Package ‘ffcAPIClient’

January 14, 2020

Type Package

Title Functional Flows Calculator API Client

Version 0.9.3

Author Nick Santos, Ryan Peek

Maintainer Nick Santos <nrsantos@ucdavis.edu>

Description A client for the Python-based functional flows calculator API hosted at eflows.ucdavis.edu. Requires a token from the eflows.ucdavis.edu website to operate. More information forthcoming. See README at https://github.com/ceff-tech/ffc_api_client

Use four spaces when indenting paragraphs within the Description.

License MIT

Encoding UTF-8

LazyData true

Imports dplyr,
jsonlite,
httr,
uuid,
ggplot2,
dataRetrieval,
lubridate,
R6,
nhdplusTools,
units,
tidyr,
data.table (>= 1.12.8)

Suggests testthat (>= 2.1.0)

RoxygenNote 7.0.2

R topics documented:

`ffcAPIClient` *ffcAPIClient: Processes time-series flow data using the online functional flows calculator*

Description

For now, see the documentation for `get_ffc_results_for_df`

Examples

```
## Not run:
# Example 1
## Initialize a Run
test_data <- example_gagedata()  # just get some fake gage data - based on Daniel Philippus' code - you can build
ffcAPIClient::set_token(YOUR_TOKEN_VALUE_IN_QUOTES)  # you'll need to get your own of this - see above
results <- ffcAPIClient::get_ffc_results_for_df(test_data)  # send it to the FFC online to process

## Retrieve Results and Plot
## get the DRH data as a data frame with percentiles for columns and days for rows
drh_data <- ffcAPIClient::get_drh(results)
plot(drh_data$seventy_five, type="l")  # plot the seventy-fifth percentile DRH

# Example 2: Retrieve, Process, Plot USGS gage
## This example retrieves USGS gage data, runs it through the FFC online, and plots the DRH nicely
## don't forget to set your token first if you haven't already
ffcAPIClient::set_token(YOUR_TOKEN_VALUE_IN_QUOTES) # you'll need to get your own of this - see above

## retrieves flow data for North Fork American gage and sends it through the FFC
results <- ffcAPIClient::get_ffc_results_for_usgs_gage(11427000)
drh_plot <- ffcAPIClient::plot_drh(results)  # includes optional output_path argument to save to file automatic
drh_plot  # display the plot

## End(Not run)
```

`FFCProcessor` *FFCProcessor Class*

Description

The new workhorse of the client - this class is meant to bring together the scattershot functions in other parts of the package so that data can be integrated into a single class with a single set of tasks. Other functions are likely to be supported for a while (and this may even rely on them), but long run, much of the code in this file might move into this class, with the shortcut functions creating this class behind the scenes and returning an instance of this object.

Details

More details to come, and more examples.

Methods

Public methods:

- `FFCProcessor$get_ffc_results()`
- `FFCProcessor$evaluate_alteration()`
- `FFCProcessor$clone()`

Method `get_ffc_results()`**:**

Usage:

```
FFCProcessor$get_ffc_results()
```

Method `evaluate_alteration()`**:**

Usage:

```
FFCProcessor$evaluate_alteration()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
FFCProcessor$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

`flow_metrics` *Modeled flow metric predictions for all stream segments*

Description

Contains the 10th, 25th, 50th, 75th, and 90th percentile values for each flow metric and stream segment combination. It is a data frame where the metrics are rows with names in the `Metric` field, stream segment ID is in the COMID field and percentiles are available as fields such as `pct_10`, `pct_25`, etc for each percentile.

Usage

```
flow_metrics
```

Format

A data frame :

name text

name text ...

https://github.com/ceff-tech/

`get_comid_for_lon_lat` *Retrieves COMID for a given USGS gage which collects daily data.*

Description

This function returns the COMID associated with a specific USGS gage. It can be used to associate gage data with flow metric predictions a stream segment identified with the `com_id` input variable.

Usage

```
get_comid_for_lon_lat(longitude, latitude)
```

Arguments

`longitude`	numeric. Longitude or X.
`latitude`	numeric. Longitude or Y.

`get_comid_for_usgs_gage` *Retrieves COMID for a given USGS gage which collects daily data.*

Description

This function returns the COMID associated with a specific USGS gage. It can be used to associate gage data with flow metric predictions a stream segment identified with the `com_id` input variable.

Usage

```
get_comid_for_usgs_gage(gage_id)
```

Arguments

`gage_id`	character. A character formatted 8 digit USGS Gage ID.

get_drh *Returns the dimensionless reference hydrograph results as a data frame*

Description

Returns the dimensionless reference hydrograph results as a data frame

Usage

