Contents

[Package medplot usage instructions 1](#_Toc355642194)

[General usage instructions 3](#_Toc355642195)

[“Stop R Server” button 4](#_Toc355642196)

[“Update medplot R package” button 4](#_Toc355642197)

[Protected cells and sheets 5](#_Toc355642198)

[Closing the Excel file 5](#_Toc355642199)

[PlotSymptoms.xlsm spreadsheet usage instructions 6](#_Toc355642200)

[“SETTINGS” sheet 6](#_Toc355642201)

[“PATIENTS” sheet 6](#_Toc355642202)

[“DATA” sheet 7](#_Toc355642203)

[“ERRORS” sheet 7](#_Toc355642204)

[Running the plotting function 7](#_Toc355642205)

[PlotTests.xlsm spreadsheet usage instructions 9](#_Toc355642206)

[“SETTINGS sheet 9](#_Toc355642207)

[“PARAMETERS” sheet 10](#_Toc355642208)

[“DATA” sheet 10](#_Toc355642209)

[“ERRORS” sheet 11](#_Toc355642210)

[Running the plotting function 11](#_Toc355642211)

[Bibliography 13](#_Toc355642212)

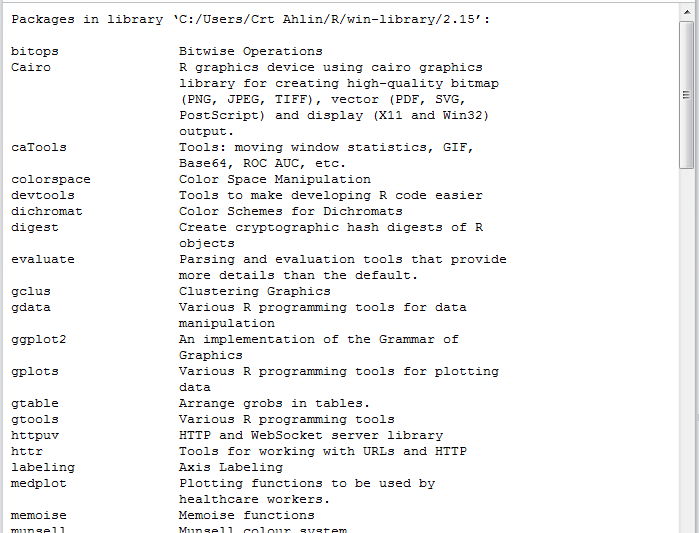
# Package medplot usage instructions

The medplot package contains several MS Excel (<http://office.microsoft.com/en-us/excel/> ) files which you can fill with your own data (let us call them template files). You can generate graphs from this data by clicking an appropriate button in Excel toolbar. A Visual Basic for Applications (VBA) script starts, using RExcel (<http://rcom.univie.ac.at/> ) (Baier & Neuwirth, 2007) connector to connect MS Excel to the R statistical environment (R Core Team, 2013), which handles the plotting.

We will assume you have successfully installed the *medplot* package for the R language and all the other required packages (installation is described in another document). Note that only 32 bit versions of MS Excel are supported and that the package might not work on Excel 2013 (not tested) and Excel versions for Mac computers (not tested)

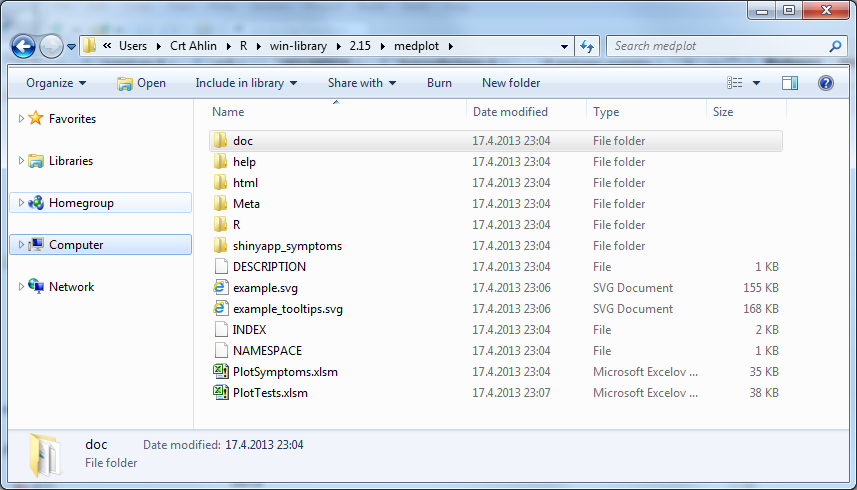
You can find the Excel template files in one of the subfolders of the medplot package folder. To see where your medplot package is installed, issue the following command at the R prompt:

> library()



E.g.: the screenshot reveals the medplot package is inside the folder C:/Users/Crt Ahlin/R/win-library/2.15.

