Package 'FloodMapper'

May 27, 2018

Title Cross-Scale Flooding Prediction under Heavy Precipitation

Version 1.0

Date 2018-05-13

Description A gridded model to facilitate cross-scale flooding prediction under heavy precipitation events. The model also allows the consideration of current urban stormwater systems and their future developments in order to help explore potential adaptation strategies against climate change. Reference: Wang, X., G. Huang, and J. Liu (2014) <doi:10.1002/2014JD022564>.

License MIT + file LICENSE

Depends R (>= 3.4.0)

NeedsCompilation no

LazyLoad no

Imports sp, raster, rgdal, magick

Author Xander Wang [aut, cre, cph]

Maintainer Xander Wang <xiuquan.wang@gmail.com>

Repository CRAN

Date/Publication 2018-05-27 13:56:39 UTC

R topics documented:

FM.animate 2
FM.init 3
FM.readDEM 4
FM.readDraininlets 4
FM.readInflow 5
FM.readLandcover 6
FM.readPrecip 7
FM.readSoil 8
FM.rerun 9
FM.start 10

Index **15**

FM.animate *Create An Animation for An Existing Model Run*

Description

Once a model run is completed successfully, users can use this function to create an animation into a *.gif file.

Usage

```
FM.animate(runname = "", workdir = "")
```

Arguments

runname	(OPTIONAL) the name of an existing model run. If it is empty, the current model run will be selected.
workdir	(OPTIONAL) full path to user's work directory. If it is empty, the work directory of current model run will be selected.

Author(s)

Xander Wang <xiuquan.wang@gmail.com>

Examples

```
## Load FloodMapper package
library(FloodMapper)

s_return <- NA

## Assume that the current model run with a name of "Lafayette_1km"
## has been completed successfully, you can run the following
## command to create an animation in *.gif format.

    s_return <- FM.animate("Lafayette_1km", workdir = tempdir())


## Check if there are any errors
if (!is.na(s_return)) {
    if (s_return != "") {
        cat("Creating animation is failed because of the following error:\n")
        cat(s_return)
    } else {
        cat("Creating animation is successful!\n")
    }
}
```

FM.init *Model Initialization*

Description

This function needs to be executed in order to initiate a FloodMapper model run. A unique run name should be specified to identify the model run.

Usage

```
FM.init(runname, startdatetime, enddatetime, outputinterval,
    internaltimestep = 30, debug = TRUE, rerun = FALSE,
    wdbreakpoints = NA, workdir = "")
```

Arguments

runname	a unique name to identify the model run. A folder with this name will be created under your current work directory.
startdatetime	start datetime of the model run, should be formated as "YYYY-MM-DD hh:mm:ss". E.g., "2016-06-09 12:00:00".
enddatetime	end datetime of the model run, should be formated as "YYYY-MM-DD hh:mm:ss". E.g., "2016-06-09 13:00:00".
outputinterval	time step for the model outputs, unit: seconds.
internaltimestep	(OPTIONAL) internal integral time step, unit: seconds, default: 30 seconds.
debug	(OPTIONAL) if the debug mode is enabled, a log file (named log.txt) will be generated under the run folder.
rerun	(OPTIONAL) assign TRUE if this is a rerun.
wdbreakpoints	(OPTIONAL) a vector defines how to display the flood depth with different colors (unit: mm). This setting will be only applied to the outputted PNG/PDF files.
workdir	Full path to a user-specified work directory. If not specified, the temporary folder defined by tempdir() will be used.

Author(s)

Xander Wang <xiuquan.wang@gmail.com>

Examples

```
## Load FloodMapper package
library(FloodMapper)

## Refer to Step 1 in the sample code of FM.start():
## Step 1. Initiate a new model run with a name of "Lafayette_1km"
```

FM.readDEM *Load Digital Elevation Model (DEM)*

Description

After a model run is initiated, this function should be first executed to load DEM raster.