```
get_drh(results)
```

get_ffc_results_for_df *Run Data Frame Through Functional Flows Calculator*

Description

This is the primary function to use from the API client itself to obtain raw FFC results. It will generate a unique ID, run the data frame through the FFC, and then delete the results for that ID from the website so as not to clutter up the user's account, or store too much data on the server side.

Usage

```
get_ffc_results_for_df(flows_df, flow_field, date_field, start_date)
```

Arguments

`flows_df`	DataFrame. A time series data frame with flow and date columns
`flow_field`	character, default "flow". The name of the field in `df` that contains flow values.
`date_field`	character, default "date". The name of the field in `df` that contains date values for each flow. The date field must be in MM/DD/YYYY format as either factor or character values - true dates likely will not work based on the API we're using. If you need to convert date values, add a field to your existing data frame with the values in MM/DD/YYYY format before providing it to this function.
`start_date`	character, default "10/1". What month and day should the water year start on? Neither month nor day needs to be zero-padded here, so March first could just be 3/1, while December 12th can be 12/12.

Value

list of results from the functional flows calculator. More information will be forthcoming as we inspect the structure of what is returned.

`get_ffc_results_for_usgs_gage` *Run Gage Data Through the Functional Flows Calculator*

Description

Provided with an integer Gage ID, this function pulls the timeseries data for the gage and processes it in a single step. Returns the functional flow calculator's results list.

Usage

```
get_ffc_results_for_usgs_gage(gage_id, start_date)
```

Arguments

`gage_id`	integer. The USGS Gage ID value for the gage you want to return timeseries data for

Value

list. Functional Flow Calculator results

`get_predicted_flow_metrics` *Retrieves flow predicted flow metric values for a stream segment*

Description

This function returns the 10th, 25th, 50th, 75th, and 90th percentile values for each flow metric as predicted for the stream segment you identify with the `com_id` input variable. It returns a data frame where the metrics are rows with names in the `metric` field, and percentiles are available as fields such as `pct_10`, `pct_25`, etc for each percentile.

Usage

```
get_predicted_flow_metrics(com_id)
```

Arguments

`com_id`	character. A string of a NHD COMID to retrieve metrics for.

`get_results_as_df` *Convert FFC results list to data frame with metric names*

Description

More documentation forthcoming

Usage

```
get_results_as_df(results, drop_fields)
```

`get_results_for_name` *Retrieve processed results from FFC.*

Description

In most cases, you won't need to use this function! If you're wondering what to do, use get_ffc_results_for_df instead.

Usage

```
get_results_for_name(name, autodelete)
```

Details

Gets the results for the given named run of the FFC. Returns the nested list - no other processing

`get_token` *Retrieve Previously Set Token*

Description

Retrieves the authorization token previously set by set_token in the same R session.

Usage

```
get_token()
```

`get_usgs_gage_data` *Retrieves USGS timeseries gage data*

Description

This is just a helper function that calls the gage constructor, gets the flows and returns them in one step. Useful in situations where we don't need the flexibility of the USGSGage class

Usage

```
get_usgs_gage_data(gage_id)
```

Arguments

`gage_id`	integer. The USGS Gage ID value for the gage you want to return timeseries data for

Value

dataframe. Will include a flow field (CFS) and a date field (MM/DD/YYYY)

`make_flow_json` *Makes the part of the JSON string that is just for the flow data - needs to be passed into make_json later as "data_json"*

Description

Makes the part of the JSON string that is just for the flow data - needs to be passed into make_json later as "data_json"

Usage

```
make_flow_json(flows_df, flow_field, date_field)
```

`make_json` *Prepares the JSON payload to send to the eflows website.*

Description

Prepares the JSON payload to send to the eflows website.

Usage

```
make_json(data_json, start_date, token, extra)
```

`merge_list` *Merges Data Frames by Year Column*

Description

Just a simple function that can be used with Reduce to merge multiple data frames together by year

Usage

```
merge_list(df1, df2)
```

`plot_drh` *Plots the Dimensionless Reference Hydrograph*

Description

Given a set of results data from get_ffc_results_for_df or get_ffc_results_for_usgs_gage, processes the DRH data and returns a plot object.

Usage

```
plot_drh(results, output_path)
```

Arguments

`results`	list.
`output_path,`	default NULL. Optional. When set, saves the DRH plot to the output file path provided.

Details

Credit to Ryan Peek for the code in this function.

`process_data` *Send flow data for processing*

Description

In most cases, you won't need to use this function! If you're wondering what to do, use get_ffc_results_for_df instead.

Usage