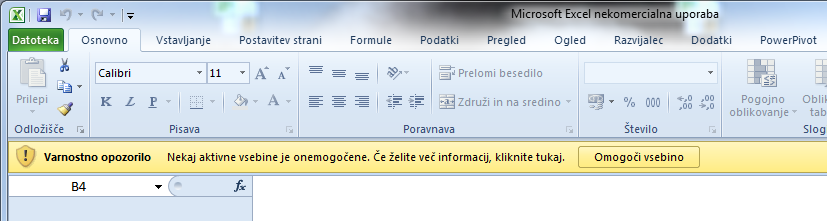
Navigate to the folder where medplot is installed, and enter the medplot subfolder.

E.g.: the contents of the medplot subfolder.

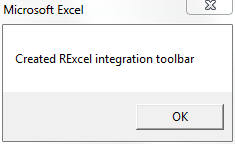
The Excel template files have the .xslm suffix, indicating they contain macros. Copy these files to your favorite working folder. You can also rename them, if you wish. **Always work on these copies to avoid losing data when updating the medplot package.**

## General usage instructions

When you open one of the files for the first time, you will probably get a security warning that some active features of the files are disabled.

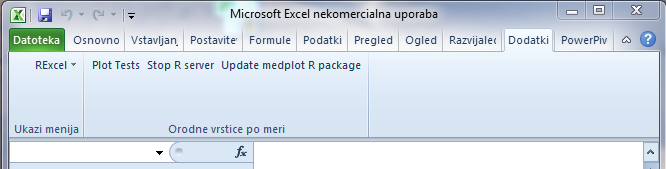


You should allow these to enable macros in these files to run, otherwise the medplot functionalities will not be available. If you get a warning like in above case, simply click the button on the right side of the warning to enable them. If the required permissions are granted, you should get a message that the RExcel integration toolbar was generated.



E.g.: the RExcel integration toolbar was created.

You can find the generated toolbar under the Addins tab. A general RExcel toolbar is on the left side of the tab and the toolbar specific to medplot is on the right (assuming you have no other custom toolbars, of course; see screenshot below).



E.g.: The RExcel general toolbar and the medplot toolbar with buttons.

Clicking one of the buttons causes VBA (Visual Basic for Applications) code to be called. Some buttons are common to all the Excel template files and will be described in the following subchapters.

### “Stop R Server” button

Clicking the Stop R Server button will stop the R process, if one is running. This is only meant to be used if R misbehaves in some way, to stop it. It will start again automatically when another function requiring R is called.

### “Update medplot R package” button

Clicking the Update medplot R package button will start the R process (via RExcel add-in) and attempt to update the medplot package with the version in the GitHub repository (<https://github.com/crtahlin/medplot>). You should have a working internet connection. The whole medplot subfolder in the R library gets updated if the update is successful. This includes both the R code and the Excel template files that contain the VBA code. This is meant to be used for on-the-fly code updates (hot fixes).

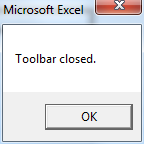
Note that you can probably still continue to use the Excel file that you have open at the moment of update; unless the update was also to the VBA code in the Excel file and the update is such, that the old VBA code is no longer compatible with the new functionality. In that case, you should use the new Excel template files, transferring your data to them.

### Protected cells and sheets

The sheets and cells in the Excel template files are protected in a sense, that the user cannot edit cells that he is not supposed to edit. The protection can be removed by users (no password is used), but this is discouraged as more mistakes are possible without protection enabled.

### Closing the Excel file

You should get a message that the RExcel toolbar was closed when closing the Excel file.



E.g.: The RExcel integration toolbar was closed.

If you do not get the message, Excel was probably not closed correctly and the toolbar will persist in your following Excel sessions, even if they are not using the medplot package. If you wish to resolve this, open and close the Excel medplot file again.

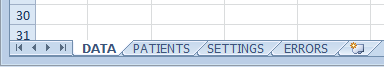
In the next chapters, each of the Excel template file specifics will be described separately.

