Package ‘cffdrs’  
June 3, 2020

Type Package  
Title Canadian Forest Fire Danger Rating System  
Version 1.8.18  
Date 2020-06-03  
Depends R(>= 3.2.2), rgdal, raster, foreach  
Imports data.table, geosphere, doParallel  
License GPL-2  
URL https://r-forge.r-project.org/projects/cffdrs/  
BugReports https://r-forge.r-project.org/tracker/?func=browse&group_id=1970&atid=5372  
Encoding UTF-8  
LazyData true  
NeedsCompilation no  
Author Alan Cantin [cre, aut],  
Xianli Wang [aut],  
Marc-André Parisien [aut],  
Mike Wotton [aut],  
Kerry Anderson [aut],  
Brett MOore [aut],  
Tom Schiks [aut],  
Mike Flannigan [aut]  
Maintainer Alan Cantin alan.cantin@canada.ca
The cffdrs package allows R users to calculate the outputs of the two main components of the Canadian Forest Fire Danger Rating System (CFFDRS; [http://cwfis.cfs.nrcan.gc.ca/background/summary/fdr](http://cwfis.cfs.nrcan.gc.ca/background/summary/fdr)): the Fire Weather Index (FWI) System ([http://cwfis.cfs.nrcan.gc.ca/background/summary/fwi](http://cwfis.cfs.nrcan.gc.ca/background/summary/fwi)) and the Fire Behaviour Prediction (FBP) System ([http://cwfis.cfs.nrcan.gc.ca/background/summary/fbp](http://cwfis.cfs.nrcan.gc.ca/background/summary/fbp)) along with additional methods created and used Canadian fire modelling. These systems are widely used internationally to assess fire danger (FWI System) and quantify fire behavior (FBP System).

The FWI System (Van Wagner 1987) is based on the moisture content and the effect of wind of three classes of forest fuels on fire behavior. It consists of six components: three fuel moisture codes (Fire Fuel Moisture Code, Duff Moisture Code, Drought Code), and three fire behavior indexes representing rate of spread (Initial Spread Index), fuel consumption (Buildup Index), and fire
intensity (Fire Weather Index). The FWI System outputs are determined from daily noon weather observations: temperature, relative humidity, wind speed, and 24-hour rainfall.

The FBP System (Forestry Canada Fire Danger Group 1992; Hirsch 1996) provides a set of primary and secondary measures of fire behavior. The primary outputs consist of estimates of fire spread rate, fuel consumption, fire intensity, and fire description (i.e., surface, intermittent, or crown fire). The secondary outputs, which are not used nearly as often, give estimates of fire area, perimeter, perimeter growth rate, and flank and back fire behavior based on a simple elliptical fire growth model. Unlike the FWI System, which is weather based, the FBP System also requires information on vegetation (hereafter, fuel types) and slope (if any) to calculate its outputs. Sixteen fuel types are included in the FBP System, covering mainly major vegetation types in Canada.

Details

Package: cffdrs
Type: Package
Version: 1.8.18
Date: 2020-06-03
License: GPL-2

This package includes eleven functions. Seven functions, fwi, fwiRaster, hffmc, hffmcRaster, sdmc, gfmc, and wDC are used for FWI System calculation, whereas two functions, fbp and fbpRaster are used for FBP System calculation. One function, fireSeason determines fire season start and end dates based on weather. Two functions pros and lros are rate of spread and direction calculations across triangles. These functions are not fully independent: their inputs overlap greatly and the users will have to provide FWI System outputs to calculate FBP System outputs. The fwi, fwiRaster, and sdmc functions calculate the outputs based on daily noon local standard time (LST) weather observations of temperature, relative humidity, wind speed, and 24-hour rainfall, as well as the previous day’s moisture content. The hffmc, gfmc, and hffmcRaster functions calculate the outputs based on hourly weather observations of temperature, relative humidity, wind speed, and hourly rainfall, as well as the previous hour’s weather conditions. The fbp and fbpRaster functions calculate the outputs of the FBP System based on given set of information about fire weather conditions (weather observations and their associated FWI System components), fuel type, and slope (optional).

Author(s)

Xianli Wang, Alan Cantin, Marc-André Parisien, Mike Wotton, Kerry Anderson, Brett Moore, Tom Schiks, and Mike Flannigan

Maintainer: Alan Cantin <Alan.Cantin@Canada.ca>

References


See Also

fbp, fireSeason, fwi, fwiRaster, gfmc, hffmc, hffmcRaster, lros, pros, sdmc, wDC

Examples

#Calculating daily FWI with overwintering DC

# This exercise demonstrates how to calculate daily FWI System variables given a chronical two years
data daily fire weather observations from one weather station. In the example, we showed first how to
decline fire season start and end dates with fireSeason, we then made overwintering DC adjustment
# with wDC for the second fire season, and eventually calculated the daily FWI System variables over
# two fire seasons with fwi. All these steps were packed up into an example user’s function, which
# could be modified by various user groups. Note: the data used in this example is also the test
# data for wDC.
library(cffdrs)

#Example of a customised function to calculate fwi and
#overwinter DC. This could be further modified by
#users with various needs.
fwifwicwDC <- function(input){
  all.fwi <- NULL
  curYr.fwi <- NULL
  #Create date variable
  input$date <- as.Date(as.POSIXlt(paste(input$yr, "-", input$mon, "-", input$day, sep="")))

  #use default fire season start and end temperature thresholds
  fs <- fireSeason(input)
  #Fire season dates, ordered chronologically
  fs <- with(fs,fs[order(yr,mon,day),])
  #Create same Date format as weather dataset for comparison
  fs$date <- as.Date(as.POSIXlt(paste(fs$yr,"-",fs$mon,"-",fs$day,sep="")))

  theyears <- unique(fs$yr)
for(curYr.row in 1:length(theyears)){
  curYr <- theyears[curYr.row]
  curYr.d <- fs[fs$yr==curYr,]
  curYr.init <- data.frame(ffmc=80,dmc=10,dc=16) #set an initial startup values

  #if there is more than one year of data, accumulate precipitation, then calculate overwinterDC
  #and continue
  if(curYr.row > 1){
    #calculate the overwinter period
    #end of last year's fire season
    curYr.owd <- curYr.fsd[nrow(curYr.fsd),]
    #rbind with beginning of current year's fire season
    curYr.owd <- rbind(curYr.owd, curYr.d[1,])

    #accumulate precipitation for the period between end of last and start of current
    curYr.owdata <- sum(input[(input$date>curYr.owd[1,"date"] &
                               input$date < curYr.owd[2,"date"]),]$prec)
    owDC <- wDC(DCf=tail(curYr.fwi$DC,n=1),rw=curYr.owdata) #calculate overwinter DC value
    curYr.init <- data.frame(ffmc=80,dmc=10,dc=owDC) #Initialize moisture codes
  }

  curYr.fsd <- curYr.d[c(1,nrow(curYr.d)),] #get first and last dates of this year
  #match input data to those dates for fire season data
  curYr.fsdata <- input[input$yr == curYr & input$date >= curYr.fsd[1,"date"] &
                        input$date <= curYr.fsd[2,"date"],]

  #run fwi on fireseason data
  curYr.fwi <- fwi(curYr.fsdata,init=curYr.init)

  #force column names to be uppercase for consistency
  names(curYr.fwi) <- toupper(names(curYr.fwi))
  all.fwi <- rbind(all.fwi,curYr.fwi)
}
all.fwi

#Usage of the custom function
# Load the test dataset, which is also the test data for wDC:
data("test_wDC")
#select 1 weather station
localWX_1 <- test_wDC[test_wDC$id==1,]
#run function with the data and fire season values
fwi_withFSwDC <- fwi_fs_wDC(localWX_1)
#Check the resulting fwi indices, calculated with a fire season start and end date, and using
#overwintered DC
fwi_withFSwDC

fbp  

Fire Behavior Prediction System function
Description

fbp calculates the outputs from the Canadian Forest Fire Behavior Prediction (FBP) System (Forestry Canada Fire Danger Group 1992) based on given fire weather and fuel moisture conditions (from the Canadian Forest Fire Weather Index (FWI) System (Van Wagner 1987)), fuel type, date, and slope. Fire weather, for the purpose of FBP System calculation, comprises observations of 10 m wind speed and direction at the time of the fire, and two associated outputs from the Fire Weather Index System, the Fine Fuel Moisture Content (FFMC) and Buildup Index (BUI). FWI System components can be calculated with the sister function fwi.

Usage

```r
fbp(input, output="Primary", m=NULL, cores=1)
```

Arguments

- **input**: The input data, a data.frame containing fuel types, fire weather component, and slope (see below). Each vector of inputs defines a single FBP System prediction for a single fuel type and set of weather conditions. The data.frame can be used to evaluate the FBP System for a single fuel type and instant in time, or multiple records for a single point (e.g., one weather station, either hourly or daily for instance) or multiple points (multiple weather stations or a gridded surface). All input variables have to be named as listed below, but they are case insensitive, and do not have to be in any particular order. Fuel type is of type character; other arguments are numeric. Missing values in numeric variables could either be assigned as NA or leave as blank.

Required Inputs:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description/Full name</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Unique identifier of a weather station or spatial point (no restriction on data type)</td>
<td>N/A</td>
</tr>
<tr>
<td>FuelType</td>
<td>FBP System Fuel Types including &quot;C-1&quot;, &quot;C-2&quot;, &quot;C-3&quot;, &quot;C-4&quot;, &quot;C-5&quot;, &quot;C-6&quot;, &quot;C-7&quot;, &quot;D-1&quot;, &quot;M-1&quot;, &quot;M-2&quot;, &quot;M-3&quot;, &quot;M-4&quot;, &quot;S-1&quot;, &quot;S-2&quot;, &quot;S-3&quot;, &quot;O-1a&quot;, and &quot;O-1b&quot;. &quot;WA&quot; and &quot;NF&quot; stand for &quot;water&quot; and &quot;non-fuel&quot;, respectively. The &quot;.&quot; in the Fuel Type names could be omitted, and the Fuel Type names are also case-insensitive.</td>
<td>&quot;C2&quot;</td>
</tr>
<tr>
<td>LAT</td>
<td>Latitude [decimal degrees]</td>
<td>55</td>
</tr>
<tr>
<td>LONG</td>
<td>Longitude [decimal degrees]</td>
<td>-120</td>
</tr>
<tr>
<td>FFMC</td>
<td>Fine fuel moisture code [FWI System component]</td>
<td>90</td>
</tr>
<tr>
<td>BUI</td>
<td>Buildup index [FWI System component]</td>
<td>60</td>
</tr>
<tr>
<td>WS</td>
<td>Wind speed [km/h]</td>
<td>10</td>
</tr>
<tr>
<td>GS</td>
<td>Ground Slope [percent]</td>
<td>0</td>
</tr>
<tr>
<td>Dj</td>
<td>Julian day</td>
<td>180</td>
</tr>
<tr>
<td>Aspect</td>
<td>Aspect of the slope [decimal degrees]</td>
<td>0</td>
</tr>
</tbody>
</table>

Optional Inputs (1): Variables associated with certain fuel types. These could be skipped.
if relevant fuel types do not appear in the input data.