```
process_data(flows_df, flow_field, date_field, start_date, name)
```

Details

Sends flow timeseries data off to the functional flows calculator. Does not retrieve results!

`set_token` *Set Eflows Website Access Token*

Description

Provide the token string used for accessing the Eflows site. A token is a method of authorization for identifying your user account within scripts. By providing the token, this package uses your user account when interacting with the eflows web service/API.

Usage

```
set_token(token_string)
```

Arguments

`token_string`	character

`USGSGage` *USGS Gage Retrieval Tools*

Description

This class retrieves data for a USGS gage.

Details

#library(ffcAPIClient) #gageid <- 11427000 #gage <- USGSGage$new() #gage$id <- gageid #gage$get_data() #gage$get_comid() #gage$comid [1] 14996611 #ffcAPIClient::get_predicted_flow_metrics(gage$comid) Metric COMID p10 p25 p50 p75 p90 source 70804 DS_Dur_WS 14996611 1.051875e+02 1.273438e+02 154.0625 1.785563e+02 1.953908e+02 model 211050 DS_Mag_50 14996611 4.998793e+01 6.732828e+01 104.4028 1.464183e+02 1.882733e+02 model 351296 DS_Mag_90 14996611 9.314097e+01 1.291930e+02 173.6844 2.382053e+02 3.393799e+02 model 491542 DS_Tim 14996611 2.720000e+02 2.823875e+02 296.8875 3.070000e+02 3.210167e+02 model 586665 FA_Dur 14996611 2.000000e+00 3.000000e+00 4.0000 6.000000e+00 8.000000e+00 obs 702508 FA_Mag 14996611 1.294269e+02 1.886283e+02 289.6838 4.540329e+02 8.514823e+02 model 842754 FA_Tim 14996611 7.816667e+00 1.400000e+01 24.6250 2.900000e+01 4.217000e+01 model 983000 Peak_10 14996611 1.243107e+04 1.947545e+04 22830.3355 3.124928e+04 3.767889e+04 model 1123246 Peak_20 14996611 8.078893e+03 1.227363e+04 20218.4829 2.087196e+04 2.087196e+04 model 1263492 Peak_50 14996611 3.532988e+03 7.350986e+03 8542.1191 8.969386e+03 8.969386e+03 model 1358615 Peak_Dur_10 14996611 1.000000e+00 1.000000e+00 1.0000 2.000000e+00 4.000000e+00 obs 1429335 Peak_Dur_20 14996611 1.000000e+00 1.000000e+00 2.0000 3.000000e+00 6.000000e+00 obs 1500055 Peak_Dur_50 14996611 1.000000e+00 1.000000e+00 4.0000 1.000000e+01 2.900000e+01 obs 1570775 Peak_Fre_10 14996611 1.000000e+00 1.000000e+00 1.0000 1.000000e+00 2.000000e+00 obs 1641495 Peak_Fre_20 14996611 1.000000e+00 1.000000e+00 1.0000 2.000000e+00 3.000000e+00 obs 1712215 Peak_Fre_50 14996611 1.000000e+00 1.000000e+00 2.0000 3.000000e+00 5.000000e+00 obs 1828058 SP_Dur 14996611 4.700000e+01 5.900000e+01 72.0000 9.527500e+01 1.215417e+02 model 1968304 SP_Mag 14996611 1.067727e+03 1.662598e+03 2489.0563 3.771512e+03 5.809320e+03 model 2063427 SP_ROC 14996611 3.845705e-02 4.863343e-02 0.0625 8.132020e-02 1.141117e-01 obs 2179270 SP_Tim 14996611 1.607717e+02 1.905000e+02 218.7500 2.354750e+02 2.447583e+02 model 2319516 Wet_BFL_Dur 14996611

7.633333e+01 1.073000e+02 141.1958 1.633750e+02 1.875000e+02 model 2459762 Wet_BFL_Mag_10 14996611 1.519943e+02 1.960031e+02 278.2581 4.384614e+02 5.489183e+02 model 2600008 Wet_BFL_Mag_50 14996611 4.148992e+02 5.902507e+02 924.1728 1.175461e+03 1.426576e+03 model 2740254 Wet_Tim 14996611 4.937500e+01 5.905000e+01 73.0000 8.835625e+01 1.035083e+02 model

Methods

Public methods:

- `USGSGage$validate()`
- `USGSGage$get_data()`
- `USGSGage$get_comid()`
- `USGSGage$get_predicted_metrics()`
- `USGSGage$clone()`

Method `validate()`**:**

Usage:

```
USGSGage$validate(latlong)
```

Method `get_data()`**:**

Usage:

```
USGSGage$get_data()
```

Method `get_comid()`**:**

Usage:

```
USGSGage$get_comid()
```

Method `get_predicted_metrics()`**:**

Usage:

```
USGSGage$get_predicted_metrics()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
USGSGage$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Index