Siren User Guide

Release 2.1.1

Esther Galbrun and Pauli Miettinen

July, 2014

CONTENTS

1 Introduction **1**

2 Getting Started **3**
2.1 Importing data . . . 3
2.2 Mining redescriptions from scratch . . . 3
2.3 Expanding a redescription . . . 3
2.4 Filtering redescriptions . . . 4
2.5 Exporting redescriptions . . . 4
2.6 Saving as a package . . . 4

3 Interface **5**
3.1 Siren Windows and Tabs . . . 5
3.2 Menu . . . 7
3.3 Views . . . 8
3.4 Process . . . 9
3.5 Windows . . . 9
3.6 Help . . . 10

4 Formats **11**
4.1 Data formats . . . 11
4.2 Redescriptions formats . . . 12

5 Preferences **15**
5.1 Mining parameters . . . 15
5.2 Interface parameters . . . 15
5.3 Preferences file . . . 17

CHAPTER

ONE

INTRODUCTION

Redescription mining is a powerful data analysis tool that aims at finding alternative descriptions of the same entities.

For example, in biology, an important task is to identify the bioclimatic constraints that allow some species to survive, that is, to describe geographical regions in terms of both their bioclimatic conditions and the fauna that inhabit them.

Siren *is a tool for interactive mining and visualization of redescriptions. It is based on the ReReMi mining algorithm.*

CHAPTER

TWO

GETTING STARTED

Siren *allows you to interactively mine and visualize redescriptions from your data.*

We outline here some high-level interactions offered by Siren.

2.1 Importing data

After you get *Siren* installed and running, it will open the *tools window*, with tabs for variables and redescriptions, all of them empty. Hence, the first thing to do is to import data or open an existing siren package to start working.

If you already have some siren package (i.e. with a `.siren` extension) you can open it via the interface menu *File* → *Open*.

Otherwise, or if you want to work on a new data set, you can *import data* to *Siren*, this will populate the `Entities` and `Variables` tabs.

2.2 Mining redescriptions from scratch

Once the two sets of variables are loaded and can be seen in the tabs you may want to let the tool mine redescriptions in an fully automated way, using currently enabled variables. The menu entry *Process* → *Mine* redescriptions let you do just that.

Alternatively, if both queries in a visualization are empty *Siren* simply mines redescriptions on your data when clicking on the `Expand` button.

Results will be appended to the redescriptions list in the `Expansions` tab.

Before running a mining task make sure you have adjusted the mining preferences...

2.3 Expanding a redescription

You can also press the expansion button to automatically find expansions of any redescription your are currently visualizing and editing. *Siren* will try to append literals to the current redescription, using the enabled variables.

Again, results will be appended to the redescriptions list in the `Expansions` tab.

Before running an expansion task make sure you have adjusted the mining preferences...

2.4 Filtering redescriptions

A list of redescription can be filtered automatically. That is, the algorithm will go through the redescriptions, from top to bottom in the current order, an check for each redescription, whether it is redundant given the previous ones. If a redescription is found redundant it will be disabled. This is done via the interface menu *Edit* → *Filter redundant*.

It is also possible to select a redescription and filter following redescriptions which are redundant to that particular redescription only, via the interface menu *Edit* → *Filter redundant* to current.

2.5 Exporting redescriptions

The redescriptions from the *Redescriptions* tab can be exported under *different formats*.

2.6 Saving as a package

You can save your current project, i.e. the data, current redescription list and preferences as a siren package (i.e. with a `.siren` extension) via the interface menu *Edit* → *Save as...*

If you continue working on the current project you can save changes to the current siren package via the interface menu *Edit* → *Save*.

Existing siren packages can be opened via the interface menu *File* → *Open*.