<table>
<thead>
<tr>
<th>Input</th>
<th>Full names of inputs</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC</td>
<td>Percent Conifer for M1/M2 [percent]</td>
<td>50</td>
</tr>
<tr>
<td>PDF</td>
<td>Percent Dead Fir for M3/M4 [percent]</td>
<td>35</td>
</tr>
<tr>
<td>cc</td>
<td>Percent Cured for O1a/O1b [percent]</td>
<td>80</td>
</tr>
<tr>
<td>GFL</td>
<td>Grass Fuel Load [kg/m²]</td>
<td>0.35</td>
</tr>
</tbody>
</table>

**Optional Inputs (2):** Variables that could be ignored without causing major impacts to the primary outputs

<table>
<thead>
<tr>
<th>Input</th>
<th>Full names of inputs</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBH</td>
<td>Crown to Base Height [m]</td>
<td>3</td>
</tr>
<tr>
<td>WD</td>
<td>Wind direction [decimal degrees]</td>
<td>0</td>
</tr>
<tr>
<td>Accel</td>
<td>Acceleration: 1 = point, 0 = line</td>
<td>0</td>
</tr>
<tr>
<td>ELV*</td>
<td>Elevation [meters above sea level]</td>
<td>NA</td>
</tr>
<tr>
<td>BUIEff</td>
<td>Buildup Index effect: 1=yes, 0=no</td>
<td>1</td>
</tr>
<tr>
<td>D0</td>
<td>Julian day of minimum Foliar Moisture Content</td>
<td>0</td>
</tr>
<tr>
<td>hr</td>
<td>Hours since ignition</td>
<td>1</td>
</tr>
<tr>
<td>ISI</td>
<td>Initial spread index</td>
<td>0</td>
</tr>
<tr>
<td>CFL</td>
<td>Crown Fuel Load [kg/m²]</td>
<td>1.0</td>
</tr>
<tr>
<td>FMC</td>
<td>Foliar Moisture Content if known [percent]</td>
<td>0</td>
</tr>
<tr>
<td>SH</td>
<td>C-6 Fuel Type Stand Height [m]</td>
<td>0</td>
</tr>
<tr>
<td>SD</td>
<td>C-6 Fuel Type Stand Density [stems/ha]</td>
<td>0</td>
</tr>
<tr>
<td>theta</td>
<td>Elliptical direction of calculation [degrees]</td>
<td>0</td>
</tr>
</tbody>
</table>

**output**

FBP output offers 3 options (see details in **Values** section):

<table>
<thead>
<tr>
<th>Outputs</th>
<th>Number of outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary (default)</td>
<td>8</td>
</tr>
<tr>
<td>Secondary</td>
<td>34</td>
</tr>
<tr>
<td>All</td>
<td>42</td>
</tr>
</tbody>
</table>

**m**

Optimal number of pixels at each iteration of computation when `nrow(input)` >= 1000. Default `m = NULL`, where the function will assign `m = 1000` when `nrow(input)` is between 1000 and 500,000, and `m = 3000` otherwise. By including this option, the function is able to process large dataset more efficiently. The optimal value may vary with different computers.

**cores**

Number of CPU cores (integer) used in the computation, default is 1. By signing `cores > 1`, the function will apply parallel computation technique provided by the **foreach** package, which significantly reduces the computation time for large input data (over a million records). For small dataset, `cores=1` is actually faster.
* Elevation is only used in the calculation of Foliar Moisture Content (FMC). However, FMC can also be calculated without elevation input. The default is to not use elevation in the calculation of FMC.

**Details**

The Canadian Forest Fire Behavior Prediction (FBP) System (Forestry Canada Fire Danger Group 1992) is a subsystem of the Canadian Forest Fire Danger Rating System, which also includes the Canadian Forest Fire Weather Index (FWI) System. The FBP System provides quantitative estimates of head fire spread rate, fuel consumption, fire intensity, and a basic fire description (e.g., surface, crown) for 16 different important forest and rangeland types across Canada. Using a simple conceptual model of the growth of a point ignition as an ellipse through uniform fuels and under uniform weather conditions, the system gives, as a set of secondary outputs, estimates of flank and back fire behavior and consequently fire area perimeter length and growth rate.

The FBP System evolved since the mid-1970s from a series of regionally developed burning indexes to an interim edition of the nationally develop FBP system issued in 1984. Fire behavior models for spread rate and fuel consumption were derived from a database of over 400 experimental, wild and prescribed fire observations. The FBP System, while providing quantitative predictions of expected fire behavior is intended to supplement the experience and judgment of operational fire managers (Hirsch 1996).

The FBP System was updated with some minor corrections and revisions in 2009 (Wotton et al. 2009) with several additional equations that were initially not included in the system. This fbp function included these updates and corrections to the original equations and provides a complete suite of fire behavior prediction variables. Default values of optional input variables provide a reasonable mid-range setting. Latitude, longitude, elevation, and the date are used to calculate foliar moisture content, using a set of models defined in the FBP System; note that this latitude/longitude-based function is only valid for Canada. If the Foliar Moisture Content (FMC) is specified directly as an input, the fbp function will use this value directly rather than calculate it. This is also true of other input variables.

Note that Wind Direction (WD) is the compass direction from which wind is coming. Wind azimuth (not an input) is the direction the wind is blowing to and is 180 degrees from wind direction; in the absence of slope, the wind azimuth is coincident with the direction the head fire will travel (the spread direction azimuth, RAZ). Slope aspect is the main compass direction the slope is facing. Slope azimuth (not an input) is the direction a head fire will spread up slope (in the absence of wind effects) and is 180 degrees from slope aspect (Aspect). Wind direction and slope aspect are the commonly used directional identifiers when specifying wind and slope orientation respectively. The input theta specifies an angle (given as a compass bearing) at which a user is interested in fire behavior predictions; it is typically some angle off of the final spread rate direction since if for instance theta=RAZ (the final spread azimuth of the fire) then the rate of spread at angle theta (TROS) will be equivalent to ROS.

**Value**

fbp returns a dataframe with primary, secondary, or all output variables, a combination of the primary and secondary outputs.

Primary FBP output includes the following 8 variables:

- CFB: Crown Fraction Burned by the head fire
CFC  Crown Fuel Consumption [kg/m^2]
FD   Fire description (S=Surface, I=Intermittent, C=Crown)
HFI  Head Fire Intensity [kW/m]
RAZ  Spread direction azimuth [degrees]
ROS  Equilibrium Head Fire Rate of Spread [m/min]
SFC  Surface Fuel Consumption [kg/m^2]
TFC  Total Fuel Consumption [kg/m^2]

Secondary FBP System outputs include the following 34 raster layers. In order to calculate the reliable secondary outputs, depending on the outputs, optional inputs may have to be provided.

BE   BUI effect on spread rate
SF   Slope Factor (multiplier for ROS increase upslope)
ISI  Initial Spread Index
FFMC Fine fuel moisture code [FWI System component]
FMC  Foliar Moisture Content [%]
Do   Julian Date of minimum FMC
RSO  Critical spread rate for crowning [m/min]
CSI  Critical Surface Intensity for crowning [kW/m]
FROS Equilibrium Flank Fire Rate of Spread [m/min]
BROS Equilibrium Back Fire Rate of Spread [m/min]
HROSt Head Fire Rate of Spread at time hr [m/min]
FROSt Flank Fire Rate of Spread at time hr [m/min]
BROSt Back Fire Rate of Spread at time hr [m/min]
FCFB Flank Fire Crown Fraction Burned
BCFB Back Fire Crown Fraction Burned
FFI  Equilibrium Spread Flank Fire Intensity [kW/m]
BFI  Equilibrium Spread Back Fire Intensity [kW/m]
FTFC Flank Fire Total Fuel Consumption [kg/m^2]
BTFC Back Fire Total Fuel Consumption [kg/m^2]
DH   Head Fire Spread Distance after time hr [m]
DB   Back Fire Spread Distance after time hr [m]
DF   Flank Fire Spread Distance after time hr [m]
TI   Time to Crown Fire Initiation [hrs since ignition]
FTI  Time to Flank Fire Crown initiation [hrs since ignition]
BTI  Time to Back Fire Crown initiation [hrs since ignition]
LB   Length to Breadth ratio
LBt  Length to Breadth ratio after elapsed time hr
WSV  Net vectored wind speed [km/hr]
TROS*  Equilibrium Rate of Spread at bearing theta [m/min]
TROSt*  Rate of Spread at bearing theta at time t [m/min]
TCFB*  Crown Fraction Burned at bearing theta
TFI*  Fire Intensity at bearing theta [kW/m]
TTFC*  Total Fuel Consumption at bearing theta [kg/m^2]
TTI*  Time to Crown Fire initiation at bearing theta [hrs since ignition]

*These outputs represent fire behaviour at a point on the perimeter of an elliptical fire defined by a user input angle theta. theta represents the bearing of a line running between the fire ignition point and a point on the perimeter of the fire. It is important to note that in this formulation the theta is a bearing and does not represent the angle from the semi-major axis (spread direction) of the ellipse. This formulation is similar but not identical to methods presented in Wotton et al (2009) and Tymstra et al (2009).

Author(s)
Xianli Wang, Alan Cantin, Marc-André Parisien, Mike Wotton, Kerry Anderson, and Mike Flannigan

References

See Also
fwi,fbpRaster

Examples
library(cffdrs)
# The dataset is the standard test data for FBP system
# provided by Wotton et al (2009)
data("test_fbp")
head(test_fbp)
# id FuelType LAT LONG ELV FFMC BUI WS WD GS Dj D0 hr PC PDF GFL
#1 1 C-1 55 110 NA 90 130 20.0 0 15 182 NA 0.33333333 NA NA NA
# Primary output (default)
fbp(test_fbp)

# or
fbp(test_fbp,output="Primary")

# Secondary output
fbp(test_fbp,"Secondary")

# All output
fbp(test_fbp,"All")

# For a single record:
fbp(test_fbp[7,])

# For a section of the records:
fbp(test_fbp[8:13,])

# Fbp function produces the default values if no data is fed to
# the function:
fbp()

fbpRaster  

Raster-based Fire Behavior Prediction System Calculations

Description

fbpRaster calculates the outputs from the Canadian Forest Fire Behavior Prediction (FBP) System (Forestry Canada Fire Danger Group 1992) based on raster format fire weather and fuel moisture conditions (from the Canadian Forest Fire Weather Index (FWI) System (Van Wagner 1987)), fuel type, date, and slope. Fire weather, for the purpose of FBP System calculation, comprises observations of 10 m wind speed and direction at the time of the fire, and two associated outputs from the Fire Weather Index System, the Fine Fuel Moisture Content (FFMC) and Buildup Index (BUI). Raster-based FWI System components can be calculated with the sister function fwiRaster.

