|  |  |  |  |
| --- | --- | --- | --- |
|  | 26/04/2017 |  | |
| |  |  | | --- | --- | |  |  | |  |  | |  |  | |  |  | | |  | |
| The Moving Epidemics Method  *The mem Shiny web application* | | | |
|  |  | | |  |  |  | | --- | --- | --- | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |  |  |  | |
|  |  | | José Eugenio Lozano Alonso and Jakob Bergström  1 Foundation Institute of Health Sciences Studies of Castilla y León (Spain)  2 The Public Health Agency (Sweden) |

The Moving Epidemics Method

The mem Shiny web application

Tabla de contenido

[Introduction 2](#_Toc481149293)

[Running the app 2](#_Toc481149294)

[Run memshy locally 2](#_Toc481149295)

[Installation 3](#_Toc481149296)

[Shiny app 5](#_Toc481149297)

[Run memshy remotely 6](#_Toc481149298)

[The memshy application 6](#_Toc481149299)

[Check & describe 6](#_Toc481149300)

[Model 6](#_Toc481149301)

[Surveillance 6](#_Toc481149302)

[Visualize 7](#_Toc481149303)

# Introduction

The Moving Epidemics Method (MEM) is a tool developed in Castilla y León (Spain) to help in the routine influenza surveillance in health systems. It gives a better understanding of the annual influenza epidemics and allows the weekly assessment of the epidemic status and intensity.

Thought in its conception it was originally created to be used with influenza data and health sentinel networks, MEM has been tested with different diseases and surveillance systems so nowadays it can be used with any disease which present a seasonal accumulation of cases that can be considered an epidemic.

MEM development started in 2000 and the first record of is existence is dated in 2003 in the Options for the Control of Influenza V(Vega Alonso et al. 2004).

It was presented to the baselines working group of the European Influenza Surveillance Scheme (EISS) in the 12th EISS Annual Meeting (Malaga, Spain, 2007), with whom started a collaboration that continued when EISS was dissolved in 2008 to create the European Influenza Surveillance Network.

In 2009 MEM appears for the first time in an official European document: the Who European guidance for influenza surveillance in humans. A year later MEM was implemented in the European Centre for Disease Prevention and Control (ECDC) platform, and in 2012, after piloting, in the World Health Organization Regional Office for Europe (WHO-E).

As a result of the collaboration with ECDC and WHO-E, two papers have been published, one related to the establishment of epidemic thresholds(Vega et al. 2013) and other in the comparison of intensity levels in Europe(Vega et al. 2015).

In 2014 a tool was created to help users around the world to apply mem on their data. It was released in July as a library for R, a free software environment for statistical computing and graphics. It is available at the official repositories: The Comprehensive R Archive Network (CRAN), it is the stable mem version(José E Lozano Alonso, n.d.).

In 2015 the second version of the mem R library was published open source at GitHub, a web-based Git or version control repository and Internet hosting service. This is available directly from github(Lozano, José E 2017) and is considered as the development version and includes a lot of new features and graphics.

In 2017 a web application was created to serve as a graphical user interface for the R mem library using a web application framework for R called Shiny. This application is based on the development version of the mem R library.

# Running the app

There are two options to run the web application:

* Locally
* Remotely

To run it locally you need to install R in your computer, set up the app dependencies and start it.

To run it remotely, you need to connect to the server where the app is hosted, no need to install R or worry about dependencies since all is already set up in the server side.

## Run memshy locally

### Installation

The mem Shiny web application (memshy) is based on the mem R library and requires R to work. R is available as Free Software under the terms of the Free Software Foundation’s GNU General Public License in source code form. It compiles and runs on a wide variety of UNIX platforms and similar systems (including FreeBSD and Linux), Windows and MacOS. There are binaries for most operating systems at its official web page(The R Foundation 2017). To install download the binaries appropriate for your system and proceed to install it.