## PlotSymptoms.xlsm spreadsheet usage instructions

PlotSymptoms.xlsm relies on the the R shiny package (RStudio & Inc., 2013), which runs a simple web server on the local machine and renders a plot the user can manipulate via a web browser.

The PlotSymptoms.xlsm contains the following sheets:

* DATA
* PATIENTS
* SETTINGS
* ERRORS



E.g.: Sheets in PlotSymptoms.xlsm.

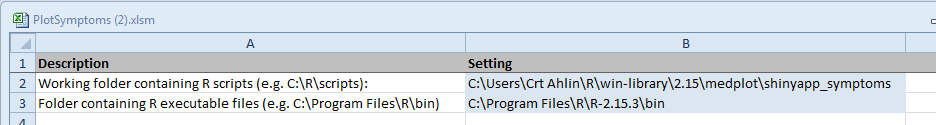
The sheets will be described in the order the user should refer to them in the following subparagraphs.

### “SETTINGS” sheet

The SETTINGS sheet contains the parameters the user must set before running the plotting function. The user should set the correct values in the light blue fields (the rest of the fields are write protected and read-only).

Working folder containing R scripts (e.g. C:\R\scripts): set path to the folder containing the R code for the shiny application, e.g. the Symptoms.R file. The code is installed together with the medplot package and should be a subfolder of the medplot folder, named “shinyapp\_symptoms”. (see screenshot for example)

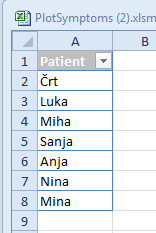
Folder containing R executable files (e.g. C:\Program Files\R\bin): set path to the folder containing the R executable files, e.g. Rscript.exe. This depends on your operating system and R version. (see screenshot for example)



E.g.: Screenshot of the SETTINGS sheet.

### “PATIENTS” sheet

The PATIENTS sheet contains names of the patients. For data integrity purposes only patients listed on this sheet can be entered on the DATA sheet.

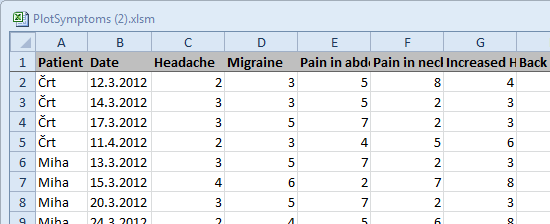


E.g.: Sample patients on the PATIENTS sheet.

### “DATA” sheet

The DATA sheet contains the patients, the dates of checkups and the severity of reported symptoms. Each line of data represents one date of one patient checkup.

The user can edit the column names after the Date column, which represent the symptom names. Names in the Patient column should be entered as they are listed on the PATIENTS sheet. Names not listed on this sheet are not valid. Dates in the Date column should be entered in DD.MM.YYYY format. Severity of symptoms should be entered a an integer value between and including 0 and 10.



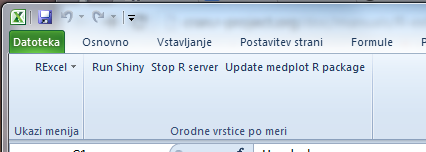
E.g.: Sample data on the DATA sheet.

### “ERRORS” sheet

Potential errors get listed on the ERRORS sheet. The user should check the sheet after running the plotting function.

