BayesCMD Documentation

Release

Joshua Russell-Buckland

Oct 06, 2017

CONTENTS:

1 bcmdModel **3**
- 1.1 Running the BCMD Model . . . 3
- 1.2 Input Creation . . . 3

2 abc **5**
- 2.1 Distances . . . 5

3 jsonParsing **9**

4 Miscellaneous **11**
- 4.1 Utility Functions . . . 11
- 4.2 Processing Results . . . 11

5 Indices and tables **17**

Python Module Index **19**

Index **21**

BayesCMD is a package intended to expand the capabilities of the Brain/Circulation Modelling (BCMD) framework. It introduces the ability to obtain posterior distributions for model parameters by using Approximate Bayesian Computation (ABC).

CHAPTER

ONE

BCMDMODEL

1.1 Running the BCMD Model

class bayescmd.bcmdModel.**ModelBCMD**(*model_name*, *inputs=None*, *params=None*, *times=None*, *outputs=None*, *burn_in=999*, *create_input=True*, *input_file=None*, *suppress=False*, *workdir=None*, *deleteWorkdir=False*, *timeout=30*, *basedir='..'*, *debug=False*, *testing=False*)

BCMD model class. this can be used to create inputs, run simulations etc.

create_default_input()

Method to create input file and write to string buffer for acces direct from memory.

create_initialised_input()

Method to create input file and write to string buffer for access direct from memory.

output_parse()

Function to parse the output files into a dictionary.

write_default_input()

Function to write a default input to file.

write_initialised_input()

Function to write a default input to file.

1.2 Input Creation

Input files are required by the BCMD model.

1.2.1 input_creation

Create input files for use with a BCMD model.

Input files are needed in order to set model parameters and provide driving inputs.

class bayescmd.bcmdModel.**InputCreator**(*times*, *inputs*, *outputs=None*, *params=None*, *filename=None*)

Create an input file by passing relevant information to the class.

This input file is then used to create an input file that can either be written to file or kept in buffer and passed directly to the model.

Parameters

- **times** (list of float or int) – List of times at which measurement data has been collected and needs to be simulated.
- **inputs** (*dict*) – Dictionary of model inputs and their values. Has form {'names' : list of str, 'values' : list of list of float} where *names* should be a list of each model input name, matching up to the model inputs and *values* would be a list of lsits, where each sublist is the input values for that time point. With this in mind, the length of *inputs['values']'* should equal length of *times*.
- **filename** (str, optional) – Name of the input file to be written to if writing to file is required. Default is None.
- **params** (dict of str: float, optional) – Dictionary of {'parameter': param_value}
- **outputs** (list of str, optional) – List of model outputs to return.

times
list of float or int – List of times at which measurement data has been collected and needs to be simulated.

inputs
dict – Dictionary of model inputs and their values. Has form {'names' : list of str, 'values' : list of list of float} where *names* should be a list of each model input name, matching up to the model inputs and *values* would be a list of lsits, where each sublist is the input values for that time point. With this in mind, the length of *inputs['values']'* should equal length of *times*.

f_out
StringIO() – String buffer object to which the input file will be written.

filename
str – Name of the input file to be written to if writing to file is required. Default is None.

params
dict of str: float. – Dictionary of {'parameter': param_value}

outputs
list of str – List of model outputs to return.

default_creation()
Create a default input file from given arguments.

Assumes parameters remain unchanged from default values.

Returns Returns the input file as a String.IO() buffer object.

Return type String.IO()

initialised_creation(*burn_in*)
Create an input file from given arguments.

Creates an input file thatcan have non-default parameter values and outputs, as well as a burn in period. Assumes parameters remain constant for the full duration of the simulation.

Parameters **burn_in** (float or int) – Length of burn in period at start of the simulation.

Returns Returns the input file as a String.IO() buffer object.

Return type String.IO()

input_file_write()
Write input file from buffer to file.

CHAPTER

TWO

ABC

The *abc* subpackage is used to handle the Approximate Bayesian Computation (ABC) specific components of BayesCMD. This includes running the model multiple times in a batch process, calculating distances between datasets and generating priors for parameters.

2.1 Distances

Use to generate distance measures between simulated and real time series.

`bayescmd.abc.distances.`**`DISTANCES`**

dict – Dictionary contianing the distance aliases, mapping to the functions.

`exception` `bayescmd.abc.distances.`**`Error`**

Base class for exceptions in this module.

`exception` `bayescmd.abc.distances.`**`ZeroArrayError`**

Exception raised for errors in the zero array.

`bayescmd.abc.distances.`**`check_for_key`**(*dictionary*, *target*)

Check that a dictionary contains a key, and if so, return its data.

Parameters

- **`dictionary`** (*`dict`*) – Dictionary to check for *target* key.
- **`target`** (*`str`*) – String containing the target variable that is expected to be found in *dictionary*

Returns data – List of data found in *dictionary*. This is likely to be the time series data collected experimentally or generated by the model.