R is a command line program but there are a lot of graphical user interfaces available to users that wants a friendlier environment. The most popular is RStudio an open source powerful and productive user interface for R. Binaries can be downloaded from its official web(RStudio, n.d.) and installed on Windows, Mac, and Linux.

Shiny is a web application framework for R created by the RStudio team, there is no need to install separately because it acts as a library for the R language and will be installed with the rest of dependencies.

MEM Shiny app is a set of two files that Shiny framework is able to interpret in order to start the web application. They can be run directly from a remote web server or in a local directory of our hard disc (running a local server).

#### Dependencies

Packages are collections of R functions, data, and compiled code in a well-defined format. The directory where packages are stored is called the library. R is based on the contribution of the community, which creates their own code and share it creating new packages, which extend the features of the base package.

Most packages has requirements, they need other packages to work, these are the dependencies of the package.

MEM Shiny app and mem R library requires to install a set of packages (dependencies) on R to start the application. The list of dependencies required for both applications are those of the original mem R library and those added by the new features of the mem Shiny web application.

Mem R library requirements:

* sm
* boot
* grDevices
* graphics
* stats
* sqldf
* reshape2
* RColorBrewer
* mixtools

memshy requeriments:

* shiny
* shinythemes
* shinydashboard
* shinyjs
* shinyBS
* plotly
* ggplot2
* ggthemes
* R.utils
* openxlsx
* XLConnect
* stringr
* readr
* magick
* DT
* gplots
* RODBC
* mem
* shinysky

Almost all the libraries can be installed directly from the CRAN repositories. To install a package simply write in the command line:

install.package(“packagename”)

Installation of packages must be done only once, after a package is installed, to use it, it has to be loaded, no need to install it again.

library(“packagename”)

To automatize the work of installing and loading libraries, here it is a custom function to check all dependencies and install in case it is needed for memshy.

testinstall.packages <- function(i.packages) {

lapply(i.packages, function (x) if(sum(installed.packages()[, 1]%in%x)==0) install.packages(x))

lapply(i.packages, require, character.only=TRUE)

lapply(i.packages, function(x) paste(x, packageVersion(x)))

}

>testinstall.packages(c("shiny", "shinythemes", "shinydashboard", "shinyjs", "RColorBrewer", "shinyBS",

"plotly", "ggplot2", "ggthemes", "reshape2", "R.utils", "openxlsx", "XLConnect",

"stringr", "readr", "magick", "DT", "gplots", "RODBC", "mixtools"))

There are two missing packages in the above code, requires special installation, the mem R library is hosted at github, and the shinysky package does not provide binaries to the lastest version of R, so it has to be installed from sources, also at github.

To install from github another package must be installed: devtools.

install.package(“devtools”)

And to install and load the last two packages:

# Install mem development version

if ("mem" %in% installed.packages()[,"Package"]){

if (as.numeric(as.character(packageVersion("mem")))<2){

testinstall.packages("devtools")

devtools::install\_github("lozalojo/mem")

}

}else{

testinstall.packages("devtools")

devtools::install\_github("lozalojo/mem")

}

library("mem")

# There are no binaries for shinysky

if (!("shinysky" %in% installed.packages()[,"Package"])) devtools::install\_github("AnalytixWare/ShinySky")

library("shinysky")

In windows, the only package that can give problems installing is rJava (required by XLConnect), which requires Java installed in your computer ([www.java.com](http://www.java.com)).

In \*nix systems, there are libraries that need some specific packages installed on your system, each system is different, but these are some issues we’ve found while installing on debian and arch linux machines:

* hexbin (required by plotly) requires a fortran compiler.
* rJava (required by XLConnect) requires Java.
* RODBC requires an odbc package (unixodbc).

Finally, memshy is able to read access files. In \*nix systems it is required to install mdbtools (<http://mdbtools.sourceforge.net/>), which is probably in the distribution repositories.

Once all requirements are fulfilled, you can start the application.

