timeseries documentation

# Welcome to timeseries’s documentation!

## Projekt

### chart module

*class* chart.Chart(*df: pandas.core.frame.DataFrame*, *title: str*, *id: int*)

Bases: object

Class contains loaded time series data in form of charts.

df*: pandas.core.frame.DataFrame = None*

Data frame of loaded signal.

displayChart(*display: tkinter.ttk.Frame*, *moduleDisplayer*)

Displays chart on screen.

Parameters

* **display** – GUI component where chart should be displayed.
* **moduleDisplayer** – Method called in-line, that applies patches on chart from all active time series modules.

exportChart(*filename: str*)

Exports chart as .png file to specified destination.

Parameters

**filename** – Directory where canvas should be saved.

id*: int = 0*

ID of the chart.

title*: str = ''*

Title of the chart.

### entryvalidators module

entryvalidators.validate\_digit(*val*)

Checks if values in text entries are positive integer value.

Parameters

**val** – Value to check.

Returns

If val is positive int. Rollbacks operation if false.

entryvalidators.validate\_floatentries(*val*)

Checks if values in text entries are positive float value.

Parameters

**val** – Value to check.

Returns

If val is positive float. Rollbacks operation if false.

### gui module

*class* gui.AnalyzerGui

Bases: [gui.Gui](#gui.Gui)

Class contains window application of time series analyzer.

buildSectionChart()

Builds section with chart that displays current data.

buildSectionOutput()

Builds section with output functions that display and export data.

buildSectionSettings()

Builds section with settings that set properties of data.

buildSections()

Constructs all sections of an application.

buildSubsectionDataSelection()

Builds subsection of settings section that allows user to choose displayed data, which was previously loaded.

buildSubsectionExportData()

Builds subsection of output section that allows exporting logs, chart and statistics.

buildSubsectionLoadData()

Builds subsection of settings section that loads of data from memory.

buildSubsectionLogger()

Builds subsection of output section that logs user’s actions.

buildSubsectionStatisctics()

Builds subsection of output section that displays data of modules in tabular form.

buildSubsectionTSModules()

Builds subsection of settings section that allows user to choose displayed time series analyzing modules.

clear\_sectionChart()

Method clears the view of section with chart.

deleteChart()

Deletes the current viewed chart.

statisticstabRefresh()

Removes the content of combobox and table view with all active time series modules.

updateChart()

Updates the view of section with chart.

updateChartsCb()

Refreshes the content of combobox with all loaded time series modules.

*class* gui.GeneratorGui

Bases: [gui.Gui](#gui.Gui)

buildSectionChart()

Builds section with chart that displays current generated data.

buildSectionSettings()

Builds section with settings that set properties of generated data.

buildSections()

Constructs all sections of an application.

buildSubSectionChartGenerator()

Builds subsection of settings section that configures parameters of generated chart.

buildSubSectionExportData()

Builds subsection of settings section that allows exporting generated chart and statistics.

*class* gui.Gui

Bases: object

Base class responsible for constructing window application.

buildGui()

Constructs and runs root window for an application.

buildSections()

Method meant to be overwritten. Constructs all sections of an application.

### hubCharts module

*class* hubCharts.HubCharts

Bases: object

Class with methods concerning loaded data in form of charts.

addChart(*df: pandas.core.frame.DataFrame*, *title: str*)

Adds chart to list.

Parameters

* **df** – Data frame from which charty will be built.
* **sectititleon** – Title of chart.

displayChart(*chartId: int*, *section: tkinter.Frame*, *moduleDisplayer*)

Displays chart in specified location.

Parameters

* **chartId** – ID of chart to display.
* **section** – GUI component where chart should be displayed.
* **moduleDisplayer** – Method called in-line, that applies patches on chart from all active time series modules.

getAllChartDisplayNames()

Returns display names for all loaded charts.

Returns

Requested list.

getAllCharts()

getChartById(*chartId: int*) → [chart.Chart](index.html#chart.Chart)

Returns chart of argument’s id. Names should be unique in order to find a single instance

Parameters

**chartId** – ID of the chart.

Returns

Requested list.

getChartIdByDisplayName(*name: str*) → int

Returns id of chart with given display name.

Parameters

**name** – Display name, with format ‘{chart title} ({chart id})’.

Returns

Requested list.

removeChart(*chartId: int*)

Removes chart from list.

Parameters

**chartId** – ID of chart to remove.

### hubImporters module

*class* hubImporters.HubImporters

Bases: object

Class with methods concerning data importers.

addMethod(*method:* [*importers.baseimport.ImportMethod*](index.html#importers.baseimport.ImportMethod))

Adds file extension method to list.