Usage

```
FM.readDEM(demfile, waterdepth = 0, waterdepthfile = "", rerun = FALSE)
```

Arguments

demfile	full file path of the DEM raster.
waterdepth	(OPTIONAL) a number represents initial surface water depth (unit: m) to be applied to all grid cells.
waterdepthfile	(OPTIONAL) full file path of the initial surface water depth raster (unit: m). If a raster file is specified, the model will use this raster to initialize the surface water depth.
rerun	OPTIONAL, assign TRUE if this is a rerun.

Author(s)

Xander Wang <xiuquan.wang@gmail.com>

Examples

```
## Load FloodMapper package
library(FloodMapper)

## Refer to Step 2 in the sample code of FM.start():
## Step 2. Load the 1000km DEM raster for Lafayette Parish,
##         LA, USA from the sample dataset.
```

FM.readDraininlets *Load Raster for Street Drain Inlets (OPTIONAL)*

Description

After soil rasters are loaded, users may use this function to load the raster for street drain inlets in order to consider the effects of urban stormwater systems on mitigating floods.

Usage

```
FM.readDraininlets(draininletfile, rerun = FALSE)
```

Arguments

`draininletfile`	full file path of the drain inlet raster.
`rerun`	(OPTIONAL) assign TRUE if this is a rerun.

Author(s)

Xander Wang <xiuquan.wang@gmail.com>

Examples

```
## Load FloodMapper package
library(FloodMapper)

## Refer to Step 5 in the sample code of FM.start():
## Step 5. Load the 1000km raster for street drain inlet
```

`FM.readInflow` *Load Inflow Data (OPTIONAL)*

Description

After precipitatoin data is loaded, users may use this function to load inflow data in case that the selected domain is expected to receive water from its outside. The inflow data can come from monitoring stations or other raster datasets.

Usage

```
FM.readInflow(datatype = 0, txtfile, startdatetime, enddatetime, interval)
```

Arguments

`datatype`	0 = monitoring stations, 1 = raster data (NOT supported in this version)
`txtfile`	full file path of the text file required for inflow data. If the data comes from multiple monitoring stations, the format of this text file should be given as follows: #: ———————————————— #: ID Lon Lat Datafile #: 1 -92.0296 30.1800 in1.txt #: 2 -92.0260 30.1779 in2.txt #: 3 ... ... #: ———————————————— If the inflow data comes from rasters, the format of this text file should be given as follows: #: ———————————————— #: ID Rasterfile #: 1 in_2016-06-09_12-00-00.tif #: 2 in_2016-06-09_13-00-00.tif #: 3 ... #: ————————————————

startdatetime	start datetime of inflow records, should be formated as "YYYY-MM-DD hh:mm:ss". E.g., "2016-06-09 12:00:00".
enddatetime	end datetime of inflow records, should be formated as "YYYY-MM-DD hh:mm:ss". E.g., "2016-06-09 13:00:00".
interval	interval of inflow records, unit: seconds. E.g., 1 hr = 60 min * 60 sec = 3600 sec.

Author(s)

Xander Wang <xiuquan.wang@gmail.com>

Examples

```
## Load FloodMapper package
library(FloodMapper)

## Refer to Step 7 in the sample code of FM.start():
## Step 7. Load inflow data from one weather station
```

FM.readLandcover *Load Land Cover Raster*

Description

After DEM raster is loaded, this function should be executed to load land cover raster.

Usage

```
FM.readLandcover(lcfile, rerun = FALSE)
```

Arguments

lcfile	full file path of the land cover raster.
rerun	(OPTIONAL) assign TRUE if this is a rerun.

Author(s)

Xander Wang <xiuquan.wang@gmail.com>

Examples

```
## Load FloodMapper package
library(FloodMapper)

## Refer to Step 3 in the sample code of FM.start():
## Step 3. Load the 1000km land cover raster from the sample dataset
```

`FM.readPrecip` *Load Precipitation Data*

Description

After all rasters (DEM, land cover, and soil) are loaded, users may use this function to load precipitation data. Precipitation data can come from multiple weather stations or gridded remote sensing data.

