Reference manual for Seqtometry Graphical User Interface (GUI)

Section 1 (setup)

Complete the following steps to ensure you have a proper environment to execute code necessary for the Seqtometry GUI.

Before installation of the SeqtometryGUI package, the following is needed:

1. A working R (v4.3.0 or above) environment (for the GUI and associated functions).
    a. Refer to R installation instructions specific to your operating system (alternatively, use conda).
2. A working Python 3 (v3.10.11 or above) environment (for MAGIC).
    a. Refer to Python installation instructions specific to your operating system (alternatively, use conda).
3. A successful install of MAGIC v3.0.0 (in the Python environment above).
    a. See installation instructions: https://github.com/KrishnaswamyLab/MAGIC.
4. A successful install of reticulate (in R), as well as its configuration to use the Python environment above.
    a. See configuration instructions on: https://rstudio.github.io/reticulate/articles/versions.html

Then, install the SeqtometryGUI package (in R) using devtools (which should automatically install the required Seqtometry package as well as other dependencies).

```
devtools::install_github("HawigerLab/SeqtometryGUI")
```

Section 2 (usage)

Complete the following steps to run the Seqtometry GUI on a data set and signature collection of your choosing.

Within an R environment:

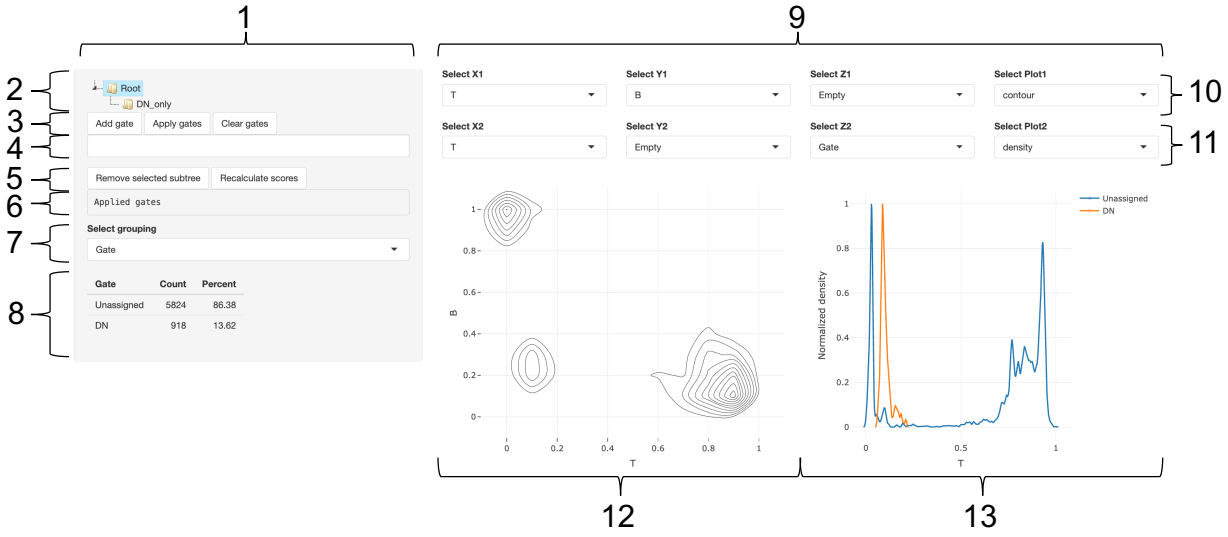
1. Load the following:
    a. A Seurat object containing the single cell data to be analyzed.
        i. The object should have been subjected to quality control (the exact manner of QC is left to the user).
        ii. At the minimum, the object must contain counts per gene (transcripts or gene activities, for scRNA-seq and scATAC-seq, respectively).
        iii. Additional metadata (such as clusters) and reduced dimensionalities are optional but will be made available for viewing.
        iv. Users must provide an appropriate reduced dimensional assay (present in the Seurat object) to be used for the optional integration or large data workflows.
        v. Parallelism is opt-in: users must specify a multicore future plan to make use of shared memory parallelism (otherwise the default sequential, non-parallel plan is used).
    b. A named list of gene sets (signatures)
        i. The nomenclature of the genes in the data and those in the signatures should match (gene symbols or Entrez/Ensembl IDs belonging to the same organism); translation between different nomenclatures is left to the user.
        ii. Each signature should be a plain R character vector.
2. Create a SeqtometryGUI object with the new method (since it is an R6 object).
    a. See the associated roxygen2 documentation for details regarding object initialization.
3. Invoke the run_gui function using the newly created object to run the Shiny app.
    a. An overview of the UI elements of the Shiny app is available below in section 3.