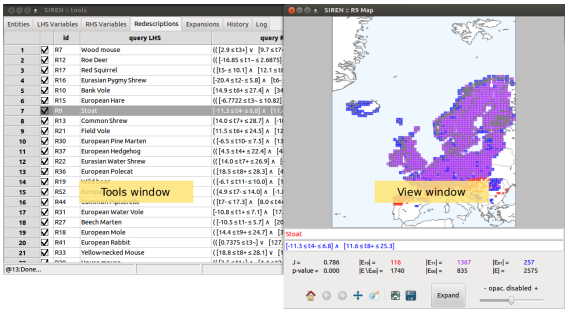
CHAPTER

THREE

INTERFACE

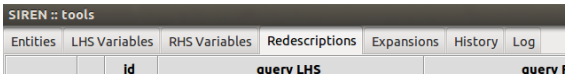
3.1 Siren Windows and Tabs

Siren *has two types of windows:* tools *and* view*, which are presented below.*

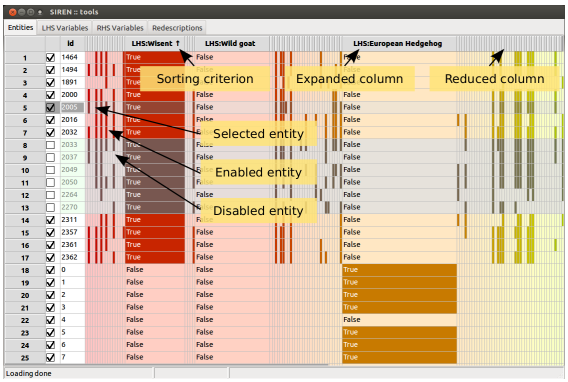


3.1.1 Tools window

The `Tools` window is unique, it is the main siren window. It contains several tabs.

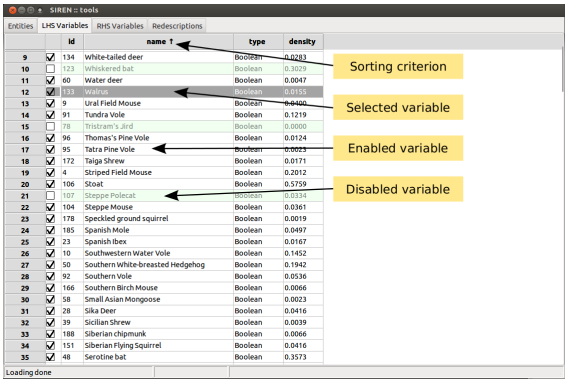


- The **Entities** tab contains the list of entities.



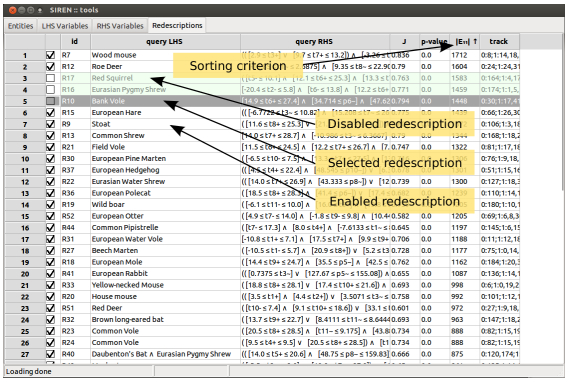
Two tabs contain lists of variables.

- The **LHS Variables** tab,
- and the **RHS Variables** tab contain the list of left-hand side variables and right-hand side variables respectively.



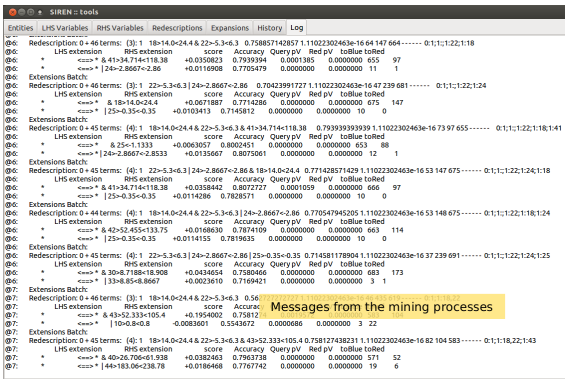
Three tabs contain lists of redescriptions.

- The **Redescriptions** tab is the main list of redescriptions. Redescriptions are imported and exported to and from that list.
- The **Expansions** tab lists redescriptions that have been generated by mining or extending queries.
- The **History** tab lists all edits made to queries, allowing to undo changes.



Finally,

- the **Log** tab contains logging output generated by the mining algorithm.



Tabs can be shown or hidden via the interface menu *Windows* → *Tabs*.

The main means to manipulate variables or redescriptions, depending on which tab you are currently viewing, are available via the `Edit` menu and the contextual menu opened by right click.