Usage

fbpRaster(input,output="Primary",select=NULL,m=NULL,cores=1)
Arguments

The input data, a RasterStack containing fuel types, fire weather component, and slope layers (see below). Each vector of inputs defines a single FBP System prediction for a single fuel type and set of weather conditions. The RasterStack can be used to evaluate the FBP System for a single fuel type and instant in time, or multiple records for a single point (e.g., one weather station, either hourly or daily for instance) or multiple points (multiple weather stations or a gridded surface). All input variables have to be named as listed below, but they are case insensitive, and do not have to be in any particular order. Fuel type is of type character; other arguments are numeric. Missing values in numeric variables could either be assigned as NA or leave as blank.

Required Inputs:

<table>
<thead>
<tr>
<th>Input</th>
<th>Description/Full name</th>
</tr>
</thead>
</table>

Fuel Type code
C-1 1
C-2 2
C-3 3
C-4 4
C-5 5
C-6 6
C-7 7
D-1 8
M-1 9
M-2 10
M-3 11
M-4 12
NF 13
O-1a 14
O-1b 15
S-1 16
S-2 17
S-3 18
WA 19

LAT Latitude [decimal degrees]
LONG Longitude [decimal degrees]
FFMC Fine fuel moisture code [FWI System component]
BUI Buildup index [FWI System component]
WS Wind speed [km/h]
GS Ground Slope [percent]
Dj Julian day
Aspect Aspect of the slope [decimal degrees]

Optional Inputs (1): Variables associated with certain fuel types. These could
be skipped if relevant fuel types do not appear in the input data.

<table>
<thead>
<tr>
<th>Input</th>
<th>Full names of inputs</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>PC</td>
<td>Percent Conifer for M1/M2 [percent]</td>
<td>50</td>
</tr>
<tr>
<td>PDF</td>
<td>Percent Dead Fir for M3/M4 [percent]</td>
<td>35</td>
</tr>
<tr>
<td>cc</td>
<td>Percent Cured for O1a/O1b [percent]</td>
<td>80</td>
</tr>
<tr>
<td>GFL</td>
<td>Grass Fuel Load [kg/m^2]</td>
<td>0.35</td>
</tr>
</tbody>
</table>

Optional Inputs (2): Variables that could be ignored without causing major impacts to the primary outputs

<table>
<thead>
<tr>
<th>Input</th>
<th>Full names of inputs</th>
<th>Defaults</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBH</td>
<td>Crown to Base Height [m]</td>
<td>3</td>
</tr>
<tr>
<td>WD</td>
<td>Wind direction [decimal degrees]</td>
<td>0</td>
</tr>
<tr>
<td>Accel</td>
<td>Acceleration: 1 = point, 0 = line</td>
<td>0</td>
</tr>
<tr>
<td>ELV*</td>
<td>Elevation [meters above sea level]</td>
<td>NA</td>
</tr>
<tr>
<td>BUIEff</td>
<td>Buildup Index effect: 1=yes, 0=no</td>
<td>1</td>
</tr>
<tr>
<td>D0</td>
<td>Julian day of minimum Foliar Moisture Content</td>
<td>0</td>
</tr>
<tr>
<td>hr</td>
<td>Hours since ignition</td>
<td>1</td>
</tr>
<tr>
<td>ISI</td>
<td>Initial spread index</td>
<td>0</td>
</tr>
<tr>
<td>CFL</td>
<td>Crown Fuel Load [kg/m^2]</td>
<td>1.0</td>
</tr>
<tr>
<td>FMC</td>
<td>Foliar Moisture Content if known [percent]</td>
<td>0</td>
</tr>
<tr>
<td>SH</td>
<td>C-6 Fuel Type Stand Height [m]</td>
<td>0</td>
</tr>
<tr>
<td>SD</td>
<td>C-6 Fuel Type Stand Density [stems/ha]</td>
<td>0</td>
</tr>
<tr>
<td>theta</td>
<td>Elliptical direction of calculation [degrees]</td>
<td>0</td>
</tr>
</tbody>
</table>

output

FBP output offers 3 options (see details in Values section):

<table>
<thead>
<tr>
<th>Outputs</th>
<th>Number of outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>Primary (default)</td>
<td>8</td>
</tr>
<tr>
<td>Secondary</td>
<td>34</td>
</tr>
<tr>
<td>All</td>
<td>42</td>
</tr>
</tbody>
</table>

select

Selected outputs

m

m Optimal number of pixels at each iteration of computation when ncell(input) >= 1000. Default m = NULL, where the function will assign m = 1000 when ncell(input) is between 1000 and 500,000, and m=3000 otherwise. By including this option, the function is able to process large dataset more efficiently. The optimal value may vary with different computers.

cores

Number of CPU cores (integer) used in the computation, default is 1. By signing cores > 1, the function will apply parallel computation technique provided by
the foreach package, which significantly reduces the computation time for large input data (over a million grid points). For small dataset, cores=1 is actually faster.

* Elevation is only used in the calculation of Foliar Moisture Content (FMC). However, FMC can also be calculated without elevation input. The default is to not use elevation in the calculation of FMC.

Details

The Canadian Forest Fire Behavior Prediction (FBP) System (Forestry Canada Fire Danger Group 1992) is a subsystem of the Canadian Forest Fire Danger Rating System, which also includes the Canadian Forest Fire Weather Index (FWI) System. The FBP System provides quantitative estimates of head fire spread rate, fuel consumption, fire intensity, and a basic fire description (e.g., surface, crown) for 16 different important forest and rangeland types across Canada. Using a simple conceptual model of the growth of a point ignition as an ellipse through uniform fuels and under uniform weather conditions, the system gives, as a set of secondary outputs, estimates of flank and back fire behavior and consequently fire area perimeter length and growth rate.

The FBP System evolved since the mid-1970s from a series of regionally developed burning indexes to an interim edition of the nationally develop FBP system issued in 1984. Fire behavior models for spread rate and fuel consumption were derived from a database of over 400 experimental, wild and prescribed fire observations. The FBP System, while providing quantitative predictions of expected fire behavior is intended to supplement the experience and judgment of operational fire managers (Hirsch 1996).

The FBP System was updated with some minor corrections and revisions in 2009 (Wotton et al. 2009) with several additional equations that were initially not included in the system. This fbp function included these updates and corrections to the original equations and provides a complete suite of fire behavior prediction variables. Default values of optional input variables provide a reasonable mid-range setting. Latitude, longitude, elevation, and the date are used to calculate foliar moisture content, using a set of models defined in the FBP System; note that this latitude/longitude-based function is only valid for Canada. If the Foliar Moisture Content (FMC) is specified directly as an input, the fbp function will use this value directly rather than calculate it. This is also true of other input variables.

Note that Wind Direction (WD) is the compass direction from which wind is coming. Wind azimuth (not an input) is the direction the wind is blowing to and is 180 degrees from wind direction; in the absence of slope, the wind azimuth is coincident with the direction the head fire will travel (the spread direction azimuth, RAZ). Slope aspect is the main compass direction the slope is facing. Slope azimuth (not an input) is the direction a head fire will spread up slope (in the absence of wind effects) and is 180 degrees from slope aspect (Aspect). Wind direction and slope aspect are the commonly used directional identifiers when specifying wind and slope orientation respectively. The input theta specifies an angle (given as a compass bearing) at which a user is interested in fire behavior predictions; it is typically some angle off of the final spread rate direction since if for instance theta=RAZ (the final spread azimuth of the fire) then the rate of spread at angle theta (TROS) will be equivalent to ROS.

Value

fbpRaster returns a RasterStack with primary, secondary, or all output variables, a combination of the primary and secondary outputs. Primary FBP output includes the following 8 raster layers:
Secondary FBP System outputs include the following 34 raster layers. In order to calculate the reliable secondary outputs, depending on the outputs, optional inputs may have to be provided.
<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSV</td>
<td>Net vectored wind speed [km/hr]</td>
</tr>
<tr>
<td>TROS*</td>
<td>Equilibrium Rate of Spread at bearing theta [m/min]</td>
</tr>
<tr>
<td>TROSt*</td>
<td>Rate of Spread at bearing theta at time t [m/min]</td>
</tr>
<tr>
<td>TCFB*</td>
<td>Crown Fraction Burned at bearing theta</td>
</tr>
<tr>
<td>TFI*</td>
<td>Fire Intensity at bearing theta [kW/m]</td>
</tr>
<tr>
<td>TTFC*</td>
<td>Total Fuel Consumption at bearing theta [kg/m^2]</td>
</tr>
<tr>
<td>TTI*</td>
<td>Time to Crown Fire initiation at bearing theta [hrs since ignition]</td>
</tr>
</tbody>
</table>

*These outputs represent fire behaviour at a point on the perimeter of an elliptical fire defined by a user input angle theta. theta represents the bearing of a line running between the fire ignition point and a point on the perimeter of the fire. It is important to note that in this formulation the theta is a bearing and does not represent the angle from the semi-major axis (spread direction) of the ellipse. This formulation is similar but not identical to methods presented in Wotton et al (2009) and Tymstra et al (2009).

**Author(s)**

Xianli Wang, Alan Cantin, Marc-André Parisien, Mike Wotton, Kerry Anderson, and Mike Flannigan

**References**


**See Also**

`fbp`, `fwiRaster`, `hffmcRaster`

**Examples**

```r
# The dataset is the standard test data for FBP system
# provided by Wotton et al (2009), and randomly assigned
# to a stack of raster layers
input <- test_fbpRaster <- stack(system.file("extdata", "test_fbpRaster.tif", package="cffdrs"))
```
# Stack doesn't hold the raster layer names, we have to assign
# them:
names(input)<-c("FuelType","LAT","LONG","ELV","FFMC","BUI","WS","WD","GS","Dj","D0","hr","PC","PDF","GFL","cc","theta","Accel","Aspect","BUIEff","CBH","CFL","ISI")

# Primary outputs:
system.time(foo<-fbpRaster(input = input))
# Using the "select" option:
system.time(foo<-fbpRaster(input = input,select=c("HFI","TFC","ROS")))
# Secondary outputs:
system.time(foo<-fbpRaster(input = input,output="S"))
# All outputs:
#system.time(foo<-fbpRaster(input = input,output="A"))

### Additional, longer running examples ###
# Keep only the required input layers, the other layers would be
# assigned with default values:
# keep only the required inputs:
dat0<-input[[c("FuelType","LAT","LONG","FFMC","BUI","WS","GS","Dj","Aspect")]]

#system.time(foo<-fbpRaster(input = dat0,output="A"))

---

**fireSeason**  
*Fire Season Start and End*

**Description**

`fireSeason` calculates the start and end fire season dates for a given weather station. The current method used in the function is based on three consecutive daily maximum temperature thresholds (Wotton and Flannigan 1993, Lawson and Armitage 2008). This function process input from a single weather station.

**Usage**

```r
fireSeason(input,fs.start=12,fs.end=5,method="WF93", consistent.snow=FALSE, multi.year=FALSE)
```

**Arguments**

- **input**  
  A data.frame containing input variables of including the date/time and daily maximum temperature. Variable names have to be the same as in the following list, but they are case insensitive. The order in which the input variables are entered is not important either.

