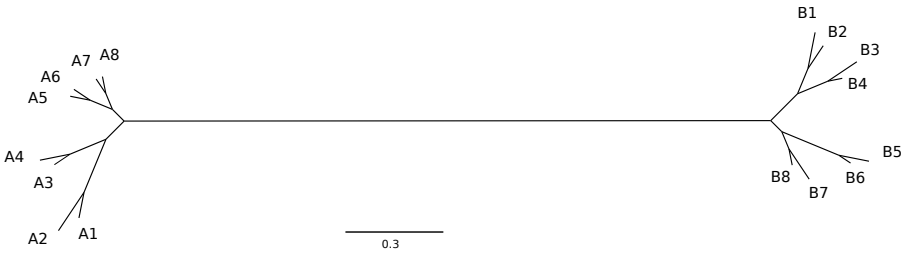


Figure 2: A balanced 16-taxon trees on the taxon set A1, ..., A8, B1, ..., B8 with a long central branch.

**Note**

Although the example focuses on a nucleotide alignment, the core concepts are equally applicable to protein alignments.

For an easy start with SatuTe, we can run the following command:

```
satute -dir ./examples/examples_dna/dir_true_tree
```

For all branches, SatuTe tests whether the alignment provides enough phylogenetic information shared between two subtrees connected by the considered branch in the 16-taxon tree. The output files are directly written into the directory and provide different information. There will be

- a log file `satute.log`,
- a `.satute` file with specific details important for the analyses,
- a test results file `satute.csv` containing the results for all branches,
- a nexus file `satute.nex` containing the results (p_value, decision_test, decision_test bonferroni_corrected) as metadata
- a file `satute.components.csv` that outputs coherence coefficient for each site and branch

For more information about the output files, refer to Section 4.

The `satute.log` file, among other details, presents information about the substitution model and its spectral composition. Since the JC model, has a second largest eigenvalue with a multiplicity of 3, the file lists the three corresponding right eigenvalues utilized in the test calculations.

On the other hand, the file `satute.csv` provides a comprehensive overview of the saturation test results for a specific branch or all branches. The output we receive is as follows:

| branch | ... | **z_score** | p_value | ... | **decision_test** | ... | branch_length | number_of_sites |
|---|---|---|---|---|---|---|---|---|
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| (NodeA5678*, NodeAroot*) | ... | **38.7316** | 0 | ... | **Informative** | ... | 0.05108455 | 1000 |
| (NodeBroot*, NodeAroot*) | ... | **1.6294** | 0.0516 | ... | **Saturated** | ... | 2 | 1000 |
| (NodeA56*, NodeA5678*) | ... | **45.1553** | 0 | ... | **Informative** | ... | 0.07450149 | 1000 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |

Only the central branch (NodeBroot*, NodeAroot*) is saturated.

**Note**

A more application-relevant case assumes that the true-tree topology is known, but the branch lengths are estimated using maximum likelihood. Simulations have shown that this test may be slightly too liberal in such cases, but this is generally considered acceptable if the sequence length is sufficiently long.

A typical question in this context would be to determine which branches in a generally accepted tree are supported by a new alignment for a sequence family that has not yet been investigated. In other words, where in the accepted tree topology is the new alignment phylogenetic informative?

Given a path to an IQ-Tree executable, SatuTe runs IQ-Tree with default options to generate the required data, namely an alignment, a model of sequence evolution with its parameters, and a phylogenetic tree. The following command can be used to generate the same results using other SatuTe options (see Section 3.2):

```
satute -msa ./dir_true_tree/example_dna.fasta -model JC \
       -tree ./dir_true_tree/labelled_true_tree.tree \
       -iqtree iqtree2 -add_iqtree_options "-blfix"
```

In this case, use the SatuTe mode MSA+Model+Tree with additional IQ-Tree arguments.