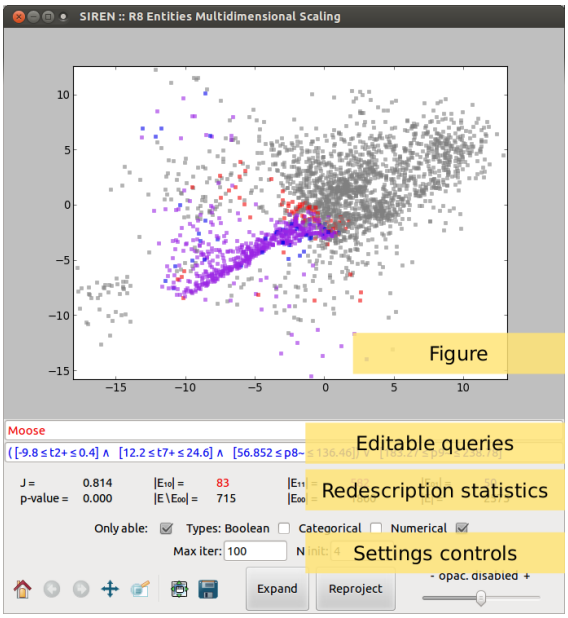
List items can be enabled/disabled by checking/unchecking the corresponding box in the left column. All items in a list can be enabled/disabled simultaneously. Lists can be sorted based on the value of the different fields displayed by clicking on the column header.

Redescriptions can be cut, copied, and pasted from and to different positions in the lists and from one list to the other and vice-versa. All disabled redescriptions can be deleted at once.

3.1.2 View window

A **view** window allows to visualize a redescription, to edit it and launch mining and expansion. Several views of different types of the same redescription can be opened simultaneously. They are linked together an to the original redescription in the list so that edits and selections made in one view are reflected in the list and other views.

A `View` window can be opened with a double click on a variable or redescription in a list from the `Tools` window.



Both queries can be edited using the text boxes. If the *syntax of a query* is incorrect, *Siren* will not be able to parse it and it will fall back on the previous correct query. Queries are parsed when `ENTER` is pressed, in order to avoid parsing error due to partial edits.

Redescriptions statistics are shown below the queries.

In case of projections, setting controls allow to set parameters and the data can be projected anew by clicking the `Reproject` button.

Sliders allow to set the opacity of disabled entities, as well as the level of details for parallel coordinates plots.

Expansion can be started by pressing the `Expand` button. The expansion will be delegated to a background process. It can be interrupted via the menu *Process* → *Stop expander XXX*, where running task are listed. Redescriptions generated during the expansion will be appended to the list of redescriptions in the `Expansions` tab.

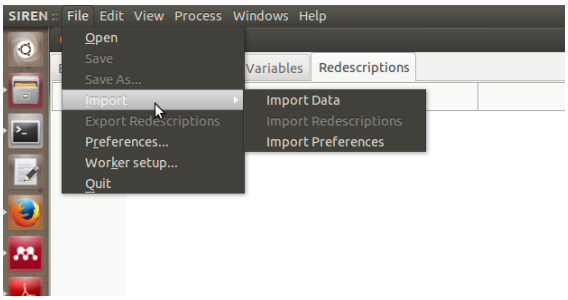
3.2 Menu

The main menu of Siren *is at the top of the* Tools window.

Here is a summary of functionalities available throught the menu.

3.2.1 File

The `File` submenu provides import, export, opening and saving functionalities and setting the preferences.

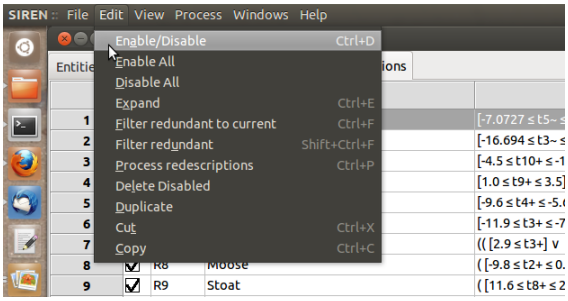


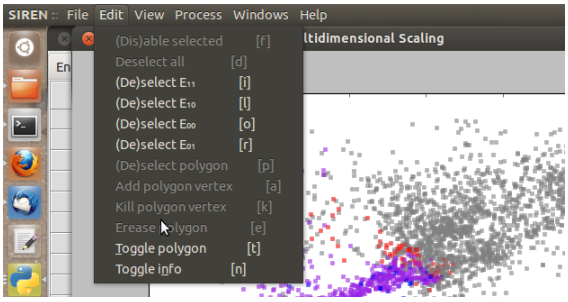
3.2.2 Edit

The content of the `Edit` submenu depends on the tab or view currently active.