Return type list

`bayescmd.abc.distances.`**`euclidean_dist`**(*data1*, *data2*)

Get the euclidean distance between two numpy arrays.

Parameters

- **`data1`** (*`np.ndarray`*) – First data array.

  The shape should match that of data2 and the number of rows should match the number of model outputs i.e. 2 model outputs will be two rows.

- **`data2`** (*`np.ndarray`*) – Second data array.

  The shape should match that of data1 and the number of rows should match the number of model outputs i.e. 2 model outputs will be two rows.

Returns **d** – Euclidean distance measure

Return type float

`bayescmd.abc.distances.`**`get_distance`**(*actual_data*, *sim_data*, *targets*, *zero_flag*, *distance='euclidean'*, *normalise=False*)

Obtain distance between two sets of data.

Get a distance as defined by *distance* between two sets of data as well as between each signal in the data.

Parameters

- **`actual_data`** (*`dict`*) – Dictionary of actual data, as generated by `bayescmd.abc.data_import.import_actual_data()`
- **`sim_data`** (*`dict`*) – Dictionary of simulated data, as created by *`bayescmd.bcmdModel.ModelBCMD.output_parse()`*
- **`targets`** (list of `str`) – List of model targets, which should all be strings.
- **`zero_flag`** (*`dict`*) – Dictionary of form target(`str`): bool, where bool indicates whether to zero that target.

  Note: zero_flag keys should match targets list.
- **`distance`** (*`str, optional`*) – Name of distance measure to use. One of ['euclidean', 'manhattan', 'MAE', 'MSE'], where default is 'euclidean'.
- **`normalise`** (*`bool, optional`*) – Boolean flag to indicate whether the signals need normalising, default is False. Current normalisation is done using z-score but that is likely to change with time.

Returns

distances –

Dictionary of form: {'TOTAL': summed distance of all signals, 'target1: distance of 1st target', … 'targetN': distance of Nth target }

Return type dict

`bayescmd.abc.distances.`**`manhattan_dist`**(*data1*, *data2*)

Get the Manhattan distance between two numpy arrays.

Parameters

- **`data1`** (*`np.ndarray`*) – First data array.

  The shape should match that of data2 and the number of rows should match the number of model outputs i.e. 2 model outputs will be two rows.
- **`data2`** (*`np.ndarray`*) – Second data array.

  The shape should match that of data1 and the number of rows should match the number of model outputs i.e. 2 model outputs will be two rows.

Returns **d** – Manhattan distance measure

Return type float

`bayescmd.abc.distances.`**`mean_absolute_error_dist`**(*data1*, *data2*)

Get the normalised manhattan distance between two numpy arrays.

Parameters

- **data1** (*np.ndarray*) – First data array.

  The shape should match that of data2 and the number of rows should match the number of model outputs i.e. 2 model outputs will be two rows.
- **data2** (*np.ndarray*) – Second data array.

  The shape should match that of data1 and the number of rows should match the number of model outputs i.e. 2 model outputs will be two rows.

Returns **d** – Normalised Manhattan distance measure

Return type float

`bayescmd.abc.distances.`**`mean_square_error_dist`**(*data1*, *data2*)

Get the Mean Square Error distance between two numpy arrays.

Parameters

- **data1** (*np.ndarray*) – First data array.

  The shape should match that of data2 and the number of rows should match the number of model outputs i.e. 2 model outputs will be two rows.
- **data2** (*np.ndarray*) – Second data array.

  The shape should match that of data1 and the number of rows should match the number of model outputs i.e. 2 model outputs will be two rows.

Returns **d** – Mean Square Error distance measure

Return type float

`bayescmd.abc.distances.`**`zero_array`**(*array*, *zero_flag*)

Zero an array of data with its initial values.

Parameters

- **array** (*list*) – List of data
- **zero_flags** (*bool*) – Boolean indicating if data needs zeroing

Returns **zerod** – Zero'd list

Return type list

CHAPTER

THREE

JSONPARSING

..automodule:: bayescmd.jsonParsing.modelJSON

CHAPTER

FOUR

MISCELLANEOUS

Here you will find a number of useful functions that are used throughout the general BayesCMD package.

4.1 Utility Functions

Miscellaneous utility functions used throughout BayesCMD.

This module contains a number of utility functions that are used throughout the different BayesCMD subpackages.

`bayescmd.util.`**`findBaseDir`**(*basename*, *max_depth=5*, *verbose=False*)

Get relative path to a BASEDIR. :param basename: Name of the basedir to path to :type basename: str

Returns Relative path to base directory.

Return type StringIO

`bayescmd.util.`**`round_sig`**(*x*, *sig=1*)

Round a value to N sig fig.

Parameters

- **`x`** (*`float`*) – Value to round
- **`sig`** (*`int, optional`*) – Number of sig figs, default is 1

Returns Rounded value

Return type float

4.2 Processing Results

Process results obtained using BayesCMD.