### 3.1.2 Example: Maximum Likelihood Tree

In phylogenomics, the true tree is often unknown. Therefore, various reconstruction methods, such as maximum likelihood, are used to determine a tree that best represents the data. From the nucleotide alignment (`example_dna.fasta`) provided above, we infer the maximum likelihood (ML) tree using IQ-Tree, with the best-fit model automatically selected by ModelFinder. The following command is used:

```
iqtree2 -s example_dna.fasta
```

All output files of the IQ-Tree run are included in the `examples/examples_dna/dir_ML_tree` folder.

**Note**

Please note that only the **reversible models** for amino acid and nucleotide sequences implemented in IQ-Tree are allowed for the testing branch saturation.

By default, the significance level of the test for phylogenetic information is set to 5%. Analogously to the first example, run SatuTe adjusting the significance level to a more stringent one:

```
satute -dir ./examples/examples_dna/dir_ML_tree -alpha 0.01
```

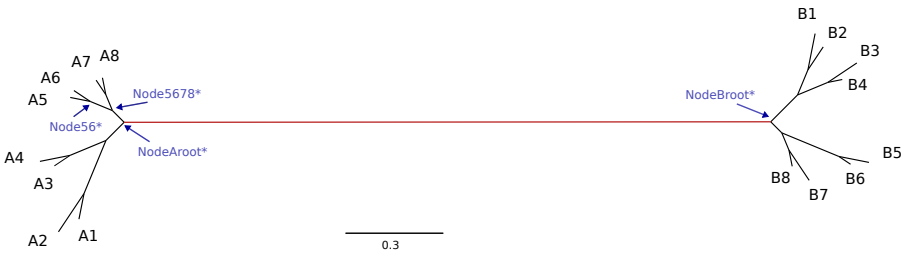


Figure 3: **Saturation in the true tree:** Branches identified as saturated are highlighted in red.

The SatuTe output files are again written directly into the directory. To determine which branches of the ML tree are supported or not supported by the phylogenetic information in the alignment, we refer to the test result file `satute.csv`.

**Note**

Reconstructing a ML tree from an alignment and then using the same alignment to assess where it is phylogenetically informative within the ML tree is considered circular analysis in statistics. Using the same data twice – first to infer the tree and its branch lengths, and then to investigate the phylogenetic signal of the branches – can compromise statistical inference and potentially inflate the type I error rate. Therefore, a Bonferroni correction is necessary.

Now, two other columns in the `satute.csv` output file are important:

| branch | ... | z_score | ... | z_alpha_bonferroni_corrected | decision_bonferroni_corrected_test_tips | ... | branch_length | number_of_sites |
|---|---|---|---|---|---|---|---|---|
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| (Node5*, Node4*) | ... | **38.7977** | ... | **3.5293** | **Informative** | ... | 0.0043852088 | 1000 |
| (Node8*, Node4*) | ... | **2.5172** | ... | **3.6047** | **Saturated** | ... | 2.2904773271 | 1000 |
| (Node6*, Node5*) | ... | **45.2164** | ... | **3.384** | **Informative** | ... | 0.0702957449 | 1000 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |

Once again, only the central branch is saturated, as illustrated in Figure 4.

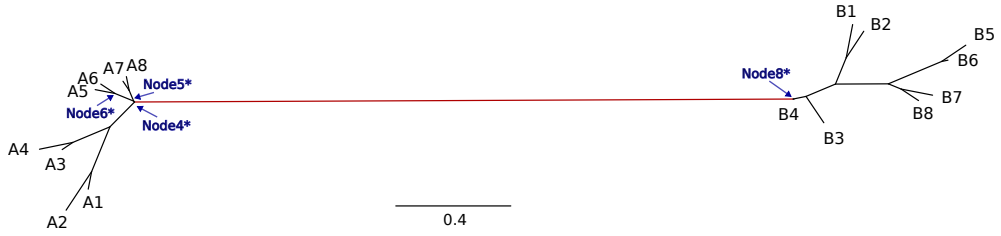


Figure 4: **Saturation in the ML tree:** Branches identified as saturated are highlighted in red.