If the tab contains redescriptions, it will also allow to filter the redescription and to copy, cut and paste them.

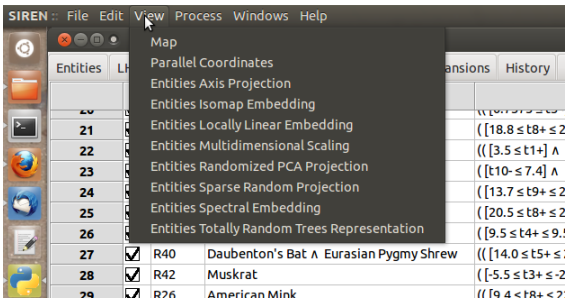
These functionalities can also be accessed via the contextual menu on right-click.





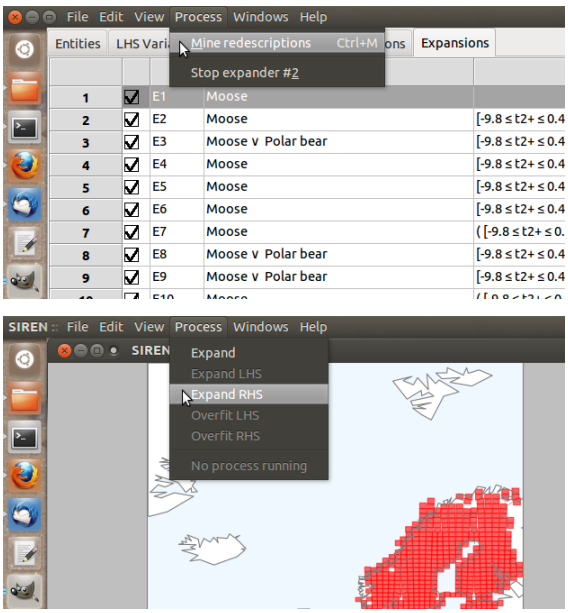
3.3 Views

If the tab contains redescriptions or variables the `View` menu will allow to open a *view window* to visualize the selected item.



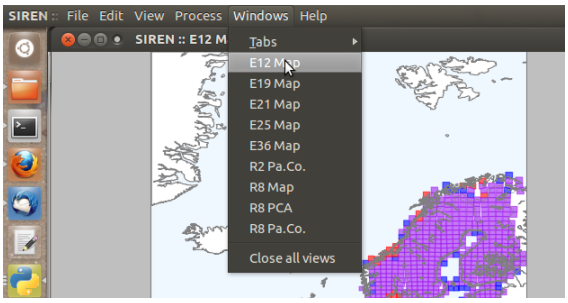
3.4 Process

The `Process` menu allows to start mining redescriptions and contains a list of running task and allows to interrupt any of them.



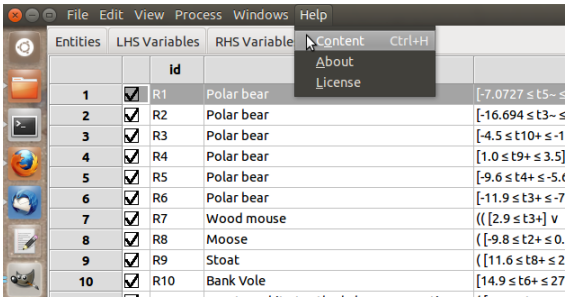
3.5 Windows

The `Windows` menu contains a list of tabs and allows to show or hide any of them. It also contains a list of Views currently opened sorted by redescription, allowing to access any of them and close all at once.



3.6 Help

The `Help` menu provides access to this help and to more information about *Siren* and licensing.



CHAPTER

FOUR

FORMATS

4.1 Data formats

For redescription mining, one considers entities discribed by variables divided into two sets, hereafter arbitrarily called left-hand side and right-hand side. This can be seen as a pair of data matrices, where entities are identified with rows and variables with columns. Both sets of variables describe the same entities, hence, the matrices have the same number of rows.

In *Siren*, data include:

- **Variables**: The variables describing the entities are divided in two sets. They can be of three types:
  1. Boolean,
  2. categorical,
  3. or real-valued.