Example usage:

```
s <- readRDS("path/to/your/seurat_object.rds")
g <- readRDS("path/to/your/gene_signatures.rds")
u <- SeqtometryGUI::SeqtometryGUI$new(s, g)
u$run_gui()
```

Section 3 (Seqtometry GUI overview)

See the image below for numbered GUI elements and their corresponding purpose in a typical workflow.



1. Side panel: contains various elements for controlling and viewing the gating hierarchy.
2. Collapsible tree selector: represents the current gating hierarchy.
   a. Select the level of the hierarchy for viewing in the main panel (active level is highlighted).
3. Gating buttons:
   a. Add gate: create or update gates based on selected region from one of the main plots (updates the “Gate” field for the current level of the hierarchy). The name of the new gate is supplied in the input text field (see 4).
   b. Apply gates: creates a new level in the gating hierarchy (see 2) containing all cells whose “Gate” field (see 3a) is not equal to “Unassigned”. The name of this new level is supplied in the input text field (see 4).
   c. Clear gates: resets the “Gate” field for all cells in the current level to “Unassigned”.
4. Input text field: provide names for gates (see 3a) and levels (see 3b) here.
5. Accessory buttons:
   a. Remove selected subtree: prunes the selected level (and all levels below it) from the gating hierarchy (see 2).
   b. Recalculate scores: reruns the Seqtometry scoring algorithm on the selected level of the gating hierarchy (see 2) for all signatures to improve resolution following a change in relative point of reference.
6. Output text field: provides messages regarding the status of operations from the side panel. It will alert users regarding missing input if they did not provide it (see 3a, 3b, and 4) and will signal successful completion of an action associated with a button press (see 3a, 3b, 3c, 5a, and 5b).
7. Grouping selector: dropdown menu for selection of a categorical variable used to tabulate statistics (see 8) for the current level of the gating hierarchy (see 2).
8. Statistics table: contains counts and percentages for the selected categorical variable (see 7) in the current level of the gating hierarchy (see 2).
9. Main panel: contains interactive plots for the current level of the gating hierarchy (see 2).
10. Left plot selectors: controls the content of the left plot (see 12)
    a. Select X1: dropdown selector for the x-axis
    b. Select Y1: dropdown selector for the y-axis
    c. Select Z1: dropdown selector for the z-axis (displayed via color overlays)
    d. Select P1: dropdown selector for plot type (scatter, contour, density, violin, or hidden).
11. Right plot selectors: controls the content of the right plot (see 13) in an analogous manner to 10.
12. Left plot: dynamic Plotly chart for visualization of the current level of the gating hierarchy (see 2). Contains controls for:
    a. Lasso selection (rectangular or freeform) for creating gates (see 3a)
    b. Fine view adjustments (such as zoom or pan)
    c. Hiding and showing cell subsets via the associated interactive legend, which groups cells according to the categorical variable selected on the Z-axis
    d. Plot export (save to .png file) for additional annotation if needed
13. Right plot: companion Plotly chart with the same capabilities as the left plot (see 12) that can be used to obtain a second view of the same data with different axes or a different plot type.