Given a path to an IQ-Tree executable, SatuTe runs IQ-Tree with default options to generate the required data for the test of saturation, namely an alignment, a model of sequence evolution with its parameters, and a phylogenetic tree. The following commands can be used to generate the same results using other SatuTe options(see Section 3.2):

```
satute -msa ./dir\_ML\_tree/example_dna.fasta \
    -iqtree path_to_iqtree_exe \
    -alpha 0.01
```

Here, use the SatuTe mode MSA.

### 3.1.3 Example: Rate Heterogeneity Model

For this example, we use an amino acid alignment (`example_aa.fasta`) for the 16 taxa A1, ..., A8, B1, ..., B8 and infer the ML tree using IQ-Tree, assuming an LG+G model. The following command is used:

```
iqtree2 -s example_aa.fasta -m LG+G -wspr
```

All output files of the IQ-Tree run are included in the `examples/examples_aa/dir_ML_tree` folder. To run SatuTe on this folder, use the following command:

```
satute -dir ./examples/examples_aa/dir_ML_tree -alpha 0.01
```

**Note**

If an evolutionary model with rate heterogeneity is used, each site is assigned to the rate category with highest posterior probability. Then, for each category $c$, SatuTe employs the test for phylogenetic information as described on the rescaled phylogenetic tree and the subalignment of the considered category. In this process, SatuTe determines for each category $c$ the variance estimator $\hat{\sigma}^2_{1,c}$.

All details about the substitution model used are documented in the `.satute` file. In this example, the chosen model is LG+G4, which incorporates rate heterogeneity with four different categories. Because the branch lengths in the tree are rescaled according to the relative rates of these categories, the `.satute` file provides detailed information on these relative rates, as shown in the following table:

| Category | Relative Rate | Proportion | Empirical Proportion (alignment length 1000) |
|---|---|---|---|
| 1 | 0.1354 | 0.25 | 0.287 |
| 2 | 0.4744 | 0.25 | 0.237 |
| 3 | 0.9987 | 0.25 | 0.242 |
| 4 | 2.3916 | 0.25 | 0.234 |

For each category, the results are provided in corresponding `satute.csv`, `satute.components.csv` and `satute.nex` files. In our example the test results indicate that only the central branch is saturated for category $c_4$. The output we receive in the file `example_aa.fasta_c4_0.01.satute.csv` is as follows:

| branch | ... | **z_score** | ... | **z_alpha_bonferroni_corrected** | **decision_bonferroni_corrected** | ... | branch_length | number_of_sites |
|---|---|---|---|---|---|---|---|---|
| ... | ... | ... | ... | ... | ... | ... | ... | ... |
| (Node5*, Node4*) | ... | **12.9131** | ... | **3.5293** | **Informative** | ... | 0.21509284633596 | 234 |
| (Node8*, Node4*) | ... | **1.4587** | ... | **3.6047** | **Saturated** | ... | 8.6358874072896 | 234 |
| (Node6*, Node5*) | ... | **13.9859** | ... | **3.384** | **Informative** | ... | 0.11624130057072 | 234 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... |

The following command can be used to generate the same results using other SatuTe options (see Section 3.2):

```
satute -msa ./examples/examples_aa/data/example_aa.fasta \
-iqtree path_to_iqtree_exe \
-alpha 0.01
```

Here, use the SatuTe mode MSA.

## 3.2 SatuTe Modes

SatuTe can handle various types of input combinations, which we define as modes. For the test for branch saturation, we need a multiple sequence alignment, a model of sequence evolution with its parameters, and a phylogenetic tree. SatuTe utilizes IQ-Tree output files to obtain all the necessary information. Given a path to an IQ-Tree executable, SatuTe runs IQ-Tree with default options to generate the missing, but required data. We distinguish different modes of SatuTe based on the provided data:

- **MSA**: Identification of the best-fit evolutionary model and tree reconstruction
- **MSA+MODEL**: Estimation of the evolutionary model parameters, if necessary, and tree reconstruction
- **MSA+MODEL+TREE**: Estimation of the evolutionary model parameters, if necessary, and branch length optimization