- **yr**  
  (required) Year of the observations

- **mon**  
  (required) Month of the observations

- **day**  
  (required) Day of the observations

- **tmax**  
  (required) Maximum Daily Temperature (degrees C)

- **snow_depth**  
  (optional) Is consistent snow data in the input?
fireSeason

fs.start  Temperature threshold (degrees C) to start the fire season (default=12)
fs.end    Temperature threshold (degrees C) to end the fire season (default=5)
method    Method of fire season calculation. Options are "wf93" or "la08" (default=WF93)
consistent.snow  Is consistent snow data in the input? (default=FALSE)
multi.year  Should the fire season span multiple years? (default=FALSE)

Details

An important aspect to consider when calculating Fire Weather Index (FWI) System variables is a definition of the fire season start and end dates (Lawson and Armitage 2008). If a user starts calculations on a fire season too late in the year, the FWI System variables may take too long to reach equilibrium, thus throwing off the resulting indices. This function presents two method of calculating these start and end dates, adapted from Wotton and Flannigan (1993), and Lawson and Armitage (2008). The approach taken in this function starts the fire season after three days of maximum temperature greater than 12 degrees Celsius. The end of the fire season is determined after three consecutive days of maximum temperature less than 5 degrees Celsius. The two temperature thresholds can be adjusted as parameters in the function call. In regions where temperature thresholds will not end a fire season, it is possible for the fire season to span multiple years, in this case setting the multi.year parameter to TRUE will allow these calculations to proceed.

This fire season length definition can also feed in to the overwinter DC calculations (wDC). View the cffdrs package help files for an example of using the fireSeason, wDC, and fwi functions in conjunction.

Value

fireSeason returns a data frame of season and start and end dates. Columns in data frame are described below.

Primary FBP output includes the following 8 variables:

yr       Year of the fire season start/end date
mon      Month of the fire season start/end date
day     Day of the fire season start/end date
fsdatetype  Fire season date type (values are either "start" or "end")
date    Full date value

Author(s)

Alan Cantin, Xianli Wang, Mike Wotton, and Mike Flannigan

References


See Also

fwi, wDC

Examples

library(cffdrs)
# The standard test data:
data("test_wDC")
print(head(test_wDC))
## Sort the data:
input <- with(test_wDC, test_wDC[order(id, yr, mon, day),])

# Using the default fire season start and end temperature
# thresholds:
a_fs <- fireSeason(input[input$id==1,])

# Check the result:
a_fs

# yr mon day fsdatetype
#1 1999 5 4 start
#2 1999 5 12 end
#3 1999 5 18 start
#4 1999 5 25 end
#5 1999 5 30 start
#6 1999 10 6 end
#7 2000 6 27 start
#8 2000 10 7 end

# In the resulting data frame, the fire season starts
# and ends multiple times in the first year. It is up to the user
# for how to interpret this.

# modified fire season start and end temperature thresholds
a_fs <- fireSeason(input[input$id==1,], fs.start=10, fs.end=3)
a_fs

# yr mon day fsdatetype
#1 1999 5 2 start
#2 1999 10 20 end
#3 2000 6 16 start
#4 2000 10 7 end

# select another id value, specify method explicitly
b_fs <- fireSeason(input[input$id==2,], method="WF93")
# print the calculated fire season
b_fs

# yr mon day fsdatetype
#1 1980 4 21 start
#2 1980 9 19 end
#3 1980 10 6 start
#4 1980 10 16 end
#5 1981 5 21 start
#6 1981 10 13 end
Fire Weather Index System

Description

fwi is used to calculate the outputs of the Canadian Forest Fire Weather Index (FWI) System for one day or one fire season based on noon local standard time (LST) weather observations of temperature, relative humidity, wind speed, and 24-hour rainfall, as well as the previous day’s fuel moisture conditions. This function could be used for either one weather station or for multiple weather stations.

Usage

fwi(input, init=data.frame(ffmc=85,dmc=6,dc=15, lat=55), batch=TRUE, out= "all", lat.adjust=TRUE, uppercase=TRUE)

Arguments

input A dataframe containing input variables of daily weather observations taken at noon LST. Variable names have to be the same as in the following list, but they are case insensitive. The order in which the input variables are entered is not important.

id (optional) Unique identifier of a weather station or spatial point (no restriction on data type); required when batch=TRUE
lat (recommended) Latitude (decimal degree, default=55)
long (optional) Longitude (decimal degree)
yr (optional) Year of observation; required when batch=TRUE
mon (recommended) Month of the year (integer 1-12, default=7)
day (optional) Day of the month (integer); required when batch=TRUE
temp (required) Temperature (centigrade)
rh (required) Relative humidity (%)
ws (required) 10-m height wind speed (km/h)
prec (required) 24-hour rainfall (mm)

init A data.frame or vector contains either the initial values for FFMC, DMC, and DC or the same variables that were calculated for the previous day and will be used for the current day’s calculation. The function also accepts a vector if the initial or previous day FWI values is for only one weather station (a warning message comes up if a single set of initial values is used for multiple weather stations). Defaults are the standard initial values for FFMC, DMC, and DC defined as the following:

ffmc Fine Fuel Moisture Code (FFMC; unitless) of the previous day. Default value is 85.
dmc Duff Moisture Code (DMC; unitless) of the previous day. Default value is 6.
dc Drought Code (DC; unitless) of the previous day. Default value is 15.
**lat**  Latitude of the weather station (optional, default=55). Latitude values are used to make day length adjustments in the function.

**batch**  Whether the computation is iterative or single step, default is TRUE. When batch=TRUE, the function will calculate daily FWI System outputs for one weather station over a period of time chronologically with the initial conditions given (init) applied only to the first day of calculation. If multiple weather stations are processed, an additional "id" column is required in the input to label different stations, and the data needs to be sorted by date/time and "id". If batch=FALSE, the function calculates only one time step (1 day) base on either the initial start values or the previous day’s FWI System variables, which should also be assigned to init argument.

**out**  The function offers two output options, out="all" will produce a data frame that includes both the input and the FWI System outputs; out="fwi" will generate a data frame with only the FWI system components.

**lat.adjust**  The function offers options for whether day length adjustments should be applied to the calculations. The default value is "TRUE".

**uppercase**  Output in upper cases or lower cases would be decided by this argument. Default is TRUE.

**Details**

The Canadian Forest Fire Weather Index (FWI) System is a major subsystem of the Canadian Forest Fire Danger Rating System, which also includes Canadian Forest Fire Behavior Prediction (FBP) System. The modern FWI System was first issued in 1970 and is the result of work by numerous researchers from across Canada. It evolved from field research which began in the 1930’s and regional fire hazard and fire danger tables developed from that early research.

The modern System (Van Wagner 1987) provides six output indices which represent fuel moisture and potential fire behavior in a standard pine forest fuel type. Inputs are a daily noon observation of fire weather, which consists of screen-level air temperature and relative humidity, 10 meter open wind speed and 24 accumulated precipitation.

The first three outputs of the system (the Fire Fuel Moisture Code (ffmc), the Duff Moisture Code (dmc), and the Drought Code (dc)) track moisture in different layers of the fuel making up the forest floor. Their calculation relies on the daily fire weather observation and also, importantly, the moisture code value from the previous day as they are in essence bookkeeping systems tracking the amount of moisture (water) in to and out of the layer. It is therefore important that when calculating FWI System outputs over an entire fire season, an uninterrupted daily weather stream is provided; one day is the assumed time step in the models and thus missing data must be filled in.

The next three outputs of the System are relative (unitless) indicators of aspects of fire behavior potential: spread rate (the Initial Spread Index, isi), fuel consumption (the Build-up Index, bui) and fire intensity per unit length of fire front (the Fire Weather Index, fwi). This final index, the fwi, is the component of the System used to establish the daily fire danger level for a region and communicated to the public. This final index can be transformed to the Daily Severity Rating (dsr) to provide a more reasonably-scaled estimate of fire control difficulty.

Both the Duff Moisture Code (dmc) and Drought Code (dc) are influenced by day length (see Van Wagner 1987). Day length adjustments for different ranges in latitude can be used (as described in
Lawson and Armitage 2008 (http://cfs.nrcan.gc.ca/pubwarehouse/pdfs/29152.pdf)) and are included in this R function; latitude must be positive in the northern hemisphere and negative in the southern hemisphere.

The default initial (i.e., "start-up") fuel moisture code values (FFMC=85, DMC=6, DC=15) provide a reasonable set of conditions for most springtime conditions in Canada, the Northern U.S., and Alaska. They are not suitable for particularly dry winters and are presumably not appropriate for different parts of the world.

Value

fwi returns a dataframe which includes both the input and the FWI System variables as described below:

Input Variables
Including temp, rh, ws, and prec with id, long, lat, yr, mon, or day as optional.

ffmc Fine Fuel Moisture Code
dmc Duff Moisture Code
dc Drought Code
isi Initial Spread Index
bui Buildup Index
fwi Fire Weather Index
dsr Daily Severity Rating

Author(s)
Xianli Wang, Alan Cantin, Marc-André Parisien, Mike Wotton, Kerry Anderson, and Mike Flannigan

References

See Also
fbp, fwiRaster, gfmc, hffmc, hffmcRaster, sdmc, wDC, fireSeason
Examples

library(cffdrs)
# The test data is a standard test dataset for FWI system (Van Wagner and Pickett 1985)
data("test_fwi")
# Show the data, which is already sorted by time:
head(test_fwi)
#long lat yr mon day temp rh ws prec
-100 40 1985 4 13 17 42 25 0
-100 40 1985 4 14 20 21 25 2.4
-100 40 1985 4 15 8.5 40 17 0
-100 40 1985 4 16 6.5 25 6 0
-100 40 1985 4 17 13 34 24 0
# (1) FWI System variables for a single weather station:
# Using the default initial values and batch argument,
# the function calculate FWI variables chronically:
fwi.out1<-fwi(test_fwi)
# Using a different set of initial values:
fwi.out2<-fwi(test_fwi,init=data.frame(ffmc=80, dmc=10, dc=16, lat=50))
# This could also be done as the following:
fwi.out2<-fwi(test_fwi,init=data.frame(80,10,6,50))
# Or:
fwi.out2<-fwi(test_fwi,init=c(80,10,6,50))
# Latitude could be ignored, and the default value (55) will be used:
fwi.out2<-fwi(test_fwi,init=data.frame(80,10,6))
# (2) FWI for one or multiple stations in a single day:
# Change batch argument to FALSE, fwi calculates FWI components based on previous day's fwi outputs:
fwi.out3<-fwi(test_fwi,init=fwi.out1,batch=FALSE)
# Using a suite of initials, assuming variables from fwi.out1 are the initial values for different records.
ininit_suite<fwi.out1[,c("FFMC","DMC","DC","LAT")]
# Calculating FWI variables for one day but with multiple stations. Because the calculations is for one time step, batch=FALSE:
fwi.out4<-fwi(test_fwi,init=init_suite,batch=FALSE)
# (3) FWI for multiple weather stations over a period of time:
# Assuming there are 4 weather stations in the test dataset, and they are ordered by day:
# running the function with the same default initial inputs, # will receive a warning message, but that is fine:
fwi(test_fwi)
# (4) Daylength adjustment:
# Change latitude values where the monthly daylength adjustments are different from the standard ones
# with daylength adjustment
fwi(test_fwi)[1:3,]
# Without daylength adjustment
fwi(test_fwi,lat.adjust=FALSE)[1:3,]

## Description

`fwiRaster` is used to calculate the outputs of the Canadian Forest Fire Weather Index (FWI) System for one day based on noon local standard time (LST) weather observations of temperature, relative humidity, wind speed, and 24-hour rainfall, as well as the previous day’s fuel moisture conditions. This function takes rasterized input and generates raster maps as outputs.