Obviously, this is required.

- **Entities names**: Optional additional information, providing names for the entities.
- **Variable names**: Optional additional information, providing names for the variables.
- **Coordinates**: Optional location information, i.e. geographic coordinates of the entities. This makes the data geospatial.

Data can be imported to *Siren* via the interface menu *File* → *Import* → *Import Data*. Below, we present the data formats supported by *Siren*.

Data can be imported into *Siren* as CSV files. The program expects a pair of files, one for either side in character-separated values, as can be imported and exported to and from spreadsheet programms, for instance.

There are two main formats,

- **Full**: standard table format, or
- **Sparse**: compact format for dataset with few non-zeros entries.

The two data files need not be in the same format.

If entities names and/or coordinates are provided, they will be used to match entities across the two sides. Otherwise, rows will be match in order and an error will occur if the two side do not contain the same number of rows.

4.1.1 Full format

The data is stored as a table with one column for each variable and one row each entity. The first row can contain the names of the variables. The entities names can be included as columns named *ids*. Similarly the coordinates can be included as a pair of columns named *longitudes* and *latitudes*, respectively.

4.1.2 Sparse format

This format allows to store data that contains few non-zero entries more compactly, as in the Matlab sparse format (or like the edge list of a bipartite graph).

Each line contains an entry of the data as a triple (entity, variable, value). This way, the data is stored as in three columns and as many rows as there are entries. In this case the first line of the data file must contains *ids*, *cids* and *values*, indicating the three columns containing the enities, variables and corresponding value, respectively. Coordinates can be provided in a similar way under the variable names *longitudes* and *latitudes*.

Variable names can be provided inline, that is, simply by using the name of the variable for each entry involving it. Alternatively, variable names can be specified separately with a special "-1" entity. Similarly, entity names can be provided inline or separatly with a special "-1" variable. For example, the following four lines

```
Espoo; population; 260981
Helsinki; population; 614074
Tampere; population; 220609
Turku; population; 182281
```

are equivalent to the following:

```
20; -1; Espoo
7; -1; Tampere
2; -1; Turku
13; -1; Helsinki
-1; 3; population
2; 3; 182281
7; 3; 220609
13; 3; 614074
20; 3; 260981
```

Finally, in case of fully Boolean data without coordinates, the value can be left out. Each pair of (entity, variable) appearing is considered as True, the rest as False.

4.2 Redescriptions formats

The product of redescription mining is a list of redescriptions. A redescription consist of a pair of queries over the variables describing the entities, one query for each set. The two sets of variables are arbitrarily called left-hand side and right-hand side, and so are the corresponding queries.

4.2.1 Supports

The support of a query is the set of entities for which the query holds. Any given redescription partitions the entities into four sets:

- E_{10} is the set of rows for which only the left hand side query holds,
- E_{01} is the set of rows for which only the right hand side query holds,

- E_{11} is the set of rows for which both queries hold,
- and E_{00} is the set of rows for which neither of the queries hold.

Redescriptions can be imported to *Siren* via the interface menu *File* → *Import* → *Import Redescriptions*. More importantly, they can be exported via the interface menu *File* → *Export Redescriptions*. Below, we present the redescription formats supported by *Siren*.

4.2.2 Queries

A query is formed by combining literal using Boolean operators.

While *ReReMi* only generate linearly parsable query (see references for more details), *Siren* can actually evaluates arbitrary queries, as long as they are well formed following the informal grammar below. In particular, parenthesis should be used to separated conjunctive blocks and disjunctive block, alternating between operators. For example, while the later cannot be generated by *ReReMi*, $(a \wedge b) \vee \neg c$ and $(a \wedge b) \vee (c \wedge d)$ are both supported. $(a \wedge b) \wedge (c \wedge d)$ is not, because of incorrect alternance of operators between parenthesis blocks. It should simply be written as $a \wedge b \wedge c \wedge d$.

We consider three types of literals, defined over a Boolean, categorical or numerical variable respectively.