To provide data use the following options:

| Option | Description |
|---|---|
| `-msa <msa_file_path>` | Path to the Multiple Sequence Alignment (MSA) file you wish to analyze. The MSA can be in FASTA, NEXUS, PHYLIP, or TXT format. |
| `-model <evolution_model>` | Specifies the model of sequence evolution. Common models include GTR, HKY, etc. You can also specify rate heterogeneity and other model extensions, like +G4 for gamma-distributed rates. |
| `-tree <tree_file_path>` | Path to the input tree file in Newick or Nexus format. This tree will be used as the basis for the saturation analysis. |
| `-alpha <significance_level>` | Significance level for the saturation test. The default value is 0.05, indicating a 5% significance level. |
| `-iqtree <iqtree_path>` | Specifies the path to the IQ-Tree executable. If IQ-Tree is installed system-wide, just providing the executable name (iqtree or iqtree2) will suffice. |
| `-add_iqtree_options <additional_option>` | Specify additional options for the IQ-Tree run, if necessary. |

### 3.2.1 Mode MSA

As an alternative, you can provide an alignment (`-msa`) and the path to IQ-Tree (`-iqtree`). Then the best-fit evolutionary model will be identified using ModelFinder (as implemented in IQ-Tree) and a maximum likelihood tree will be inferred. IQ-Tree will run only with necessary options. For specific option choices,

**Examples:**

```
satute -msa ./example_dna/data/example_dna.fasta -iqtree iqtree2
satute -msa ./example_dna/data/example_dna.fasta \
       -iqtree path_to_iqtree_exe
```

### 3.2.2 Mode MSA + Model

In addition to specifying an alignment (`-msa`) and the path to IQ-Tree (`-iqtree`), you can also specify the evolutionary model. **All reversible models** for protein and DNA sequences implemented in IQ-Tree **are allowed** for the test of branch saturation (see IQ-Tree Substitution Models). If necessary,

IQ-Tree will estimate the parameters of the evolutionary model and infer a maximum likelihood tree. IQ-Tree will run only with the necessary options. For specific option choices, please run IQ-Tree separately and use the option `-dir` afterwards (see Section 3.1) or use the SatuTe option `-add_iqtree_options` (see Section 3.2.4).

**Examples:**

```
satute -msa ./example_dna/data/example_dna.fasta -model JC \
         -iqtree iqtree2
satute -msa ./example_dna/data/example_dna.fasta -model GTR+G4 \
         -iqtree iqtree
satute -msa ./example_dna/data/example_dna.fasta -model GTR+G \
         -iqtree iqtree
satute -msa ./example_dna/data/example_dna.fasta \
         -model TIM2{4.39,5.30,12.1}  -iqtree iqtree
```

### 3.2.3 Mode MSA + Model + Tree

In addition to the settings in the previous Section 3.2.2, a phylogenetic tree can be provided as input. In this case, IQ-Tree will optimize the branch lengths. Ensure the input tree is in Newick or Nexus format and that an evolutionary model is specified. IQ-Tree will run only with some default options. For specific option choices, please run IQ-Tree separately and use the option `-dir` afterwards (see Section 3.1) or use the SatuTe option `-add_iqtree_options` (see Section 3.2.4).

**Examples:**

```
satute -msa ./path/to/alignment.fasta \
        -tree ./path/to/treefile.tree \
        -model GTR+G4 \
        -iqtree iqtree2
```

### 3.2.4 Add IQ-Tree options

The `-add_iqtree_options` flag allows you to specify additional options for the IQ-Tree run. This provides flexibility to customize the IQ-Tree execution by including specific and multiple additional options that are not covered by the predefined arguments, see IQ-Tree Documentation.