## Usage

```r
fwiRaster(input, init=c(ffmc=85, dmc=6, dc=15), mon=7, out="all", lat.adjust=TRUE, uppercase=TRUE)
```

## Arguments

- **input**: A stack or brick containing rasterized daily weather observations taken at noon LST. Variable names have to be the same as in the following list, but they are case insensitive. The order in which the inputs are entered is not important.
  - **lat** (recommended): Latitude (decimal degree, default=55)
  - **temp** (required): Temperature (centigrade)
  - **rh** (required): Relative humidity (%)
  - **ws** (required): 10-m height wind speed (km/h)
  - **prec** (required): 24-hour rainfall (mm)

- **init**: A vector that contains the initial values for FFMC, DMC, and DC or a stack that contains raster maps of the three moisture codes calculated for the previous day, which will be used for the current day’s calculation. Defaults are the standard initial values for FFMC, DMC, and DC defined as the following:
  - **ffmc**: Fine Fuel Moisture Code (FFMC; unitless) of the previous day. Default value is 85.
  - **dmc**: Duff Moisture Code (DMC; unitless) of the previous day. Default value is 6.
  - **dc**: Drought Code (DC; unitless) of the previous day. Default value is 15.
  - **lat**: Latitude of the weather station (optional, default=55). Latitude values are used to make day length adjustments in the function.

- **mon**: Month of the year (integer 1~12, default=7). Month is used in latitude adjustment (lat.adjust), it is therefore recommended when lat.adjust=TRUE was
The function offers two output options, out="all" will produce a raster stack include both the input and the FWI System outputs; out="fwi" will generate a stack with only the FWI system components.

The function offers options for whether latitude adjustments to day lengths should be applied to the calculations. The default value is "TRUE".

Output in upper cases or lower cases would be decided by this argument. Default is TRUE.

Details

The Canadian Forest Fire Weather Index (FWI) System is a major subsystem of the Canadian Forest Fire Danger Rating System, which also includes Canadian Forest Fire Behavior Prediction (FBP) System. The modern FWI System was first issued in 1970 and is the result of work by numerous researchers from across Canada. It evolved from field research which began in the 1930’s and regional fire hazard and fire danger tables developed from that early research.

The modern System (Van Wagner 1987) provides six output indices which represent fuel moisture and potential fire behavior in a standard pine forest fuel type. Inputs are a daily noon observation of fire weather, which consists of screen-level air temperature and relative humidity, 10 meter open wind speed and 24 accumulated precipitation.

The first three outputs of the system (the Fire Fuel Moisture Code, the Duff Moisture Code, and the Drought Code) track moisture in different layers of the fuel making up the forest floor. Their calculation relies on the daily fire weather observation and also, importantly, the code value from the previous day as they are in essence bookkeeping systems tracking the amount of moisture (water) in to and out of the layer. It is therefore important that when calculating FWI System outputs over an entire fire season, an uninterrupted daily weather stream is provided; one day is the assumed time step in the models and thus missing data must be filled in.

The next three outputs of the System are relative (unitless) indicators of aspects of fire behavior potential: spread rate (the Initial Spread Index), fuel consumption (the Build-up Index) and fire intensity per unit length of fire front (the Fire Weather Index). This final index, the fwi, is the component of the System used to establish the daily fire danger level for a region and communicated to the public. This final index can be transformed to the Daily Severity Rating (dsr) to provide a more reasonably-scaled estimate of fire control difficulty.

Both the Duff Moisture Code (dmc) and Drought Code (dc) are influenced by day length (see Van Wagner, 1987). Day length adjustments for different ranges in latitude can be used (as described in Lawson and Armitage 2008 (http://cfs.nrcan.gc.ca/pubwarehouse/pdfs/29152.pdf)) and are included in this R function; latitude must be positive in the northern hemisphere and negative in the southern hemisphere.

The default initial (i.e., "start-up") fuel moisture code values (FFMC=85, DMC=6, DC=15) provide a reasonable set of conditions for most springtime conditions in Canada, the Northern U.S., and Alaska. They are not suitable for particularly dry winters and are presumably not appropriate for different parts of the world.

Value

By default, fwi returns a raster stack which includes both the input and the FWI System variables, as describe below:
Inputs

Including temp, rh, ws, and prec with lat as optional.

ffmc  Fine Fuel Moisture Code

dmc   Duff Moisture Code

dc    Drought Code

isi   Initial Spread Index

bui   Buildup Index

fwi   Fire Weather Index

dsr   Daily Severity Rating

Author(s)

Xianli Wang, Alan Cantin, Marc-André Parisien, Mike Wotton, Kerry Anderson, and Mike Flannigan

References


See Also

fbp, fbpRaster, fwi, hffmc, hffmcRaster

Examples

library(cffdrs)
require(raster)
# The test data is a stack with four input variables including
# daily noon temp, rh, ws, and prec (we recommend tif format):
day01src <- system.file("extdata","test_rast_day01.tif",package="cffdrs")
day01 <- stack(day01src)
day01 <- crop(day01,c(250,255,47,51))
# assign variable names:
names(day01)<-c("temp","rh","ws","prec")
# (1) use the initial values
foo<-fwiRaster(day01)
plot(foo)
### Additional, longer running examples ###
# (2) use initial values with larger raster
day01 <- stack(day01src)
names(day01)<-c("temp","rh","ws","prec")
gfmc

Description
gfmc calculates both the moisture content of the surface of a fully cured matted grass layer and also an equivalent Grass Fuel Moisture Code (gfmc) (Wotton, 2009) to create a parallel with the hourly ffmc (see the fwi and hffmc functions). The calculation is based on hourly (or sub-hourly) weather observations of temperature, relative humidity, wind speed, rainfall, and solar radiation. The user must also estimate an initial value of the gfmc for the layer. This function could be used for either one weather station or multiple weather stations.

Usage
gfmc(input, GFMCold=85, batch=TRUE, time.step=1, roFL=0.3, out="GFMCandMC")

Arguments

input A dataframe containing input variables of daily noon weather observations. Variable names have to be the same as in the following list, but they are case insensitive. The order in which the input variables are entered is not important.

temp (required) Temperature (centigrade)

rh (required) Relative humidity (%)

ws (required) 10-m height wind speed (km/h)

prec (required) 1-hour rainfall (mm)

isol (required) Solar radiation (kW/m^2)

mon (recommended) Month of the year (integer 1-12)

day (optional) Day of the month (integer)

GFMCold Previous value of GFMC (i.e. value calculated at the previous time step)[default is 85 (which corresponds to a moisture content of about 16%)]. On the first calculation this is the estimate of the GFMC value at the start of the time step. The GFMCold argument can accept a single initial value for multiple weather stations, and also accept a vector of initial values for multiple weather stations. NOTE: this input represents the CODE value, not a direct moisture content value. The CODE values in the Canadian FWI System increase within decreasing moisture content. To roughly convert a moisture content value to a CODE value on the FF-scale (used in the FWI Systems FFMC) use GFMCold =101-gmc (where gmc is moisture content in %)

time.step Time step (hour) [default 1 hour]

roFL The nominal fuel load of the fine fuel layer, default is 0.3 kg/m^2
Whether the computation is iterative or single step, default is TRUE. When `batch=TRUE`, the function will calculate hourly or sub-hourly GFMC for one weather station over a period of time iteratively. If multiple weather stations are processed, an additional "id" column is required in the input to label different stations, and the data needs to be sorted by time sequence and "id". If `batch=FALSE`, the function calculates only one time step (1 hour) based on either the previous hourly GFMC or the initial start value.

Output format, default is "GFMCandMC", which contains both GFMC and moisture content (MC) in a data.frame format. Other choices include: "GFMC", "MC", and "ALL", which include both the input and GFMC and MC.

Details

The Canadian Forest Fire Danger Rating System (CFFDRS) is used throughout Canada, and in a number of countries throughout the world, for estimating fire potential in wildland fuels. This new Grass Fuel Moisture Code (GFMC) is an addition (Wotton 2009) to the CFFDRS and retains the structure of that System’s hourly Fine Fuel Moisture Code (HFFMC) (Van Wagner 1977). It tracks moisture content in the top 5 cm of a fully-cured and fully-matted layer of grass and thus is representative of typical after winter conditions in areas that receive snowfall. This new moisture calculation method outputs both the actual moisture content of the layer and also the transformed moisture Code value using the FFMC’s FF-scale. In the CFFDRS the moisture codes are in fact relatively simple transformations of actual moisture content such that decreasing moisture content (increasing dryness) is indicated by an increasing Code value. This moisture calculation uses the same input weather observations as the hourly FFMC, but also requires an estimate of solar radiation incident on the fuel.

Value

`gfmc` returns GFMC and moisture content (MC) values collectively (default) or separately.

Author(s)

Xianli Wang, Mike Wotton, Alan Cantin, and Mike Flannigan

References


See Also

`fwi`, `hffmc`
library(cffdrs)
#load the test data
data("test_gfmc")
# show the data format:
head(test_gfmc)
# yr mon day hr temp rh ws prec isol
# 1 2006 5 17 10 15.8 54.6 5.0 0 0.340
# 2 2006 5 17 11 16.3 52.9 5.0 0 0.380
# 3 2006 5 17 12 18.8 45.1 5.0 0 0.626
# 4 2006 5 17 13 20.4 40.8 9.5 0 0.656
# 5 2006 5 17 14 20.1 41.7 8.7 0 0.657
# 6 2006 5 17 15 18.6 45.8 13.5 0 0.629
# (1) gfmc default:
# Re-order the data by year, month, day, and hour:
dat<-test_gfmc[with(test_gfmc,order(yr,mon,day,hr)),]
# Because the test data has 24 hours input variables
# it is possible to calculate the hourly GFMC continuously
# through multiple days(with the default initial GFMCold=85):
dat$gfmc_default<-gfmc(dat)
# two variables will be added to the input, GFMC and MC
head(dat)
# (2) For multiple weather stations:
# One time step (1 hour) with default initial value:
foo<-gfmc(dat,batch=FALSE)
# Chronical hourly GFMC with only one initial
# value (GFMCold=85), but multiple weather stations.
# Note: data is ordered by date/time and the station id. Subset
# the data by keeping only the first 10 hours of observations
# each day:
dat1<-subset(dat,hr%in%c(0:9))
#assuming observations were from the same day but with
#9 different weather stations:
dat1$day<-NULL
dat1$year<with(dat1,order(yr,mon,hr)),]
dat1$id<-rep(1:8,nrow(dat1)/8)
#check the data:
head(dat1)
# Calculate GFMC for multiple stations:
dat0$gfmc01<-gfmc(dat1,batch=TRUE)
# We can provide multiple initial GFMC (GFMCold) as a vector:
dat0$gfmc02<- gfmc(dat1,GFMCold = sample(70:100,8, replace=TRUE),batch=TRUE)
# (3)output argument
## include all inputs and outputs:
dat0<-dat[with(dat,order(yr,mon,day,hr)),]
foo<gfmc(dat0,out="ALL")
## subhourly time step:
gfmc(dat0,time.step=1.5)

hffmc  Hourly Fine Fuel Moisture Code
Description

`hffmc` is used to calculate hourly Fine Fuel Moisture Code (FFMC) and is based on a calculation routine first described in detail by Van Wagner (1977) and which has been updated in minor ways by the Canadian Forest Service to have it agree with the calculation methodology for the daily FFMC (see `fwi`). In its simplest typical use this current routine calculates a value of FFMC based on a series of uninterrupted hourly weather observations of screen level (~1.4 m) temperature, relative humidity, 10 m wind speed, and 1-hour rainfall. This implementation of the function includes an optional time.step input which is defaulted to one hour, but can be reduced if sub-hourly calculation of the code is needed. The FFMC is in essence a bookkeeping system for moisture content and thus it needs to use the last time.step’s value of FFMC in its calculation as well. This function could be used for either one weather station or for multiple weather stations.