Below is an unformal grammar of *Siren*'s query language.

query = disjunction | conjunction | literal ;
conjunction = conj_item { ("&" | " ∧ ") conj_item }+ ;
disjunction = disj_item { ("|" | " ∨ ") disj_item }+ ;
conj_item = literal | ("(" disjunction ")") ;
disj_item = literal | ("(" conjunction ")") ;
literal = categorical_literal | realvalued_literal | boolean_literal ;
categorical_literal = ("[")? variable_name cat (" = " | " ≠ " | " ∈ " | " ∈ ") category ("]")? ;
realvalued_literal = [neg] ("[")? [variable_value lth] variable_name lth variable_value ("]")? ;
realvalued_literal = [neg] ("[")? variable_value lth variable_name ("]")? ;
boolean_literal = [neg] ("[")? variable_name ("]")? ;
variable_name = STRING | ?/vd+/? ;
category = STRING | ?/d+/? ;
variable_value = ?/[+-]?d+([.])?d([Ee][-+]d+)?/? ;*
lth = "<" | " ≤ " ;
neg = "!" | "¬" ;

Naturally, the type of literal and the type of variable should match, i.e., $[4.0 \leq Va \leq 8.32]$ is a valid numerical literal only if the corresponding variable Va is a numerical variable. Furthermore, the upper bound of a numerical variable should always be greater or equal to the lower bound and either of them should be specified.

4.2.3 Redescription statistics

The statistics of a redescription include:

- accuracy, as measured by Jaccard coefficient $|E_{11}|/(|E_{10}| + |E_{11}| + |E_{01}|)$,
- p-value,
- cardinality of the *support sets* E_{10}, E_{01}, E_{11}, E_{00} (sometimes also referred to as alpha, beta, gamma and delta, respectively).

4.2.4 Exporting Redescriptions

Redescriptions from the `Redescriptions` tab can be exported to a file, one redescription per line, with both queries and basic statistics tab separated. Three of formatting options are available, determined by the provided filename:

- **named**: Uses the names of the variables instead of variable ids in the queries. Activated if the filename matches the pattern `*[a-zA-Z]named[a-zA-Z]*`.
- **all** By default disabled redescriptions will not be printed when exporting redescriptions. If the filename matches the pattern `*[a-zA-Z]all[a-zA-Z]*`, disabled redescriptions will also be printed.
- **tex** Rather than tab separated format, if the filename as `.tex` extension, a tex file is produced that can be compiled to obtain a table of the redescriptions. (Cannot be imported back)

Inside a siren package, the redescriptions are stored in tab separated format together with disabled status.

4.2.5 Importing Redescriptions

Tab separated formats can be imported into *Siren*, *TeX* cannot.

CHAPTER

FIVE

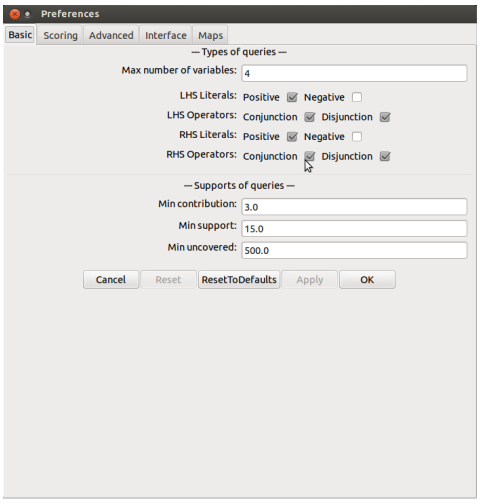
PREFERENCES

Preference parameters can be set via the interface menu Edit → Preferences.

An XML preferences file stores all non-default parameters.