**Examples:**

1. Fixing branch lengths:

   ```
   satute -msa ./example_dna/data/example_dna.fasta \
        -tree ./example_dna/data/labelled_true_tree.tree \
        -model GTR+G4  \
        -iqtree iqtree2 -add_iqtree_options "-blfix"
   ```

2. Specifying minimal and maximal branch lengths:

   ```
   satute -msa /path/to/alignment.fasta \
           -tree /path/to/treefile.tree \
           -model HKY \
           -iqtree /path/to/iqtree_exe \
           -add_iqtree_options "-blmin 0.00001 -blmax 5"
   ```

# 4 SatuTe Output

SatuTe's output file names comprise various details about the analysis and are structured as follows: `<dataset_name>_<rate_category>_<significance_level>_<edge>.<suffix>`

The table below lists the different suffixes and provides descriptions of their file contents:

| Output file | Description |
|---|---|
| `.satute` | The .satute file provides detailed information on the substitution model, spectral decomposition results, analysis execution, and a summary of site categories. |
| `satute.log` | The .log file provides records of SatuTe's execution, including initialization, configuration, the parameters used when calling IQ-TREE, and any errors that occur. |
| `satute.csv` | This file provides a comprehensive overview of the saturation test results for a specific branch or all branches. |
| `satute.components.csv` | The components file provides the estimated (category) variance and the coherence coefficient for each site and branch in the tree. |
| `satute.nex` | The file contains a block for the taxon labels and a block for the phylogenetic tree, with the most important test results integrated into the NEWICK string as metadata. |
| `satute.asr.csv` | The file contains the posterior distributions of ancestral sequences for the left and right node of each edge in the tree, see Section 5.4. |

## 4.1 .satute File Description

The `.satute` file provides as a detailed record of the steps and processes performed by SatuTe during its execution. The key elements documented in the `.satute` include:

- **Timestamp and Versioning**: Logs the specific date and time of the analysis, along with the version of SatuTe used, ensuring accurate record-keeping and reproducibility.
- **Reference**: Provides guidelines on how the software should be cited.
- **Input Configuration**: Details the input files used, such as alignment files and trees.
- **Substitution Model and Parameters**: Describes the substitution model applied during the analysis, including detailed parameters like the rate matrix, stationary distributions, and any modifications to the default settings.
- **Model for Rate Heterogeneity**: Details the application of the rate heterogeneity model across phylogenetic sites, including specific parameters like relative rates. It also covers the proportion of each category relative to empirical observations from the alignment.
- **Spectral Decomposition**: Records the second-largest eigenvalue and its corresponding right eigenvectors
- **Saturation Test**: Records significance level, tested branches and rate categories.
- **Output Generation**: Details the types and formats of the output generated by the software, including CSV and Nexus files, which contain comprehensive results of the phylogenetic analysis.

## 4.2 Log File Description

The `satute.log` file provides as a detailed record of the steps and processes performed by SatuTe during its execution. The key elements recorded in the `satute.log` file are as follows:

- **Initialization and Configuration**: Provides details on the initial settings and configurations used for the SatuTe run, offering context for the following steps.
- **Analysis Execution**: A step-by-step log of the analysis performed by SatuTe, capturing each stage of the process.
- **Output File Writing**: Records the creation and writing of various output files, ensuring that all generated data is properly documented.

## 4.3 Test Result File Description

By using SatuTe, we can quantify whether the given alignment provides sufficient phylogenetic information shared between two subtrees connected by a branch in a phylogeny, thereby supporting the considered branch. The `satute.csv` file provides a comprehensive overview of the saturation test results for a specific branch or all branches. The file includes the following columns:

| Column Name | Description |
|---|---|
| branch | The branch or edge in the tree being analyzed |
| mean_coherence | The mean value of the coherence coefficients calculated for the branch |
| standard_error_of_mean | The standard error of the mean for the coherence coefficient |
| z_score | The z-score used to evaluate if the branch is saturated or informative |
| p_value | The p-value indicating the significance of the test statistic |
| z_alpha | The z-value corresponding to the alpha level for the test |
| decision_test | The decision based on the z-score (e.g., Informative or Saturated) |
| z_alpha_bonferroni_corrected | The corrected z-value considering multiple testing corrections |
| decision_bonferroni_corrected | The decision based on the Bonferroni corrected test using the number of tips of each subtree |
| branch_length | The length of the considered branch or edge in the tree |
| number_of_sites | The number of sites in the alignment |
| rate_category | The rate category for which the analysis was performed |