Parameters

**module** – Import method to add.

getAllDrivers()

getAllMethods()

getImporterByFileextension(*fileextension: str*) → [importers.baseimport.ImportMethod](index.html#importers.baseimport.ImportMethod)

Returns import method based on operated file extension.

Parameters

**fileextension** – File extension of operated method.

Returns

Requested method.

loadImporters() → None

Loads the list with all provided importers.

### hubTSModules module

from tkinter import \* from tkinter import ttk import numpy as np import matplotlib.patches as patches import math import colorsys import statsmodels.api as sm from pandas.core.frame import DataFrame import pandas as pd

*class* hubTSModules.HubTSModules

Bases: object

Class with methods concerning loaded data in form of charts.

addModule(*module:* [*tsmodules.basemod.TSModule*](index.html#tsmodules.basemod.TSModule))

Adds time series module to list.

Parameters

**module** – Time series module to add.

displayModules(*ax*, *plotdf*)

Displays all selected modules’ patches on selected plot.

Parameters

* **ax** – Reference to plot’s axes.
* **plotdf** – Data frame with original signal.

getAllActiveModuleNames()

Returns all active time series module names.

Returns

Requested list.

getAllActiveModules()

Returns all active time series modules.

Returns

Requested list.

getAllModuleNames()

Returns all module names.

Returns

All module names.

getAllModules()

getModuleByName(*name: str*) → [tsmodules.basemod.TSModule](index.html#tsmodules.basemod.TSModule)

Returns module of argument’s name.

Parameters

**name** – Name of the module.

Returns

Requested module.

loadModules()

Loads the list with all provided modules.

### importers package

#### Submodules

#### importers.baseimport module

*class* importers.baseimport.ImportMethod(*extension*, *displayname*, *defaultTitle*)

Bases: object

Base class responsible for loading time series data from different files.

defaultTitle*: str = ''*

Default, preffered title of dataset, which should be applied after extracting data.

directory*: str = ''*

Directory where data is located.

displayname*: str = ''*

Format of import method displayed in file dialog.

extension*: str = ''*

Supported format of import method.

extractData() → pandas.core.frame.DataFrame

Method meant to be overwritten. Extracts signal from provided data.

Returns

Data frame of single, selected time series.

importSettingsGui(*section: tkinter.ttk.Frame*)

Method meant to be overwritten. Provides GUI elements for selection of specific signal from provided data.

Parameters

**section** – GUI component where chart should be displayed.

#### importers.csvimport module

*class* importers.csvimport.CsvImportMethod(*extension*, *displayname*, *defaultTitle*)

Bases: [importers.baseimport.ImportMethod](#importers.baseimport.ImportMethod)

Class responsible for loading time series data from .csv files.

extractData() → pandas.core.frame.DataFrame

Extracts signal from provided data.

Returns

Data frame of single, selected time series.

importSettingsGui(*section: tkinter.ttk.Frame*)

Provides GUI elements for selection of specific signal from provided data.

onXYprovided()

Determines behaviour of import method’s GUI behaviour when X and Y axis data are provided.

onXchange(*event*)

Determines behaviour of import method’s GUI behaviour when X axis data is provided.

onYchange(*event*)

Determines behaviour of import method’s GUI behaviour when Y axis data is provided.

#### importers.xlsimport module

*class* importers.xlsimport.XlsImportMethod(*extension*, *displayname*, *defaultTitle*)

Bases: [importers.baseimport.ImportMethod](#importers.baseimport.ImportMethod)

Class responsible for loading time series data from .xls files.

extractData() → pandas.core.frame.DataFrame

Extracts signal from provided data.

Returns

Data frame of single, selected time series.

importSettingsGui(*section: tkinter.ttk.Frame*)

Provides GUI elements for selection of specific signal from provided data.

onSheetChange(*event*)

Determines behaviour of import method’s GUI behaviour when sheet is chosen.

onXYprovided()

Determines behaviour of import method’s GUI behaviour when X and Y axis data are provided.

onXchange(*event*)

Determines behaviour of import method’s GUI behaviour when X axis data is provided.

onYchange(*event*)

Determines behaviour of import method’s GUI behaviour when Y axis data is provided.

#### Module contents

### loader module

loader.conf *= None*

List with application configurations.

loader.dirs *= None*

List with directories.

loader.lang *= None*

List with language labels.

loader.loadConfig(*language: str*) → bool

Loads with configuration lists from config.yml file.

Parameters

**language** – Language to load.

Returns

Should program be started.

### logger module

logger.log(*text: str*)

Logs a message to a component.

Parameters

**text** – Text to log.

logger.win*: tkinter.Text = None*

Text component where logs are stored.