Usage

```
FM.readPrecip(datatype = 0, txtfile, startdatetime, enddatetime,
    interval, idwpower = 2)
```

Arguments

`datatype`	0 = weather stations, 1 = raster data from NWP models, reanalysis datasets, or other remote sensing datasets.
`txtfile`	full file path of the text file required for precipitation data. If the data comes from multiple weather stations, the format of this text file should be given as follows: #: ———————————— #: ID Lon Lat Datafile #: 1 -92.0296 30.1800 p1.txt #: 2 -92.0260 30.1779 p2.txt #: 3 ... ... #: ———————————— If the precipitation data comes from rasters, the format of this text file should be given as follows: #: ———————————— #: ID Rasterfile #: 1 p_2016-06-09_12-00-00.tif #: 2 p_2016-06-09_13-00-00.tif #: 3 ... #: ————————————
`startdatetime`	start datetime of precipitation records, should be formated as "YYYY-MM-DD hh:mm:ss". E.g., "2016-06-09 12:00:00".
`enddatetime`	end datetime of precipitation records, should be formated as "YYYY-MM-DD hh:mm:ss". E.g., "2016-06-09 13:00:00".
`interval`	interval of precipitation records, unit: seconds. For example, 1 hr = 60 min * 60 sec = 3600 sec.
`idwpower`	(OPTIONAL) the power of IDW method, default: 2, suggested options: 1, 2, or 3.

Author(s)

Xander Wang <xiuquan.wang@gmail.com>

Examples

```
## Load FloodMapper package
library(FloodMapper)

## Refer to Step 6 in the sample code of FM.start():
## Step 6. Load precipitation data from one weather station
```

`FM.readSoil` *Load Rasters for Soil Texture, Soil Depth (Groudwater Level), and Soil Moisture Content*

Description

After land cover raster is loaded, this function should be executed to load rasters for soil texture, surface soil depth (groundwater level), and initial soil moisture content.

Usage

```
FM.readSoil(soilfile, soildepth, soildepthfile = "", soilmoistcontent,
    soilmoistcontentfile = "", rerun = FALSE)
```

Arguments

`soilfile`	full file path of the soil texture raster.
`soildepth`	a number indicates the depth of unsaturated soil (i.e., aquifer or groundwater level) to be applied to all grid cells, unit: m.
`soildepthfile`	full file path of the soil depth raster. If a raster file is specified, the model will use this raster to initialize the depth of unsaturated soil.
`soilmoistcontent`	a number indicates the initial soil moisture content (range: 0.0 - 1.0) to be applied to all grid cells.
`soilmoistcontentfile`	full file path of the soil moisture content raster. If a raster file is specified, the model will use this raster to initialize the soil moisture content.
`rerun`	(OPTIONAL) assign TRUE if this is a rerun.

Author(s)

Xander Wang <xiuquan.wang@gmail.com>

Examples

```
## Load FloodMapper package
library(FloodMapper)

## Refer to Step 4 in the sample code of FM.start():
## Step 4. Load the 1000km rasters for soil texture and
##         soil depth from the sample dataset, set the
##         initial soil moisture content to 0.4785.
```

FM.rerun *Rerun An Existing Model*

Description

Once a model run is completed successfully, users may want to adjust some parameters and rerun it. If this is the case, users can use this function.

Usage

```
FM.rerun(runname, startdatetime, enddatetime, outputinterval,
    internaltimestep = 30, debug = TRUE, wdbreakpoints = NA,
    workdir = "", animation = FALSE, bgtype = 0,
    aerialraster = "", pdfoutput = FALSE)
```

Arguments

runname	refer to FM.init().
startdatetime	refer to FM.init().
enddatetime	refer to FM.init().
outputinterval	refer to FM.init().
internaltimestep	refer to FM.init().
debug	refer to FM.init().
wdbreakpoints	refer to FM.init().
workdir	refer to FM.init().
animation	refer to FM.start().
bgtype	refer to FM.start().
aerialraster	refer to FM.start().
pdfoutput	refer to FM.start().