## 4.4 File for Components of Test Statistic

The `satute.components.csv`file provides the estimated (category) variance and the coherence coefficient for each site and branch in the tree, enabling other analysis of the saturation status like sliding-window analysis.

| Column Name | Description |
|---|---|
| branch | The branch or edge in the tree being analyzed |
| site | The specific site in the alignment being analyzed |
| coherence | The coherence value for the site and branch in the specified rate category |
| category_variance | The estimated variance of the coherence using all considered sites |
| rate_category | The rate category for which the analysis was performed |

This file allows researchers to perform a detailed analysis of the phylogenetic information for different regions of the alignment, facilitating the detection of saturation patterns across the alignment, see Analysis with SatuTe.

## 4.5 Nexus File

A `satute.nex` file consists of two blocks, each enclosed by BEGIN and END statements. These blocks contain the taxon labels and the phylogenetic tree, with the most important test results integrated into the NEWICK string as metadata. A NEXUS file for a saturation analysis of a specific branch looks like this:

```
#NEXUS
BEGIN TAXA;
    DIMENSIONS NTAX=7;
     TAXLABELS
        t7
        t3
        t2
        t5
        t6
        t1
        t4
    ;
END;

BEGIN TREES;
Tree tree1 = (t7:3.01328e-06,(((t3:1.55499,(t2:1.77629,t5:2.76104e-06)
Node5*:0.377782[&z_score=1.4587,p_value=0.0723,decision_test=
Saturated,decision_bonferroni_corrected=Saturated])Node4*:0.368276,
t6:2.16996e-06)Node3*:1.23617,t1:2.22639e-06)Node2*:1.05052,
t4:1.85109)Node1*:0;
END;
```

# 5 Additional Features

## 5.1 Edge Specification

If the tree is pre-labeled, the user can apply the test to a specific branch using the `-edge` option, for example, `-edge "(NodeManhattan*, NodeBoston*)"`. If the tree is not pre-labeled, SatuTe will automatically label it.

**SatuTe options:**

| Option | Description |
|---|---|
| `-edge <edge_name>` | Specify a branch or edge name to focus the analysis on. Useful when you want to check saturation on a specific branch. |

**Example:**

```
satute -msa ./example_dna/data/example_dna.fasta  \
        -model GTR+G4 \
        -edge "(Node1*, Node2*)"
```

## 5.2 Category Features

If an evolutionary model with rate heterogeneity is specified, the user can run SatuTe for the category of interest using the `-category` option.

**SatuTe options:**

| Option | Description |
|---|---|
| `-category <rate_category>` | Rate categories of interest. Relevant for models with gamma-distributed rate variations or FreeRate model. If the `-model` option includes rate variation (e.g., +G4), the `-category` should be a number between 1 and 4. |
| `-category_assignment` | Write assignment of the individual sites to the rate heterogeneity categories. |

**Example:**

```
satute -msa ./example_dna/data/example_dna.fasta  \
        -model GTR+G4 \
        -category 2
```

## 5.3 Bootstrap Analysis

SatuTe can also be seen as a complementary tool to bootstrap methods, which measure how strongly an alignment supports a split between two groups. In the classic bootstrap interpretation, a high bootstrap value is interpreted as a measure of credibility for a branch in the tree. SatuTe, however, helps to determine whether the two groups are genuinely related – indicating a shared evolutionary history – or if the high bootstrap value merely reflects dissimilarity between the groups.