Usage

```r
hffmc(weatherstream, ffmc_old=85, time.step=1, calc.step=FALSE, batch=TRUE, hourlyFWI=FALSE)
```

Arguments

- **weatherstream**: A dataframe containing input variables of hourly weather observations. It is important that variable names have to be the same as in the following list, but they are case insensitive. The order in which the input variables are entered is not important.
  - `temp` (required) Temperature (centigrade)
  - `rh` (required) Relative humidity (%)
  - `ws` (required) 10-m height wind speed (km/h)
  - `prec` (required) 1-hour rainfall (mm)
  - `hr` (optional) Hourly value to calculate sub-hourly ffmc
  - `bui` (optional) Daily BUI value for the computation of hourly FWI. It is required when `hourlyFWI=TRUE`.

- **ffmc_old**: Initial FFMC. At the start of calculations at a particular station there is a need to provide an estimate of the FFMC in the previous timestep; this is because the FFMC is, in essence, a bookkeeping system for moisture. If no estimate of previous hour’s FFMC is available the function will use default value, `ffmc_old=85`. When using the routine to calculate hourly FFMC at multiple stations the `ffmc_old` argument can also accept a vector with the same number of weather stations.

- **time.step**: Is the time (in hours) between the previous value of FFMC and the current time at which we want to calculate a new value of the FFMC. When not specified it will take on a default value of `time.step=1`.

- **calc.step**: Optional for whether time step between two observations is calculated. Default is `FALSE`, no calculations. This is used when time intervals are not uniform in the input.
Whether the computation is iterative or single step, default is TRUE. When batch=TRUE, the function will calculate hourly or sub-hourly FFMC for one weather station over a period of time iteratively. If multiple weather stations are processed, an additional "id" column is required in the input weatherstream to label different stations, and the data needs to be sorted by date/time and "id". If batch=FALSE, the function calculates only one time step base on either the previous hourly FFMC or the initial start value.

Optional for the computation of hourly ISI, FWI, and DSR. Default is FALSE. While hourlyFWI=TRUE, daily BUI is required for the computation of FWI.

Details

The hourly FFMC is very similar in its structure and calculation to the Canadian Forest Fire Weather Index System’s daily FFMC (fwi) but has an altered drying and wetting rate which more realistically reflects the drying and wetting of a pine needle litter layer sitting on a decaying organic layer. This particular implementation of the Canadian Forest Fire Danger Rating System’s hourly FFMC provides for a flexible time step; that is, the data need not necessarily be in time increments of one hour. This flexibility has been added for some users who use this method with data sampled more frequently that one hour. We do not recommend using a time step much greater than one hour. An important and implicit assumption in this calculation is that the input weather is constant over the time step of each calculation (e.g., typically over the previous hour). This is a reasonable assumption for an hour; however it can become problematic for longer periods. For brevity we have referred to this routine throughout this description as the hourly FFMC.

Because of the shortened time step, which can lead to more frequent calculations and conversion between moisture content and the code value itself, we have increased the precision of one of the constants in the simple formula that converts litter moisture content to the 'Code' value. This is necessary to avoid a potential bias that gets introduced during extremely dry conditions. This is simply a change in the precision at which this constant is used in the equation and is not a change to the standard FFMC conversion between moisture and code value (which is referred to as the FF-scale).

The calculation requires the previous hour’s FFMC as an input to the calculation of the current hour’s FFMC; this is because the routine can be thought of as a bookkeeping system and needs to know the amount of moisture being held in the fuel prior to any drying or wetting in the current period. After each hour’s calculation that newly calculated FFMC simply becomes the starting FFMC in the next hour’s calculation. At the beginning of the calculations at a station this previous hours FFMC must be estimated. It is typical to use a value of 85 when this value cannot be estimated more accurately; this code value corresponds to a moisture content of about 16% in typical pine litter fuels.

Value

hffmc returns a vector of hourly or sub-hourly FFMC values, which may contain 1 or multiple elements. Optionally when hourlyFWI=TRUE, the function also output a data.frame contains input weatherstream as well as the hourly or sub-hourly FFMC, ISI, FWI, and DSR.

Author(s)

Xianli Wang, Mike Wotton, Alan Cantin, Brett Moore, and Mike Flannigan
References

See Also
fbp, fwi, hffmcRaster

Examples

library(cffdrs)
data("test_hffmc")
# show the data format:
head(test_hffmc)
# (1) hffmc default:
# Re-order the data by year, month, day, and hour:
test_hffmc<-test_hffmc[with(test_hffmc, order(yr,mon,day,hr)),]
# Because the test data has 24 hours input variables
# it is possible to calculate the hourly FFMC chronically
# through multiple days(with the default initial ffmc_old=85):
test_hffmc$ffmc_default<-hffmc(test_hffmc)
# (2) Calculate FFMC for multiple stations:
# Calculate hourly FFMC with only one initial
# value (ffmc_old=85), but multiple weather stations.
# Sort the input by date/time and the station id:
test_hffmc<-test_hffmc[with(test_hffmc,order(yr,mon,hr)),]
# Add weather station id:
test_hffmc$id<-rep(1:10,nrow(test_hffmc)/10)
#check the data:
head(test_hffmc)
test_hffmc$ffmc01<-hffmc(test_hffmc, batch=TRUE)
# With multiple initial FFMC (ffmc_old) as a vector:
test_hffmc$ffmc02<-hffmc(test_hffmc, ffmc_old = sample(70:100,10, replace=TRUE), batch=TRUE)
# One time step assuming all records are from different
# weather stations:
foo<-hffmc(test_hffmc, batch=FALSE)
# (3) output all hourly FWI System variables:
test_hffmc$id<--NULL
test_hffmc<-test_hffmc[with(test_hffmc, order(yr,mon,day,hr)),]
foo<-hffmc(test_hffmc, hourlyFWI=TRUE)
# this will not run: warning message requesting for daily BUI
test_hffmc$bui<--100
foo<-hffmc(test_hffmc, hourlyFWI=TRUE)
# (4) Calculate time steps in case the time intervals are
# not uniform:
dat0<-test_hffmc[sample(1:30,20),]
dat0<-dat0[with(dat0, order(yr,mon,day,hr)),]
# with or without calc.step, hffmc is going to generate
# different FFMC values.
# without calculating time step (default):
Description

`hffmcRaster` is used to calculate hourly Fine Fuel Moisture Code (FFMC) based on hourly weather observations of screen level (~1.4 m) temperature, relative humidity, 10 m wind speed, and 1-hour rainfall. This implementation of the function includes an optional timestep input which is defaulted to one hour, but can be reduced if sub-hourly calculation of the code is needed. The FFMC is in essence a bookkeeping system for moisture content and thus it needs to use the last timestep’s value of FFMC in its calculation. `hffmcRaster` takes rasterized inputs and generates raster maps as outputs.

Usage

```r
hffmcRaster(weatherstream, ffmc_old=85, time.step=1, hourlyFWI=FALSE)
```

Arguments

- `weatherstream`: A stack or brick containing rasterized hourly weather observations. Variable names have to be the same as in the following list, but they are case insensitive. The order in which the input variables are entered is not required.

- `temp`: (required) Temperature (centigrade)

- `rh`: (required) Relative humidity (%)

- `ws`: (required) 10-m height wind speed (km/h)

- `prec`: (required) 1-hour rainfall (mm)

- `bui`: (optional) Daily BUI value for the computation of hourly FWI. It is required when `hourlyFWI=TRUE`.

- `ffmc_old`: A single value of FFMC or a raster of FFMC for the previous hour which will be used for the current hour’s calculation. In some situations, there are no previous-hourly FFMC values to calculate the current hourly FFMC, the function will use a default value, `ffmc_old=84`.

- `time.step`: timestep in hours. Default is 1 hour, set for standard hourly FFMC calculation. While `time.step` is set to values with decimal places, sub-hourly FFMC would be calculated.

- `hourlyFWI`: Optional for the computation of hourly ISI, FWI, and DSR. Default is FALSE. While `hourlyFWI=TRUE`, daily BUI is required for the computation of FWI.
Details

The hourly FFMC is very similar in its structure and calculation to the Canadian Forest Fire Weather Index System’s daily FFMC (fwi) but has an altered drying and wetting rate which more realistically reflects the drying and wetting of a pine needle litter layer sitting on a decaying organic layer. This particular implementation of the Canadian Forest Fire Danger Rating System’s hourly FFMC provides for a flexible timestep; that is, the data need not necessarily be in time increments of one hour. This flexibility has been added for some users who use this method with data sampled more frequently that one hour. We do not recommend using a timestep much greater than one hour. An important and implicit assumption in this calculation is that the input weather is constant over the timestep of each calculation (e.g., typically over the previous hour). This is a reasonable assumption for an hour; however it can become problematic for longer periods. For brevity we have referred to this routine throughout this description as the hourly FFMC.

Because of the shortened timestep, which can lead to more frequent calculations and conversion between moisture content and the code value itself, we have increased the precision of one of the constants in the simple formula that converts litter moisture content to the ‘Code’ value. This is necessary to avoid a potential bias that gets introduced during extremely dry conditions. This is simply a change in the precision at which this constant is used in the equation and is not a change to the standard FFMC conversion between moisture and code value (which is referred to as the FF-scale).