If you find any troubles installing dependencies, please refer to the installation chapter of the manual of each specific library to find a solution for your OS.

### Shiny app

Normally, shiny apps consist on two text files:

* server.R
* ui.R

The file ui.R sets the user interface of the application and server.R configure the server side of the application, running all processed, producing the outputs and creating the graphs that will be returned to the interface.

Someone can share the files with you, or you can download them from the internet. Those two files are hosted at github:

<https://github.com/jakob2025/MEM-app>

Just click on clone to download a zip file containing the app and extract the two required files to any directory of your hard disk.

From RStudio (or from R), to run a Shiny app you only need to pass the directory where you extracted the files:

runApp(path.to.the.files, launch.browser = T)

Set the launch.browser to F (false) if you want to use the R internal browser to open the application.

Alternatively you can run directly applications from github without storing them in your hard drive, this option is recommended since you always start the application that is hosted at the official repositories, so you are sure it is always updated to the latest version.

shiny::runGitHub('MEM-app', 'jakob2025')

This option requires an internet connection every time you run the app.

## Run memshy remotely

Start your browser and go to the server address:

<https://lozalojo.shinyapps.io/memshy>

The application will start your default browser and run the application.

# The memshy application

The application have four panels to navigate:

* Left panel: Allows to load data and has options that affects directly to the outputs of the application, like model or thresholds selection.
* Right panel: Has options to modify graphs, like the palette to use in plots or the axis names.
* Bottom panel: Options related to the mem function and all the parameters.
* Main panel: Where all outputs are displayed.

The main panel has four tabs:

* Check & describe
* Model
* Surveillance
* Visualize

While describing different outputs of the application we will explore more the options in the left, bottom and right panels.

## Loading data and table structure

Click load file to open a dataset. Currently memshy is able to read the following formats:

* Text: files, with any encoding, any separator and any decimal point.
* R: files with rds extension from R.
* MS Excel: files with xls extension from MS Excel prior to 2007 and xlsx from 2007 on.
* MS Access: files with mdb extension from MS Access prior to 2007, and accdb from 2007 on.

When the format is able to contain more than one dataset (excel files with multiple sheets or access files with multiple tables), the application will show all the datasets in the file to let you chose which one you one to load.

Independently from the format used to store the information, the table must have one row per surveillance week and one column per season. A season is a full surveillance period from the beginning to the end, where occurs at some point one single epidemic wave on it. No epidemic wave can be spared in two consecutive seasons. If so, you have to redefine the start and end of the season defined in your dataset. If a season have two waves, it must be split in two periods and must be named accordingly with the seasons name conventions described below.

The first row must contain the names of the seasons. This application understand the naming of a season when it contains one or two four digits year separated by ‘/’ and one one-digit number between parenthesis to identify the part of the season in case it has two different waves.

Examples:

* 2016/2017: Season starting at any point of 2016 (normally at 40th week) and ending in 2017 (normally at 20th week). This is normally used in northern hemisphere countries.
* 2016: Season starting and ending at 2016. Used in southern hemisphere countries.
* 2016/2016: The same as above.
* 2016/2017(1): First wave of season 2016/2017.
* 2016(1): Second wave of season 2016.
* 2016/2016(2): Second wave of season 2016.

The first column should contain the names of the weeks. When the season contains two different calendar years, the week will go from 40th of the first year to 52nd, and then from 1st to 20th. When the season contains one year, the weeks will go from 1st to 52nd.

If there is no column with week names, the application will name the weeks numbering from 1 to the number of rows.

## Data format

### Data series

Data must containing information from the historical series. Surveillance period can start and end at any given week (tipically start at week 40th and ends at week 20th), and data can have any units and can be expressed in any scale (typically rates per 100,000 inhabitants or consultations).

Each row is an epidemiological week, each column a surveillance season. Each cell the value for a given week in a given season.

The first column must contain the week name (epidemiological week). The first row contains the name of the seasons.