Author(s)

Xander Wang <xiuquan.wang@gmail.com>

Examples

```
## Load FloodMapper package
library(FloodMapper)

s_return <- NA

## Assume that the model run with a name of "Lafayette_1km" has been completed successfully,
## now you want to rerun this model after adjust some parameter files (under the input folder),
## you can run the following command to rerun the model.

    s_return <- FM.rerun(runname = "Lafayette_1km",
        startdatetime = "2016-08-12 10:00:00",
        enddatetime = "2016-08-12 11:00:00", outputinterval = 3600,
        internaltimestep = 300, wdbreakpoints = seq(0, 3000, 150),
        workdir = tempdir(), animation = TRUE, bgtype = 0,
        aerialraster = "", pdfoutput = FALSE)


## Check if there are any errors
if (!is.na(s_return)) {
    if (s_return != "") {
        cat("Model rerun is failed because of the following error:\n")
        cat(s_return)
    } else {
        cat("Model rerun is successful!\n")
    }
}
```

FM.start *Start A Model Run*

Description

After all required data are loaded successfully, users can use this function to start the current model run. If users want to run another model by specifying its run name, users should use FM.rerun().

Usage

```
FM.start(animation = FALSE, bgtype = 0, aerialraster = "", pdfoutput = FALSE)
```

Arguments

animation	FALSE = no animation png files will be generated; TRUE = png files for animation will be generated. This must be enabled if you want to create an animation later on.
bgtype	0 = transparent background; 1 = use DEM raster as the background; 2 = use user-defined aerial raster as the background (must specify the file name in the next parameter).

aerialraster	(OPTIONAL) full file path to an aerial image raster for your domain. If the aerial image is specified, it will be used as the background of all annimation png files.
pdfoutput	FALSE = no pdf files will be generated; TRUE = each time step will come with a pdf file showing surface water depth.

Author(s)