### tsmodules package

#### Submodules

#### tsmodules.autocorrelation module

*class* tsmodules.autocorrelation.AutocorrelationsModule(*name*)

Bases: [tsmodules.basemod.TSModule](#tsmodules.basemod.TSModule)

Class implementing autocorrelation and partial autocorrelation time series module.

buildConfig(*section: tkinter.ttk.Frame*)

Provides GUI elements for time series module configuration.

Parameters

**section** – GUI component where module configuration should be displayed.

displayModule(*ax*, *plotdf: pandas.core.frame.DataFrame*)

Processes analyzing module and displays results on chart’s axes.

Parameters

* **ax** – Reference to plot’s axes.
* **plotdf** – Data frame with original signal.

#### tsmodules.basemod module

*class* tsmodules.basemod.TSModule(*name*)

Bases: object

Base class for implementation of single time series module.

buildCheckbox(*section: tkinter.ttk.Frame*)

Provides GUI elements for time series module activation.

Parameters

**section** – GUI component where module activation should be displayed.

buildConfig(*section: tkinter.ttk.Frame*)

Method meant to be overwritten. Provides GUI elements for time series module configuration.

Parameters

**section** – GUI component where module configuration should be displayed.

buildMenu(*section: tkinter.ttk.Frame*)

Provides GUI elements for time series module items.

Parameters

**section** – GUI component where module items should be displayed.

displayModule(*ax*, *plotdf: pandas.core.frame.DataFrame*)

Method meant to be overwritten. Processes analyzing module and displays results on chart’s axes.

Parameters

* **ax** – Reference to plot’s axes.
* **plotdf** – Data frame with original signal.

getDisplayColor() → str

Returns color used to display module’s data.

Returns

Color used to display module’s data.

isactive *= None*

Should this module be considered in time series data analysis.

name *= 'Unnamed Module'*

Name of time series module.

outputDataframe *= Empty DataFrame Columns: [] Index: []*

Data frame with processed time series module data, that was displayed on chart’s axes, or that couldn’t be displayed with line graph.

#### tsmodules.decompose module

*class* tsmodules.decompose.DecomposeModule(*name*)

Bases: [tsmodules.basemod.TSModule](#tsmodules.basemod.TSModule)

Class implementing decomposition of time series.

buildConfig(*section: tkinter.ttk.Frame*)

Provides GUI elements for time series module configuration.

Parameters

**section** – GUI component where module configuration should be displayed.

displayModule(*ax*, *plotdf: pandas.core.frame.DataFrame*)

Processes analyzing module and displays results on chart’s axes.

Parameters

* **ax** – Reference to plot’s axes.
* **plotdf** – Data frame with original signal.

#### tsmodules.differentation module

*class* tsmodules.differentation.DifferentiationModule(*name*)

Bases: [tsmodules.basemod.TSModule](#tsmodules.basemod.TSModule)

Class implementing differentiation time series module.

buildConfig(*section: tkinter.ttk.Frame*)

Provides GUI elements for time series module configuration.

Parameters

**section** – GUI component where module configuration should be displayed.

displayModule(*ax*, *plotdf: pandas.core.frame.DataFrame*)

Processes analyzing module and displays results on chart’s axes.

Parameters

* **ax** – Reference to plot’s axes.
* **plotdf** – Data frame with original signal.

#### tsmodules.originaldata module

*class* tsmodules.originaldata.OriginalDataDisplayModule(*name*)

Bases: [tsmodules.basemod.TSModule](#tsmodules.basemod.TSModule)

Class implementing original data display time series module.

displayModule(*ax*, *plotdf: pandas.core.frame.DataFrame*)

Processes analyzing module and displays results on chart’s axes.

Parameters

* **ax** – Reference to plot’s axes.
* **plotdf** – Data frame with original signal.

#### tsmodules.rollingmean module

*class* tsmodules.rollingmean.RollingMeanModule(*name*)

Bases: [tsmodules.basemod.TSModule](#tsmodules.basemod.TSModule)

Class implementing rolling mean time series module.

buildConfig(*section: tkinter.ttk.Frame*)

Provides GUI elements for time series module configuration.

Parameters

**section** – GUI component where module configuration should be displayed.

displayModule(*ax*, *plotdf: pandas.core.frame.DataFrame*)

Processes analyzing module and displays results on chart’s axes.

Parameters

* **ax** – Reference to plot’s axes.
* **plotdf** – Data frame with original signal.

#### tsmodules.rollingstd module

*class* tsmodules.rollingstd.RollingStdModule(*name*)

Bases: [tsmodules.basemod.TSModule](#tsmodules.basemod.TSModule)

Class implementing rolling standard deviation time series module.

buildConfig(*section: tkinter.ttk.Frame*)

Provides GUI elements for time series module configuration.