In the northern hemisphere countries, the surveillance period usually goes from week 40 to 20 of the following year (notation: season 2016-2017).

In the southern hemisphere countries, the surveillance period usually goes from week 18 to 39 same year (notation: season 2017).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week No | 2012/2013 | 2013/2014 | 2014/2015 | 2015/2016 | 2016/2017 |
| *40* |  |  |  |  |  |
| *41* |  |  |  |  |  |
| *…* |  |  |  |  |  |
| *17* |  |  |  |  |  |
| *18* |  |  |  |  |  |
| Northern hemisphere surveillance example | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Week No | 2013 | 2014 | 2015 | 2016 | 2017 |
| *18* |  |  |  |  |  |
| *19* |  |  |  |  |  |
| *…* |  |  |  |  |  |
| *37* |  |  |  |  |  |
| *38* |  |  |  |  |  |
| *39* |  |  |  |  |  |
| Southern hemisphere surveillance example | | | | | |

### Missing values and zero data

Be sure that a cell is blank when there is no data (missing value) and 0 when the data is 0 cases/rate.

### Week 53

Since not all the seasons contain week 53rd, data must be adjusted in order to avoid gaps. When a season has a rate for week 53rd it must be pulled forward one week. So week 53 becomes week 1 of the following year, week 1 then turns to be week 2 and so on:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Original season | |  | Modified season | |
| Week No | 2000/2001 |  | Week No | 2000/2001 |
| *50* | 112.1 |  | *50* | 112.1 |
| *51* | 150.4 |  | *51* | 150.4 |
| *52* | 170.7 |  | *52* | 170.7 |
| ***53*** | 210.2 | **->** | *1* | 210.2 |
| *1* | 270.0 |  | *2* | 270.0 |
| *2* | 280.0 |  | *3* | 280.0 |
| *3* | 160.2 |  | *4* | 160.2 |
| *4* | 120.4 |  | *5* | 120.4 |
| Fixing week 53 problem | | | | |

Week 53 must not be deleted, especially in northern hemisphere countries, since epidemics occur in winter time, and it is probable that week 53 is part of the epidemic. Deleting it will shorten the epidemic one week and also would affect the estimators, which probably use that weekly rate to calculate the thresholds.

### Two waves in the same season

Mem is designed to identify one single epidemic wave. When a season has two epidemics waves, data must be split in two, so that each column have information from one single wave.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Original season | |  | Modified season | | |
| Week No | 2000/2001 |  | Week No | 2000/2001(1) | 2000/2001(2) |
| *30* | 30.1 |  | *30* | 30.1 |  |
| *31* | 120.3 |  | *31* | 120.3 |  |
| *32* | 240.3 | *first* | *32* | 240.3 | *missing* |
| *33* | 231.6 | *epidemic* | *33* | 231.6 | *values* |
| *34* | 160.3 |  | *34* | 160.3 |  |
| *35* | 100.1 |  | *35* | 100.1 |  |
| *36* | 40.9 |  | *36* | 40.9 |  |
| *…* |  | **->** | *…* |  |  |
| *52* | 60.4 |  | *52* |  | 60.4 |
| *1* | 140.7 |  | *1* |  | 140.7 |
| *2* | 210.2 |  | *2* | *missing* | 210.2 |
| *3* | 270.0 | *second* | *3* | *values* | 270.0 |
| *4* | 120.0 | *epidemic* | *4* |  | 120.0 |
| *5* | 40.2 |  | *5* |  | 40.2 |
| *6* | 30.4 |  | *6* |  | 30.4 |
| Fixing two epidemics seasons | | | | | |

Seasons thus split must be named using the same notation and adding 1 or 2 at the end between parentheses.

## Check & describe

Check data series, timing and describe the data.

## Model

Summary, graphs, goodness and optimization of the MEM model.

## Surveillance

Surveillance tools.

## Visualize

Visualize different sets of data with a MEM model.