Xander Wang <xiuquan.wang@gmail.com>

Examples

```
## Load FloodMapper package
library(FloodMapper)

## Step 1. Initialize a new model run with a name of "Lafayette_1km"
s_runfolder <- file.path(tempdir(), "Lafayette_1km")
if (dir.exists(s_runfolder)) {
    ## delete this folder if it already exists
    unlink(s_runfolder, recursive = TRUE, force = TRUE)
}
s_return_sp1 <- FM.init(runname = "Lafayette_1km",
    startdatetime = "2016-08-12 10:00:00",
    enddatetime = "2016-08-12 11:00:00",
    outputinterval = 3600, internaltimestep = 300,
    wdbreakpoints = seq(0, 3000, 150), workdir = tempdir())

## Check if there are any errors
if (s_return_sp1 != "") {
    cat("Model initiation is failed because of the following error:\n")
    cat(s_return_sp1)
} else {
    cat("Model initiation is successful!\n")
}

## Step 2. Load the 1000km DEM raster for Lafayette Parish, LA, USA from the sample dataset
s_return_sp2 <- NA
if (s_return_sp1 == "") {
    s_return_sp2 <- FM.readDEM(system.file("extdata", "LF_DEM_1000m.tif",
                               package = "FloodMapper", mustWork = TRUE))

    ## Check if there are any errors
    if (s_return_sp2 != "") {
        cat("Loading DEM is failed because of the following error:\n")
        cat(s_return_sp2)
    } else {
        cat("Loading DEM is successful!\n")
    }
}

## Step 3. Load the 1000km land cover raster from the sample dataset
s_return_sp3 <- NA
```

```
if (!is.na(s_return_sp2)) {
    if (s_return_sp2 == "") {
        s_return_sp3 <- FM.readLandcover(system.file("extdata",
             "LF_land_1000m.tif", package = "FloodMapper", mustWork = TRUE))

        ## Check if there are any errors
        if (s_return_sp3 != "") {
            cat("Loading land cover is failed because of the following error:\n")
            cat(s_return_sp3)
        } else {
            cat("Loading land cover is successful!\n")
        }
    }
}

## Step 4. Load the 1000km rasters for soil texture and
##         soil depth from the sample dataset, set the
##         initial soil moisture content to 0.4785.
s_return_sp4 <- NA
if (!is.na(s_return_sp3)) {
    if (s_return_sp3 == "") {
        s_return_sp4 <- FM.readSoil(soilfile = system.file("extdata",
                  "LF_soil_1000m.tif", package = "FloodMapper", mustWork = TRUE),
                  soildepthfile = system.file("extdata", "LF_gwl_1000m.tif",
                             package = "FloodMapper", mustWork = TRUE),
                  soilmoistcontent = 0.4785)

        ## Check if there are any errors
        if (s_return_sp4 != "") {
            cat("Loading soil rasters is failed because of the following error:\n")
            cat(s_return_sp4)
        } else {
            cat("Loading soil rasters is successful!\n")
        }
    }
}

## Step 5. Load the 1000km raster for street drain inlet
s_return_sp5 <- NA
if (!is.na(s_return_sp4)) {
    if (s_return_sp4 == "") {
        ## Assume that the drain inlet raster is stored in LF_draininlet_1000m.tif
        s_return_sp5 <- FM.readDraininlets(draininletfile = system.file("extdata",
                           "LF_draininlet_1000m.tif", package = "FloodMapper",
                           mustWork = TRUE))

        ## Check if there are any errors
        if (!is.na(s_return_sp5)) {
            if (s_return_sp5 != "") {
             cat("Loading street drain inlet is failed because of the following error:\n")
                cat(s_return_sp5)
            } else {
                cat("Loading street drain inlet is successful!\n")
```

```
            }
        }
    }
}

## Step 6. Load precipitation data from one weather station
s_return_sp6 <- NA
if (!is.na(s_return_sp5)) {
    if (s_return_sp5 == "") {
        s_return_sp6 <- FM.readPrecip(datatype = 0,
            txtfile = system.file("extdata", "Pstations_1hr.txt",
                          package = "FloodMapper", mustWork = TRUE),
            startdatetime = "2016-08-12 10:00:00",
            enddatetime = "2016-08-12 11:00:00",
            interval = 3600)

        ## Check if there are any errors
        if (s_return_sp6 != "") {
           cat("Loading precipitation data is failed because of the following error:\n")
            cat(s_return_sp6)
        } else {
            cat("Loading precipitation data is successful!\n")
        }
    }
}

## Step 7. Load inflow data from one weather station
s_return_sp7 <- NA
if (!is.na(s_return_sp6)) {
    if (s_return_sp6 == "") {
      ## Assume that the information of inflow monitoring stations is stored in INstations_1hr.txt
s_return_sp7 <- FM.readInflow(datatype = 0,
            txtfile = system.file("extdata", "INstations_1hr.txt",
                          package = "FloodMapper", mustWork = TRUE),
            startdatetime = "2016-08-12 10:00:00",
            enddatetime = "2016-08-12 11:00:00", interval = 3600)

        ## Check if there are any errors
        if (!is.na(s_return_sp7)) {
            if (s_return_sp7 != "") {
                cat("Loading inflow data is failed because of the following error:\n")
                cat(s_return_sp7)
            } else {
                cat("Loading inflow data is successful!\n")
            }
        }
    }
}

## Step 8. Start a model run
s_return_sp8 <- NA
if (!is.na(s_return_sp7)) {
    if (s_return_sp7 == "") {
```

```
        s_return_sp8 <- FM.start(animation = TRUE, bgtype = 0,
                             aerialraster = "", pdfoutput = FALSE)

        ## Check if there are any errors
        if (s_return_sp8 != "") {
            cat("Model run is failed because of the following error:\n")
            cat(s_return_sp8)
        } else {
            cat("Model run is successful!\n")
        }
    }
}
```

Index

FM.animate, 2
FM.init, 3
FM.readDEM, 4
FM.readDraininlets, 4
FM.readInflow, 5
FM.readLandcover, 6
FM.readPrecip, 7
FM.readSoil, 8
FM.rerun, 9
FM.start, 10