The calculation requires the previous hour’s FFMC as an input to the calculation of the current hour’s FFMC; this is because the routine can be thought of as a bookkeeping system and needs to know the amount of moisture being held in the fuel prior to any drying or wetting in the current period. After each hour’s calculation that newly calculated FFMC simply becomes the starting FFMC in the next hour’s calculation. At the beginning of the calculations at a station this previous hours FFMC must be estimated. It is typical to use a value of 85 when this value cannot be estimated more accurately; this code value corresponds to a moisture content of about 16% in typical pine litter fuels.

Value

hffmcRaster returns a vector of hourly or sub-hourly FFMC values, which may contain 1 or multiple elements. Optionally when hourlyFWI=TRUE, the function also output a data.frame contains input weatherstream as well as the hourly or sub-hourly FFMC, ISI, FWI, and DSR.

Author(s)

Xianli Wang, Mike Wotton, Alan Cantin, Brett Moore, and Mike Flannigan

References


See Also

fbp, fwi, hffmc
Examples

```r
library(cffdrs)
require(raster)
## load the test data for the first hour, namely hour01:
hour01src <- system.file("extdata","test_rast_hour01.tif",package="cffdrs")
hour01 <- stack(hour01src)
# Assign names to the layers:
names(hour01)<-c("temp","rh","ws","prec")
# (1) Default, based on the initial value:
foo<-hffmcRaster(hour01)
plot(foo)
### Additional, longer running examples ###
# (2) Based on previous day's hffmc:
# load the test data for the second hour, namely hour02:
hour02src <- system.file("extdata","test_rast_hour02.tif",package="cffdrs")
hour02 <- stack(hour02src)
# Assign variable names to the layers:
names(hour02)<-c("temp","rh","ws","prec")
foo1<-hffmcRaster(hour02,hffmc_old=foo)
plot(foo1)
# (3) Calculate other hourly FWI components (ISI, FWI, and DSR):
# Need BUI layer,
bui<-hour02$temp
values(bui)<-50
hour02<-stack(hour02,bui)
# Re-assign variable names to the layers:
names(hour02)<-c("temp","rh","ws","prec","bui")
# Calculate all the variables:
foo2<-hffmcRaster(hour02,hffmc_old=foo,hourlyFWI=TRUE)
# Visualize the maps:
plot(foo2)
```

Iros

Line-based input for Simard Rate of Spread and Direction

Description

Iros is used to calculate the rate of spread and direction given one set of three point-based observations of fire arrival time. The function requires that the user specify the time that the fire crossed each point, along with the measured lengths between each pair of observational points, and a reference bearing (one specified side of the triangle). This function allows quick input of a dataframe specifying one or many triangles.

Usage

```
Iros(input)
```
Arguments

input A dataframe containing input variables of time fire front crossed points 1, 2, 3, and latitude/longitude for those same points. Variable names have to be the same as in the following list, but they are case insensitive. The order in which the input variables are entered is not important.

\( T1 \) (required) Time that the fire front crossed point 1. Time entered in fractional format. Output ROS will depend on the level of precision entered (minute, second, decisecond)

\( T2 \) (required) Time that the fire front crossed point 2. Time entered in fractional format. Output ROS will depend on the level of precision entered (minute, second, decisecond)

\( T3 \) (required) Time that the fire front crossed point 3. Time entered in fractional format. Output ROS will depend on the level of precision entered (minute, second, decisecond)

\( \text{Length}T1T2 \) (required) Length between each pair of observation points T1 and T2 (subscripts denote time-ordered pairs). (meters)

\( \text{Length}T2T3 \) (required) Length between each pair of observation points T2 and T3 (subscripts denote time-ordered pairs). (meters)

\( \text{Length}T1T3 \) (required) Length between each pair of observation points T1 and T3 (subscripts denote time-ordered pairs). (meters)

\( \text{Bearing}T1T2 \) (required) Reference bearing. For reference, North = 0, West = -90, East = 90 (degrees)

\( \text{Bearing}T1T3 \) (required) Reference bearing. For reference, North = 0, West = -90, East = 90 (degrees)

Details

\text{Iros} Allows R users to calculate the rate of spread and direction of a fire across a triangle, given three time measurements and details about the orientation and distance between observational points. The algorithm is based on the description from Simard et al. (1984). See \text{pros} for more information.

The functions require the user to arrange the input dataframe so that each triangle of interest is identified based on a new row in the dataframe. The input format forces the user to identify the triangles, one triangle per row of input dataframe. Very complex arrangements of field plot layouts are possible, and the current version of these functions do not attempt to determine each triangle of interest automatically.

Value

\text{Iros} returns a dataframe which includes the rate of spread and spread direction. Output units depend on the user’s inputs for distance (typically meters) and time (seconds or minutes).

Author(s)

Tom Schiks, Xianli Wang, Alan Cantin
References


See Also

pros

Examples

library(cffdrs)
# manual single entry, but converted to a data frame
lros.in1 <- data.frame(t(c(0, 24.5, 50, 22.6, 120, 20.0, 90, 35)))
colnames(lros.in1)<-c("T1","LengthT1T2", "T2", "LengthT1T3", "T3",
"LengthT2T3", "bearingT1T2", "bearingT1T3")
lros.out1 <- lros(lros.in1)
lros.out1

# multiple entries using a dataframe
# load the test dataframe for lros
data("test_lros")
lros(test_lros)

---

pros Point-based input for Simard Rate of Spread and Direction

Description

pros is used to calculate the rate of spread and direction given one set of three point-based observations of fire arrival time. The function requires that the user specify the time that the fire crossed each point, along with the latitude and longitude of each observational point. This function allows quick input of a dataframe specifying one or many triangles.

Usage

pros(input)
Arguments

input

A dataframe containing input variables of Time fire front crossed points 1, 2, 3, and latitude/longitude for those same points. Variable names have to be the same as in the following list, but they are case insensitive. The order in which the input variables are entered is not important.

T1 (required) Time that the fire front crossed point 1. Time entered in fractional format. Output ROS will depend on the level of precision entered (minute, second, decisecond)

T2 (required) Time that the fire front crossed point 2. Time entered in fractional format. Output ROS will depend on the level of precision entered (minute, second, decisecond)

T3 (required) Time that the fire front crossed point 3. Time entered in fractional format. Output ROS will depend on the level of precision entered (minute, second, decisecond)

Long1 (required) Longitude for dataloggcr 1. (decimal degrees).

Long2 (required) Longitude for dataloggcr 2. (decimal degrees).

Long3 (required) Longitude for dataloggcr 3. (decimal degrees).

Lat1 (required) Latitude for dataloggcr 1. (decimal degrees).

Lat2 (required) Latitude for dataloggcr 2. (decimal degrees).

Lat3 (required) Latitude for dataloggcr 3. (decimal degrees).

Details

pros allows R users to calculate the rate of spread and direction of a fire across a triangle, given three time measurements and details about the orientation and distance between observational points. The algorithm is based on the description from Simard et al. (1984).

Rate of spread and direction of spread are primary variables of interest when observing wildfire growth over time. Observations might be recorded during normal fire management operations (e.g., by a Fire Behaviour Analyst), during prescribed fire treatments, and during experimental research burns. Rate of spread is especially important for estimating Byram’s fireline intensity, fireline intensity = heat constant of fuel × weight of fuel consumed × forward rate of spread (Byram 1959).

Rate of spread is difficult to measure and highly variable in the field. Many techniques were proposed over the years, but most were based on observations collected from a pre-placed reference grid and stopwatch (Curry and Fons 1938; Simard et al. 1982). Early approaches required that observers be in visual contact with the reference grid, but later, thermocouples and dataloggers were employed to measure the onset of the heat pulse at each point.

Simard et al. (1982) proposed calculations for spread based on an equilateral triangle layout. Simard et al. (1984) proposed calculations for spread based on any type of triangle. Both articles also discussed field sampling design and layout, with special attention to the size of the triangles (large enough that the fire traverses the triangle in one to two minutes) and even using triangles of varying size within one field plot (but no triangle larger than one fourth of the site’s total area).

The underlying algorithms use trigonometry to solve for rate of spread and direction of spread. One important assumption is that the spread rate and direction is uniform across one triangular plot, and that the fire front is spreading as a straight line; Simard et al. (1982, 1984) acknowledge that these assumption are likely broken to some degree during fire spread events.
The functions require the user to arrange the input dataframe so that each triangle of interest is identified based on a new row in the dataframe. The input format forces the user to identify the triangles, one triangle per row of input dataframe. Very complex arrangements of field plot layouts are possible, and the current version of these functions do not attempt to determine each triangle of interest automatically.

Value

`pros` returns a dataframe which includes the rate of spread and spread direction. Output units depend on the user’s inputs for distance (typically meters) and time (seconds or minutes).

Author(s)

Tom Schiks, Xianli Wang, Alan Cantin

References


See Also

`lros`.

Examples

```r
library(cffdrs)
# manual single entry
pros.in1 <- data.frame(t(c(2, -79.701027, 43.808872, 50, -79.699650, 43.808833, 120, -79.700387, 43.809816)))
colnames(pros.in1)<-c("T1", "LONG1", "LAT1", "T2", "LONG2", "LAT2", "T3", "LONG3", "LAT3")
pros.out1 <- pros(pros.in1)
# multiple entries using a dataframe
# load the test dataframe for pros
data("test_pros")
pros(test_pros)
```
**Description**

`sdmc` is used to calculate sheltered DMC (sDMC, Wotton et al., 2005) based on daily noon weather observations of temperature, relative humidity, wind speed, 24-hour rainfall, and a previous day’s calculated or estimated value of sDMC. This function calculates sDMC for either one weather station or for multiple weather stations over the duration of the daily weather data set, typically over a fire season.

**Usage**

```r
sdmc(input, sdmc_old=NULL, batch=TRUE)
```

**Arguments**

- `input` A data.frame containing input variables of daily noon weather observations. Variable names have to be the same as in the following list, but they are case insensitive. The order in which the input variables are entered is not important either.
  - `temp` (required) Temperature (centigrade)
  - `rh` (required) Relative humidity (%)
  - `ws` (required) 10-m height wind speed (km/h)
  - `prec` (required) 1-hour rainfall (mm)
  - `mon` (recommended) Month of the observations (integer 1-12)
  - `day` (optional) Day of the observations (integer)

- `sdmc_old` Previous day’s value of SDMC. At the start of calculations, when there is no calculated previous day’s SDMC value to use, the user must specify an estimate of this value. Where `sdmc_old=NULL`, the function will calculate the initial SDMC values based on the initial DMC. The `sdmc_old` argument can accept a single initial value for multiple weather stations, and also accept a vector of initial values for multiple weather stations.

- `batch` Whether the computation is iterative or single step, default is TRUE. When `batch=TRUE`, the function will calculate daily SDMC for one weather station over a period of time iteratively. If multiple weather stations are processed, an additional "id" column is required in the input to label different stations, and the data needs to be sorted by date/time and "id". If `batch=FALSE`, the function calculates only one time step base on either the previous day’s SDMC or the initial start value.
**Details**

The Duff Moisture Code (DMC) component of the Canadian Forest Fire Weather Index (FWI) System tracks moisture content of the forest floor away from the sheltering influences of overstory trees. This sheltered Duff Moisture Code (sDMC) was developed to track moisture in the upper 5 cm of the organic layer in the rain sheltered areas near (<0.5 m) the boles of overstory trees (Wotton et al. 2005), an area where lightning strikes usually ignite the forest floor when they run to ground. The sDMC is very similar in structure (and identical in data requirements) to the DMC. The sDMC, like all the FWI System moisture codes, is a bookkeeping system that tracks gain and loss of moisture from day-to-day; thus an estimate of the previous day’s sDMC value is needed to provide a starting point for each day’s moisture calculation. Like the other moisture codes in the FWI System the sDMC is converted from a moisture content value to an outputted CODE value which increases in value with decreasing moisture content.