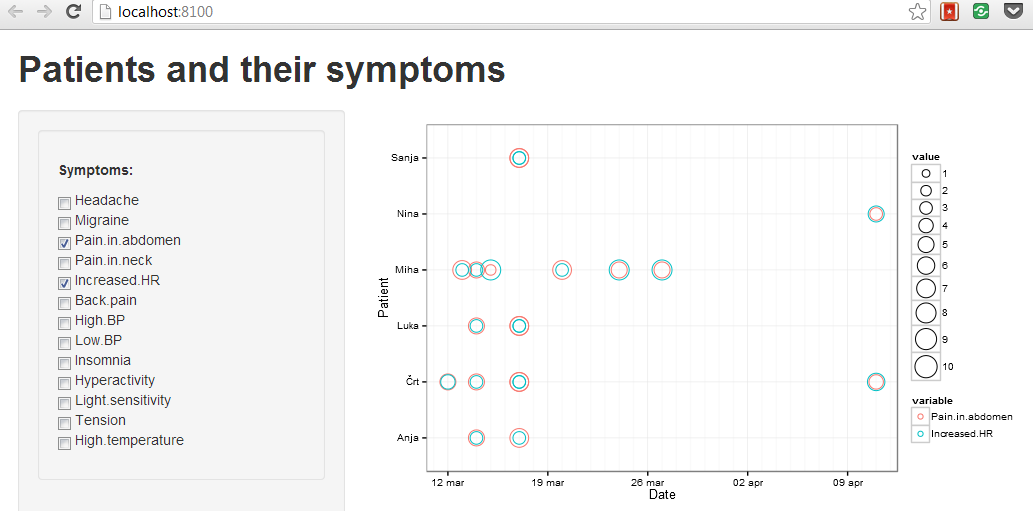
### Running the plotting function

The user can start the plotting function by clicking the button Run Shiny on the medplot toolbar.



E.g.: Example of the Run Shiny button.

A web browser window should open with options to select which symptoms the user wishes to plot. The plot updates every time after the user changes the selection.



E.g.: Example of the generated plot.

The size of the circles on the plot represents symptom severity while the color of the circle represents a particular symptom.

Note that the shiny web server runs on the local host – the firewall on the user’s computer has to allow connections to the server in order for the web browser to be able to connect to it.

## PlotTests.xlsm spreadsheet usage instructions

PlotTest.xlsm relies on the R SVGAnnotation package (Nolan & Lang, 2012) to generate a scalable vector graphics (SVG) file containing the plots.

The PlotTests.xlsm contains the following sheets:

* DATA
* PARAMETERS
* SETTINGS
* ERRORS

The sheets will be described in the order the user should refer to them in the following subparagraphs.

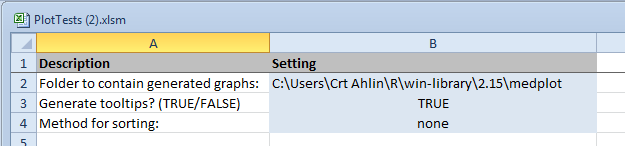
### “SETTINGS sheet

The SETTINGS sheet contains the parameters the user must set before running the plotting function. The user should set the correct values in the light blue fields (the rest of the fields are write protected and read-only).

Folder to contain generated graphs: set path to a folder that will contain the files containing the graphs. **Note that existing files containing graphs in this path will be overwritten.**

Generate tooltips? (TRUE/FALSE): set to TRUE if you wish the generated SVG file to contain the functionality of generating pop-up windows with extra information about the data point when the user moves the mouse cursor over a point in the graph (tooltips). This is slower than generating a graph without tooltips.

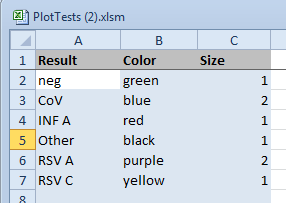
Method for sorting: select an option how to sort the units in the generated graph. None – does not sort the units in the graph and uses the order in which they are listed in the Excel sheet. DateIn – will sort according to the DateIn column, which contains dates of arrivals. The rest of the options use the R package seriation (Hahsler, Hornik, & Buchta, 2007) for sorting, using the values of test results entered on a particular date. BEA – will sort via the bond energy algorithm; BEA\_TSP – uses the travelling salesperson algorithm; PCA – uses the first principal component for sorting. The idea behind sorting is that patterns of test results might emerge visually if the units are sorted on the graph.



E.g.: The SETTINGS sheet example.

### “PARAMETERS” sheet

The PARAMETERS sheet contains the names of valid test results under the Result column, the color for a particular test result under the Color column and the size of the dot on the graph in the Size column. The light blue colored fields are editable by the user.



E.g.: The PARAMETERS sheet example.

### “DATA” sheet

The DATA sheet contains data about subjects (e.g. patients or medical staff) and their test results on different test days.