**SatuTe options:**

| Option | Description |
|---|---|
| `-ufboot <number_of_replicates>` | Number of replicates for the ultrafast bootstrap analysis. Typically, a higher number like 1000 or 5000 is used. Ultrafast bootstrap provides rapid approximations to traditional bootstrap values. |
| `-boot <number_of_replicates>` | Number of replicates for traditional bootstrap analysis. This also computes a Maximum Likelihood (ML) tree and a consensus tree. Common value is 100. |

**Examples:**

```
satute -msa ./example_dna/data/example_dna.fasta \
        -model GTR+G4 \
        -ufboot 1000
satute -msa ./example_dna/data/example_dna.fasta \
        -model GTR+G4 \
        -boot 100
```

## 5.4 Ancestral sequence reconstruction

**SatuTe options:**

| Option | Description |
|---|---|
| `-asr` | Write ancestral sequences for all tree nodes, using the empirical Bayesian method, for the right and left sides of a branch in the .asr.csv file. |

**Example:**

```
satute -msa /path/to/alignment.phy -model GTR+G4 \
        -iqtree /path/to/iqtree2 -asr
```

The `.asr.csv` file contains the posterior distributions of ancestral sequences for both the left and right nodes of the split trees at each edge. The columns in the file are:

| Column Name | Description |
|---|---|
| pA_left | The probability of nucleotide A in the left subtree at the specified site. |
| pC_left | The probability of nucleotide C in the left subtree at the specified site. |
| pG_left | The probability of nucleotide G in the left subtree at the specified site. |
| pT_left | The probability of nucleotide T in the left subtree at the specified site. |
| node_left | Name of the node in the left subtree. |
| pA_right | The probability of nucleotide A in the right subtree at the specified site. |
| pC_right | The probability of nucleotide C in the right subtree at the specified site. |
| pG_right | The probability of nucleotide G in the right subtree at the specified site. |
| pT_right | The probability of nucleotide T in the right subtree at the specified site. |
| node_right | Name of the node in the right subtree. |
| site | The site index in the alignment. |
| branch | The branch in the tree that splits the left and right subtrees. |

## 5.5 Others

**SatuTe options:**

| Option | Description |
|---|---|
| `-output_suffix <output_suffix>` | Specify a suffix for the output file. |
| `-verbose` | Enable verbose logging. By default, SatuTe only prints warnings and errors to the standard output. |
| `-quiet` | Even no warnings on the the terminal. |
| `-dev` | Show the stack trace. |

**Example:**

```
satute -dir ./examples/examples_dna/dir_true_tree -verbose
```

# 6 Troubleshooting

## 6.1 Invalid Command Combinations

Certain combinations of command-line arguments are invalid:

- Directory with Model, MSA, Tree, Ufboot, Boot: Providing an input directory with `-dir` shouldn't be combined with specifying a msa, a model, a tree, ufboot or boot option.
- Model and Tree without MSA: Just providing the `-model` and `-tree` without a msa (`-msa`) is insufficient.
- MSA+Model+Tree with ufboot or boot option: In the msa+model+tree mode, the inference is not re-done again, such that no ufboot and boot values can be determined.
- Edge without MSA or DIR: The `-edge` option, used to focus the analysis on a specific branch, requires the `-msa` option or `-dir` option.

## 6.2 Potential Errors and Warnings

### 6.2.1 General Input Validation

| **Error Message** | **Explanation** |
|---|---|
| Error: An MSA file must be specified when `-dir` is not used. | This error occurs when neither the `-msa` nor the `-dir` option is provided. One of these options must be specified to indicate the input source. |
| Error: The `msa` and `dir` options cannot be used together. | This error occurs when both the `-msa` and `-dir` options are provided simultaneously. Only one should be used to avoid conflicting input sources. |
| Error: The `-dir` option cannot be used with `-msa`, `-tree`, `-model`, `-ufboot`, `-boot`, or `-add_iqtree_options`. Choose either `-dir` or the other options. | This error occurs when the `-dir` option is used in combination with other options like `-msa`, `-tree`, `-model`, `-ufboot`, `-boot`, or `-add_iqtree_options`. These options are mutually exclusive. |