**Value**

sdmc returns either a single value or a vector of SDMC values.

**Author(s)**

Xianli Wang, Mike Wotton, Alan Cantin, and Mike Flannigan

**References**


**See Also**

fwi

**Examples**

```r
library(cffdhrs)
data("test_sdmc")
#order the data:
test_sdmc<-test_sdmc[with(test_sdmc,order(yr,mon,day)),]
  # Default of sdmc, calculate sdmc for a chronical period
  # of time.
  # Because sdmnc_old is better to be calculated, we normally
  # ignore this option:
test_sdmc$sDMC<-sdmc(test_sdmc)
  # (2) multiple weather stations:
  # Batch process with multiple stations (2 stations) assuming
  # they are from the same month:
test_sdmc$mon<-7
test_sdmc$day<-rep(1:24,2)
test_sdmc$id<-rep(1:2,each=24)
  # Sort the data by date and weather station id:
test_sdmc<-test_sdmc[with(test_sdmc,order(yr,mon,day,id)),]
```
# Apply the function
test_sdmc$SDMC_mult_stn <- sdmc(test_sdmc, batch=TRUE)
# Assuming each record is from a different weather station, and
# calculate only one time step:
foo <- sdmc(test_sdmc, batch=FALSE)

---

test_fbp Fire Behaviour Prediction Sample Data Set

**Description**

This data set is a set of input data for each of the test cases in the publication supplied below.

**Usage**

data(test_fbp)

**Format**

A data frame containing 24 columns, 21 rows, including 1 header line

**Source**


**References**


---

test_fwi Fire Weather Index Sample Input Data Set

**Description**

This data set is the sample input data that was used in original FWI program calibration.

**Usage**

data(test_fwi)

**Format**

A data frame containing 9 columns and 49 rows, with 1 header line
Source


References


---

**test_gfmc**

Grass Fuel Moisture Code Sample Input Data Set

Description

This data set is the sample input data that was used in original FWI program calibration.

Usage

data(test_gfmc)

Format

A data frame containing 9 columns and 199 rows, with 1 header line

---

**test_hffmc**

Hourly Fine Fuel Moisture Code Sample Input Data Set

Description

Sample dataset for use with the hffmc function.

Usage

data(test_hffmc)

Format

A data frame containing 8 columns and 481 rows, including 1 header line
**test_lros**  
*Line-based Simard function Sample Data Set*

**Description**

This is a set of input data to test the lros function.

**Usage**

```r
data(test_lros)
```

**Format**

A data frame containing 8 columns, 4 rows, including 1 header line.

**Source**

no source

**References**


---

**test_pros**  
*Point-based Simard function Sample Data Set*

**Description**

This is a set of input data to test the pros function.

**Usage**

```r
data(test_pros)
```

**Format**

A data frame containing 9 columns, 4 rows, including 1 header line.
Source

no source

References


description

This data set is the sample input data that was used in original FWI program calibration, but with an initial dmc value populated.

Usage

data(test_fwi)

Format

A data frame containing 10 columns and 49 rows, including 1 header line

Source


References

overwintered DC

This dataset has 2 ID values (weather stations), and each have 2 sequential years. This data can be used as an example to calculate overwintered DC. There are 10 columns and 1463 rows, including 1 header row.

Usage

```r
data(test_wDC)
```

Format

A data frame containing 10 columns and 1463 rows, including 1 header line

---

fire season dataset

This dataset has pre-set start and end dates to the fire season for 2 weather stations. The point of this dataset is to demonstrate that a data frame of start and end dates for the fire season can be calculated and applied to the program.

Usage

```r
data(test_wDC_fs)
```

Format

A data frame containing 7 columns and 9 rows, including 1 header line
**Description**

wDC calculates an initial or season starting Drought Code (DC) value based on a standard method of overwintering the Drought Code (Lawson and Armitage 2008). This method uses the final DC value from previous year, over winter precipitation and estimates of how much over-winter precipitation 'refills' the moisture in this fuel layer. This function could be used for either one weather station or for multiple weather stations.

**Usage**

\[ wDC(DC_f=100, rw=200, a=0.75, b=0.75) \]

**Arguments**

- **DC\(_f\)**: Final fall DC value from previous year
- **rw**: Winter precipitation (mm)
- **a**: User selected values accounting for carry-over fraction (view table below)
- **b**: User selected values accountain for wetting efficiency fraction (view table below)


<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
<th>Criteria</th>
</tr>
</thead>
<tbody>
<tr>
<td>Carry-over fraction of last fall’s moisture (a)</td>
<td>1.0</td>
<td>Daily DC calculated up to 1 November</td>
</tr>
<tr>
<td></td>
<td>0.75</td>
<td>Daily DC calculations stopped before any of the above conditions met or the area is subject to occasional winter chinook conditions, leaving the ground bare and subject to moisture depletion</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>Forested areas subject to long periods in fall or winter that favor depletion of soil moisture</td>
</tr>
<tr>
<td>Effectiveness of winter precipitation in recharging moisture reserves in spring (b)</td>
<td>0.9</td>
<td>Poorly drained, boggy sites with deep organic layers</td>
</tr>
<tr>
<td></td>
<td>0.75</td>
<td>Deep ground frost does not occur until late fall, if at all; moderately drained sites that allow infiltration of most of the melting snowpack</td>
</tr>
<tr>
<td></td>
<td>0.5</td>
<td>Chinook-prone areas and areas subject to early and deep ground frost; well-drained soils favoring rapid percolation or topography favoring rapid runoff before melting of ground frost</td>
</tr>
</tbody>
</table>

**Details**

Of the three fuel moisture codes (i.e. FFMC, DMC and DC) making up the FWI System, only the DC needs to be considered in terms of its values carrying over from one fire season to the next. In Canada both the FFMC and the DMC are assumed to reach moisture saturation from overwinter precipitation at or before spring melt; this is a reasonable assumption and any error in these assumed starting conditions quickly disappears. If snowfall (or other overwinter precipitation) is not large enough however, the fuel layer tracked by the Drought Code may not fully reach saturation after spring snow melt; because of the long response time in this fuel layer (53 days in standard conditions) a large error in this spring starting condition can affect the DC for a significant portion of the fire season. In areas where overwinter precipitation is 200 mm or more, full moisture recharge occurs and DC overwintering is usually unnecessary. More discussion of overwintering and fuel drying time lag can be found in Lawson and Armitage (2008) and Van Wagner (1985).
Value

wDC returns either a single value or a vector of wDC values.

Author(s)

Xianli Wang, Mike Wotton, Alan Cantin, and Mike Flannigan

References


See Also

fwi, fireSeason

Examples

library(cffdrs)
# The standard test data:
data("test_wDC")
# (1) Simple case previous fall's DC was 300, overwinter
# rain 110mm
winter_DC <- wDC(DCf=300,rw=110)
winter_DC
#(2) modified a and b parameters. Find table values in listed
# reference for Lawson and Armitage, 2008.
winter_DC <- wDC(DCf=300,rw=110,a=1.0,b=0.9)
winter_DC
#(3) with multiple inputs:
winter_DC <- wDC(DCf=c(400,300,250), rw=c(99,110,200),
a=c(0.75,1.0,0.75), b=c(0.75,0.9,0.75))
winter_DC
#(4) A realistic example:
# precipitation accumulation and date boundaries
input <- test_wDC
# order data by ID and date
input <- with(input,input[order(id,yr,mon,day),])
input$date <- as.Date(as.POSIXlt(paste(input$yr,"-",input$mon,"-",input$day,sep="")))
# select id value 1
input.2 <- input[input$id==2,]
# Explicitly defined fire start and end dates.
data("test_wDC_fs")
print(test_wDC_fs)
# Set date field
test_wDC_fs$date <- as.Date(as.POSIXlt(paste(test_wDC_fs$yr,"-",test_wDC_fs$mon,"-",
test_wDC_fs$day,sep="")))
#match to current id value
input.2.fs <- test_wDC.fs[test_wDC.fs$id==2,]

#assign start of winter date (or end of fire season date)
winterStartDate <- input.2.fs[2,"date"]

#assign end of winter date (or start of new fire season date)
winterEndDate <- input.2.fs[3,"date"]

#Accumulate overwinter precip based on chosen dates
curYr.prec <- sum(input.2[(input.2$date>=winterStartDate & input.2$date<winterEndDate),]$prec)

#Assign a fall DC value
fallDC <- 500

#calculate winter DC
winter_DC <- wDC(DCf=fallDC,rw=curYr.prec)

winter_DC

#Assign a different fall DC value
fallDC <- 250

#calculate winter DC
winter_DC <- wDC(DCf=fallDC,rw=curYr.prec,a=1.0)

winter_DC
Index

*Topic **datasets**
  test_fbp, 42
  test_fwi, 42
  test_gfmc, 43
  test_hffmc, 43
  test_lros, 44
  test_pros, 44
  test_sdmc, 45
  test_wDC, 46
  test_wDC_fs, 46

*Topic **lros**
  test_lros, 44
  test_pros, 44

*Topic **methods**
  fbp, 5
    fbpRaster, 11
  fireSeason, 17
  fwi, 20
  fwiRaster, 24
  gfmc, 27
  hffmc, 29
  hffmcRaster, 33
  sdmc, 40
  wDC, 47

*Topic **package**
  cffdrs-package, 2

*Topic **ros**
  lros, 35
  pros, 37

*Topic **simard**
  test_lros, 44
  test_pros, 44

  cffdrs (cffdrs-package), 2
  cffdrs-package, 2

  fbp, 3, 4, 5, 16, 22, 26, 32, 34
  fbpRaster, 3, 10, 11, 26
  fireSeason, 3, 4, 17, 17, 18, 22, 48

  fwi, 3, 4, 6, 10, 18, 19, 20, 26–28, 30–32, 34, 41, 48
  fwiRaster, 3, 4, 11, 16, 22, 24
  gfmc, 3, 4, 22, 27
  hffmc, 3, 4, 22, 26–28, 29, 34
  hffmcRaster, 3, 4, 16, 22, 26, 32, 33
  lros, 3, 4, 35, 39
  pros, 3, 4, 37, 37
  sdmc, 3, 4, 22, 40
  test_fbp, 42
  test_fwi, 42
  test_gfmc, 43
  test_hffmc, 43
  test_lros, 44
  test_pros, 44
  test_sdmc, 45
  test_wDC, 46
  test_wDC_fs, 46

  wDC, 3, 4, 18, 19, 22, 47