The columns used in the sheet are:

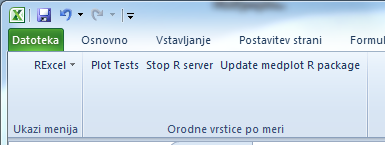
* ID: identity code of the subject; alphanumeric
* Name: name of the subject; alphanumeric
* Age: age of the subject; numeric
* Sex: sex of the subject; string
* Type: type of subject; alphanumeric  
  The label used in this field determines the 1st level group under which the subject will be placed on the graph. Examples are: Patient, Staff. Can be any label the user chooses. Can be left blank.
* Diagnosis: the diagnosis the patient started treatment for; alphanumeric  
  The label used in this field determines the 2nd level group under which the subject will be placed on the graph. Can be any label the user chooses. Can be left blank.
* Outcome: the final outcome of the treatment; alphanumeric  
  Can be any label the user chooses. The label “died” plots a special character on the graph at the date under DateOut column. Examples are: improved, died.
* DateIn: the date the treatment started; expected format: DD.MM.YYYY
* DateOut: the date the treatment ended; expected format: DD.MM.YYYY
* The date columns: they contain the dates of tests taken as the column header (expected format: DD.MM.YYYY) and the labels of positive test results on that particular day for a particular subject in the cells. Multiple positive test results should be separated by a comma (e.g.: “CoV,Other”). The user can edit (add) the dates column header labels.

### “ERRORS” sheet

Potential errors and warnings get listed on the ERRORS sheet. The user should check the sheet after running the plotting function. For example, errors might occur if invalid test result labels are used on the DATA sheet. Compared to errors, warnings usually do not stop the plotting, but are caused situations the user should be aware of.

### Running the plotting function

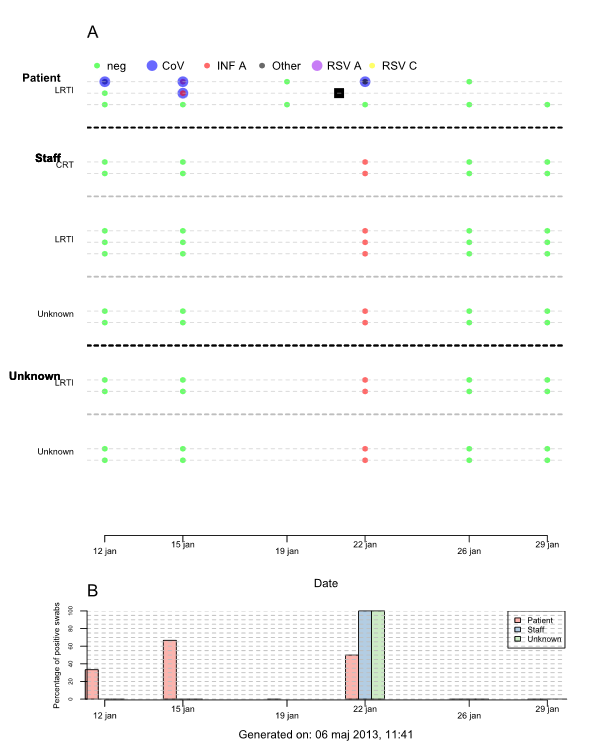
The user can start the plotting function by clicking the button Plot Tests on the medplot toolbar.



E.g.: The Plot Test button on the medplot toolbar.

Depending on the amount of data, the plot generation can take a while. Potential errors are reported on the ERRORS sheet. The user should look for the generated graphs inside the folder specified on the SETTINGS sheet. Depending on the other settings used, the folder should contain one or two image files with .svg suffix. They can be opened with SVG viewer applications or most modern browsers (e.g. Chrome version 26+, Internet Explorer 10).

The plots should look similar to the plot bellow, depending on the version of medplot used.



E.g.: Example of Plot Tests plot.

The graph under the label A shows test results for subjects on different days of testing. Moving the mouse over a plotted dot will raise a window with additional details about the subject (on the plot with tooltips, not on the “normal” SVG plot). The graph under the label B shows the percentage of positive swabs for different 1st level groups of subjects on different dates via a bar chart. All test results that are not “neg” or NULL are considered positive.

## Bibliography

Baier, T., & Neuwirth, E. (2007). Excel :: COM :: R. *Computational Statistics*, *22*(1, April 2007), 91–108. doi:10.1007/s00180-007-0023-6

Hahsler, M., Hornik, K., & Buchta, C. (2007). Getting Things in Order: An introduction to the R package seriation, (2001). Retrieved from http://epub.wu.ac.at/852/

Nolan, D., & Lang, D. T. (2012). Interactive and Animated Scalable Vector Graphics. *Journal of Statistical Software*, *46*(1).

R Core Team. (2013). R: A Language and Environment for Statistical Computing. Vienna, Austria. Retrieved from http://www.r-project.org/

RStudio, & Inc. (2013). shiny: Web Application Framework for R. Retrieved from http://cran.r-project.org/package=shiny