Parameters

**section** – GUI component where module configuration should be displayed.

displayModule(*ax*, *plotdf: pandas.core.frame.DataFrame*)

Processes analyzing module and displays results on chart’s axes.

Parameters

* **ax** – Reference to plot’s axes.
* **plotdf** – Data frame with original signal.

#### tsmodules.ruptureeval module

*class* tsmodules.ruptureeval.RupturesEvaluation(*name*)

Bases: [tsmodules.basemod.TSModule](#tsmodules.basemod.TSModule)

Class implementing change point detection evaluation in time series modules. Evaluations include Hausdorff metric, rand index, precision and recall.

buildConfig(*section: tkinter.ttk.Frame*)

Provides GUI elements for time series module configuration.

Parameters

**section** – GUI component where module configuration should be displayed.

displayModule(*ax*, *plotdf: pandas.core.frame.DataFrame*)

Processes analyzing module and displays results on chart’s axes.

Parameters

* **ax** – Reference to plot’s axes.
* **plotdf** – Data frame with original signal.

#### tsmodules.rupturesmods module

*class* tsmodules.rupturesmods.RupturesBinsegModule(*name*)

Bases: [tsmodules.rupturesmods.RupturesModule](#tsmodules.rupturesmods.RupturesModule)

Class implementing Binary segmentaton algorithm that detects change points.

calculateChangePoints(*signal*) → list

Calculates change point indexes.

Parameters

**signal** – Time series data.

Returns

Indexes of change points.

*class* tsmodules.rupturesmods.RupturesBottomUpModule(*name*)

Bases: [tsmodules.rupturesmods.RupturesModule](#tsmodules.rupturesmods.RupturesModule)

Class implementing Bottom-Up algorithm that detects change points.

calculateChangePoints(*signal*) → list

Calculates change point indexes.

Parameters

**signal** – Time series data.

Returns

Indexes of change points.

*class* tsmodules.rupturesmods.RupturesModule(*name*)

Bases: [tsmodules.basemod.TSModule](#tsmodules.basemod.TSModule)

Class implementing change point detection algorithms in time series modules.

buildConfig(*section: tkinter.ttk.Frame*)

Provides GUI elements for time series module configuration.

Parameters

**section** – GUI component where module configuration should be displayed.

buildConfigAlgoParams(*section: tkinter.ttk.Frame*)

Method meant to be overwritten. Provides GUI elements for specific change point detection algorithms’ parameters.

Parameters

**section** – GUI component where specific parameters should be displayed.

calculateChangePoints(*signal*) → list

Method meant to be overwritten. Calculates change point indexes.

Parameters

**signal** – Time series data.

Returns

Indexes of change points.

displayModule(*ax*, *plotdf: pandas.core.frame.DataFrame*)

Processes analyzing module and displays results on chart’s axes.

Parameters

* **ax** – Reference to plot’s axes.
* **plotdf** – Data frame with original signal.

*class* tsmodules.rupturesmods.RupturesPeltModule(*name*)

Bases: [tsmodules.rupturesmods.RupturesModule](#tsmodules.rupturesmods.RupturesModule)

Class implementing Pelt algorithm that detects change points.

calculateChangePoints(*signal*) → list

Calculates change point indexes.

Parameters

**signal** – Time series data.

Returns

Indexes of change points.

*class* tsmodules.rupturesmods.RupturesWindowModule(*name*)

Bases: [tsmodules.rupturesmods.RupturesModule](#tsmodules.rupturesmods.RupturesModule)

Class implementing Window algorithm that detects change points.

buildConfigAlgoParams(*section: tkinter.ttk.Frame*)

Provides GUI elements for specific change point detection algorithms’ parameters.

Parameters

**section** – GUI component where specific parameters should be displayed.

calculateChangePoints(*signal*) → list

Calculates change point indexes.

Parameters

**signal** – Time series data.

Returns

Indexes of change points.

#### Module contents

# Indices and tables

* [Index](genindex.html)
* [Module Index](py-modindex.html)
* [Search Page](search.html)

# [timeseries](#Xa39a3ee5e6b4b0d3255bfef95601890afd80709)

### Navigation

Contents:

* [Projekt](index.html#document-modules)

### Related Topics

* [Documentation overview](#Xa39a3ee5e6b4b0d3255bfef95601890afd80709)

©2022, bp. | Powered by [Sphinx 4.3.2](http://sphinx-doc.org/) & [Alabaster 0.7.12](https://github.com/bitprophet/alabaster)