cyTRON: Installation Guide

August 4, 2017

This guide is intended to help the user to install the software, get the R libraries, and setup the environmental variables required for the correct execution of cyTRON. It is not an exhaustive step-by-step guide and it is working on Windows 10, Ubuntu 17.04, and macOS Sierra at the moment of writing.

1 Required Software

The software required for the execution of cyTRON is the following:

- Java SE Development Kit (all OS)
- Cytoscape (all OS)
- R (Windows, Ubuntu, macOS)

You can download the installers from the links between parenthesis. Remember that TRONCO requires R 3.4 or higher.

2 Setup

Once you have installed the software listed above, you can proceed by setting up the environment. Follow one of the guides, depending on your OS.

2.1 Windows

2.1.1 R Setup

Open an R session and run the following commands:

- `update.packages(ask = FALSE)`
- `install.packages("rJava", dependencies=TRUE)`
- `install.packages("devtools", dependencies=TRUE)`
- `library("devtools")`
- `install_github("BIMIB-DISCo/TRONCO", ref="cyTRONdev")`

- `system.file("jri", package="rJava")`
- `Sys.getenv("R_HOME")`
- `.libPaths()`

Remember not to exit the R session since you will need get back to it later.

2.1.2 Environment Setup

At first, you have to make JRI available to JVMs:

- go to the folder specified by the R command `system.file("jri", package="rJava")`
- copy the `jri.dll` file to the `C:\Windows\System32` folder

Note that if you are running an x64 OS, you have to copy the `jri.dll` file inside the `x64` folder located into the folder specified by `system.file("jri", package="rJava")`.

Also R has to be exported to JVMs:

- go to the installation folder of R and proceed to the `bin` folder
- go to the `x64` or `i386` depending on you OS
- copy all the `.dll` files (i.e. `R.dll`, `Rblas.dll`, `Rgraphapp.dll`, `Rinconv.dll`, and `Rlapack.dll`) to the `C:\Windows\System32` folder

If `C:\Windows\System32` is not a valid Java Library Path, you have to move the copied `.dll` files into a valid one. You can reach a list of valid Java Library Paths by running Cytoscape in debug mode:

- open a new cmd session
- navigate to the Cytoscape installation folder (e.g. `cd C:\Program Files\Cytoscape_v3.5.1`)
- run `cytoscape.bat`
- get the list Java Library Paths from the cmd

The Java Library Paths are output only if the cyTRON app has been installed and are separated by colons.

Finally, you need to set two environmental variables:

- `R_HOME` with the value obtained by the R command `Sys.getenv("R_HOME")`
- `R_LIBS_USER` with the values obtained by the R command `.libPaths()` divided by semicolons

For a comprehensive guide on how to set environmental variables in Windows 10 and previous, refer to this web page. Do not forget to reboot at the end of the process.

2.2 Ubuntu

2.2.1 R Setup

Before setting up R directly, open a Terminal session and run the following commands:

- `sudo apt-get update`
- `sudo apt-get install libcurl4-openssl-dev libssl-dev`
- `sudo apt-get install r-cran-igraph`

Now, you can open an R session and run the following commands:

- `update.packages(ask = FALSE)`
- `install.packages("rJava", dependencies=TRUE)`
- `install.packages("devtools", dependencies=TRUE)`
- `library("devtools")`
- `install_github("BIMIB-DISCo/TRONCO", ref="cyTRONdev")` (installation should fail at the first try)
- `install.packages("gRapHD", dependencies=TRUE)`
- `install_github("BIMIB-DISCo/TRONCO", ref="cyTRONdev")`
- `system.file("jri", package="rJava")`
- `Sys.getenv("R_HOME")`

Remember not to exit the R session since you will need get back to it later.

2.2.2 Environment Setup

At first, you have to make JRI available to JVMs:

- open a new Terminal session
- navigate to the folder specified by the R command `system.file("jri", package="rJava")`
- execute `cp libjri.so /usr/lib`

If the `cp` command fails, replace `/usr/lib` with a valid Java Library Path. You can reach a list of valid Java Library Paths by running Cytoscape in debug mode:

- open a new Terminal session
- navigate to the Cytoscape installation folder (e.g. `cd ~/Cytoscape_v3.5.1`)

- run `./cytoscape.sh`
- get the list Java Library Paths from the Terminal

The Java Library Paths are output only if the cyTRON app has been installed and are separated by colons.

Finally, you need to set the `R_HOME` environmental variable:

- open a new Terminal session
- run `sudo nano /etc/environment`
- add the variable `R_HOME` with the value obtained by the R command `Sys.getenv("R_HOME")` (e.g. `R_HOME="/usr/lib/R"`)
- save and exit from `nano`
- reboot

2.3 macOS

2.3.1 R Setup

Open an R session and run the following commands:

- `update.packages(ask = FALSE)`
- `install.packages("rJava", dependencies=TRUE)`
- `install.packages("devtools", dependencies=TRUE)`
- `library("devtools")`
- `install_github("BIMIB-DISCo/TRONCO", ref="cyTRONdev")` (installation should fail at the first try)
- `install.packages("gRapHD", dependencies=TRUE)`
- `install_github("BIMIB-DISCo/TRONCO", ref="cyTRONdev")`
- `system.file("jri", package="rJava")`
- `Sys.getenv("R_HOME")`

Remember not to exit the R session since you will need get back to it later.

2.3.2 Environment Setup

At first, you have to make JRI available to JVMs:

- open a new Terminal session
- navigate to the folder specified by the R command `system.file("jri", package="rJava")`
- execute `cp libjri.jnilib /Library/Java/Extensions`

If the `cp` command fails, replace `/Library/Java/Extensions` with a valid Java Library Path. You can reach a list of valid Java Library Paths by running Cytoscape in debug mode:

- open a new Terminal session
- navigate to the Cytoscape installation folder (e.g. `cd /Applications/Cytoscape_v3.5.1`)
- run `./cytoscape.sh`
- get the list Java Library Paths from the Terminal

The Java Library Paths are output only if the cyTRON app has been installed and are separated by colons.

Finally, you need to set the `R_HOME` environmental variable:

- open a new Terminal session
- run `nano ~/.bash_profile`
- export the variable `R_HOME` with the value obtained by the R command `Sys.getenv("R_HOME")` (e.g. `export R_HOME="Library/Frameworks/R.framework/Resources"`)
- save and exit from `nano`
- reboot