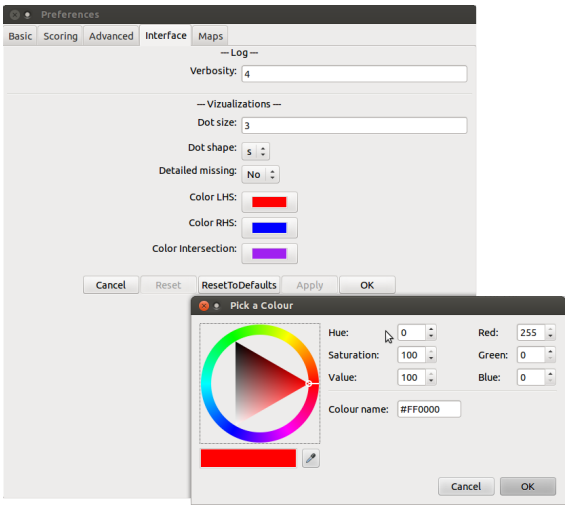
5.1 Mining parameters

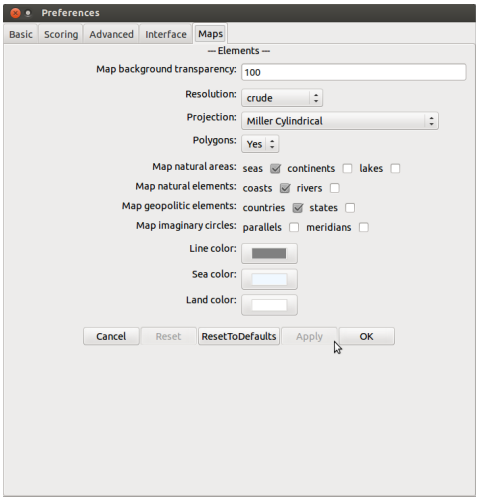
Mining parameters are specified here and can be set through the interface.



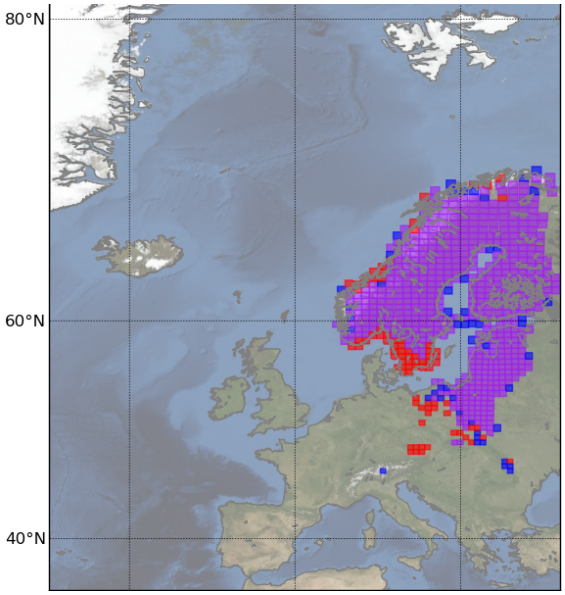
5.2 Interface parameters

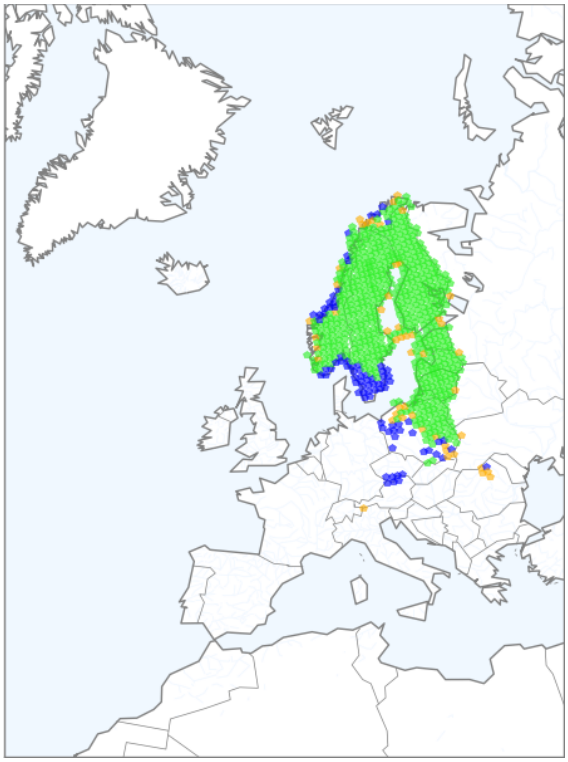
Interface parameters are specified here and can be set through the interface.





For instance, this allows to modify the appearance of the maps.





5.3 Preferences file

The ouline of a preferences file for *Siren* and *ReReMi* is as follows:

```
<root>
  <!--- start parameter -->
  <parameter>
    <!--- parameter name -->
    <name>max_red</name>
    <!--- parameter value -->
    <value>100</value>
  </parameter>
  <!--- end parameter -->
  <!--- start another parameter -->
  <parameter>
    <name>neg_query</name>
    <!--- multiple choices parameters might have several values -->
    <value>Positive</value>
    <value>Negative</value>
  </parameter>
  <!--- end another parameter -->
  <!--- etc. --->
</root>
```