### 6.2.2 Tree and Model Validation

| **Error Message** | **Explanation** |
|---|---|
| Error: A model must be specified when using a tree file. | This error occurs when a tree file (`-tree`) is provided without specifying a model (`-model`). A model is necessary to interpret the tree file. |
| Error: The `-ufboot` or `-boot` options cannot be used with `-tree`. | This error occurs when bootstrapping options (`-ufboot` or `-boot`) are used in conjunction with a tree file (`-tree`). These options should not be combined. Bootstrapping requires tree inference to be done from scratch, which is incompatible with using a pre-existing tree file. |

### 6.2.3 Category Validation

| Error Message | Explanation |
|---|---|
| Invalid category: {category}. The category must be between 1 and {number_rates}, inclusive. Please choose a valid category index. | This error occurs when the specified category (`-category`) is outside the valid range of 1 to `number_rates`. The category must be within this range to be valid. The number of rates is determined by the user or inferred by ModelFinder. |
| Error: Chosen category '{input_category}' is out of the valid range of {rate}. | This error occurs when the chosen category is not within the valid range of 1 to `number_rates`. The category must fall within this range to be acceptable. |
| Chosen category rate '{chosen_category}' is empty. Choose a different category with assigned sites. | This error occurs when the chosen category is empty, meaning no sites are assigned to it. A different category with assigned sites should be chosen. |

### 6.2.4 Directory and File Validation

| Error Message | Explanation |
|---|---|
| argparse.ArgumentTypeError: {path} is not a valid directory | This error occurs when the provided path is not a valid directory. The path must point to an existing directory. |
| argparse.ArgumentTypeError: {path} directory is empty | This error occurs when the specified directory is empty. The directory should contain relevant files for the analysis. |
| argparse.ArgumentTypeError: No .iqtree file found in the directory {path} | This error occurs when no `.iqtree` file is found in the specified directory. An `.iqtree` file is required for further processing. |
| argparse.ArgumentTypeError: No file with suffixes {suffixes_str} found in the directory {path} | This error occurs when no file with the specified suffixes (e.g., `.fasta`, `.nex`, `.phy`, `.txt`) is found in the directory. Such files are necessary for processing. |
| argparse.ArgumentTypeError: {path} is not a valid file | This error occurs when the provided path is not a valid file. The path must point to an existing file. |

### 6.2.5 Alpha Value Validation

| Error Message | Explanation |
|---|---|
| Invalid alpha value '{alpha}'. The alpha value must be between 0 and 1, exclusive. Please provide a valid alpha value for the significance level. | This error occurs when the alpha value is not between 0 and 1, exclusive. Alpha is the significance level and must be in this range for validity. |

#### 6.2.6 IQ-TREE Execution Errors

| Error Message | Explanation |
|---|---|
| IQ-TREE execution failed with error code {e.returncode}. Command: {iq_tree_command}. Output: {e.stdout}. Error: {e.stderr}. Please check the command and ensure that IQ-TREE is installed and accessible. | This error occurs when IQ-TREE fails to execute successfully. The return code, command, output, and error message from IQ-TREE are logged. Possible reasons include incorrect command syntax, missing files, or compatibility issues with IQ-TREE. |
| Error: Taxa set of the tree does not align with the taxa set of the MSA alignment. | This error occurs when the taxa in the phylogenetic tree do not match the taxa in the multiple sequence alignment (MSA). Ensure that all taxa present in the MSA are also present in the tree, and vice versa, to maintain consistency in analysis. |

### 6.3 Support and Contact Information


| Person | Email |
|---|---|
| Enes Berk Sakalli | enes.sakalli@univie.ac.at |
| Christiane Elgert | christiane.elgert@gmx.de |


## References


[1] C. Manuel, C. Elgert, E. Sakalli, H.A. Schmidt, A. von Haeseler *When the past fades: Detecting phylogenetic signal with SatuTe*, in preparation