Process the various results obtained using BayesCMD, such as the *parameters.csv* file. It is also possible to concatenate a number of different *parameters.csv* files obtained using parallel batch runs into a single parameters file.

`bayescmd.results_handling.`**`BAYESCMD`**

`str` – Absolute path to base directory. Found using *`bayescmd.util.findBaseDir`*

`bayescmd.results_handling.`**`data_import`**(*pfile*, *nan_sub=100000*, *chunk_size=10000*, *verbose=True*)

Import a parameters file produced by a batch process.

Parameters

- **`pfile`** (*`str`*) – Path to the file of parameters and distances

- **nan_sub** (*int or float, optional*) – Number to substitute for NaN distances/params. Default of 100000
- **chunk_size** (*int, optional*) – Size of chunks to load for dataframe. Default of 10000
- **verbose** (*bool, optional*) – Boolean as to whether include verbose information. Default of True

Returns **result** – Dataframe containing all the parameters and distances, with NaN swapped for nan_sub

Return type pd.DataFrame

`bayescmd.results_handling.`**`data_merge`**(*parent_directory*, *verbose=True*)

Merge a set of parameters.csv files into one.

Parameters

- **parent_directory** (`list` of `str`) – Parent directory to a set of directories each containing model runs and a parameters.csv file.
- **verbose** (`boolean`, optional) – Boolean indicator of whether to print extra information.

Returns Concatenated will be written to file in *parent_directory*

Return type None

`bayescmd.results_handling.`**`diag_kde_plot`**(*x*, ***kws*)

Plot univariate KDE and barplot with median of distribution marked on.

Includes median of distribution as a line and as text.

Parameters

- **x** (*array-like*) – Array-like of data to plot.
- **kws** (*key, value pairings.*) – Other keyword arguments to pass to `sns.distplot`.

Returns **ax** – AxesSubplot object of univariate KDE and bar plot with median marked on as well as text.

Return type `matplotlib.AxesSubplot`

`bayescmd.results_handling.`**`frac_calculator`**(*df*, *frac*)

Calculate the number of lines for a given fraction.

Parameters

- **df** (*pd.DataFrame*) – Data frame to find fraction of. Normally the output of *data_import*
- **frac** (*float*) – The fraction of results to consider. Should be given as a percentage i.e. 1=1%, 0.1=0.1%

Returns Number of lines that make up the fraction.

Return type int

`bayescmd.results_handling.`**`get_output`**(*model_name*, *p*, *times*, *input_data*, *d0*, *targets*, *distance='euclidean'*, *zero_flag=None*)

Generate model output and distances.

Parameters

- **model_name** (`str`) – Name of model

- **p** (`dict`) – Dict of form {'parameter': value} for which posteriors are being investigated.
- **times** (`list` of `float`) – List of times at which the data was collected.
- **input_data** (`dict`) – Dictionary of input data as generated by `abc.inputParse`.
- **targets** (`list` of `str`) – List of model outputs against which the model is being optimised.
- **distance** (`str`) – Distance measure. One of 'euclidean', 'manhattan', 'MAE', 'MSE'.
- **zero_flag** (*dict*) – Dictionary of form target(`str`): bool, where bool indicates whether to zero that target.

  Note: zero_flag keys should match targets list.

Returns A tuple of (p, model output data).

Return type `tuple`

`bayescmd.results_handling.`**`histogram_plot`**(*df*, *distance='euclidean'*, *fraction=1*, *n_bins=100*)

Plot histogram of distance values.

Plot a histogram showing the distribution of distance values for a given fraction of all distances in the dataframe. Distance values will have been calculated during the batch process.

Parameters

- **df** (`pandas.DataFrame`) – Dataframe of distances and parameters, generated using *`data_import()`*
- **distance** (`str`, optional) – Distance measure. One of 'euclidean', 'manhattan', 'MAE', 'MSE'. Default is 'euclidean'.
- **fraction** (`float`, optional) – Fraction of all distances to plot. Varies from 0 to 1. Default is 1.
- **n_bins** (`int`, optional) – Number of histogram bins. Default is 100.

Returns AxesSubplotobject that contains histogram of distance values.

Return type matplotlib.AxesSubplot

`bayescmd.results_handling.`**`kde_plot`**(*df*, *params*, *frac*, *plot_param=1*, *n_ticks=6*, *d='euclidean'*, *verbose=False*)

Plot the model parameters pairwide as a KDE.

Parameters

- **df** (`pandas.DataFrame`) – Dataframe of distances and parameters, generated using *`data_import()`*
- **params** (`list` of `str`) – List of model parameters to compare pairwise.
- **frac** (`float`) – Fraction of results to consider. Should be given as a percentage i.e. 1=1%, 0.1=0.1%
- **plot_param** (`int`) – Which group to plot:

  0: Outside posterior 1: Inside posterior 2: Failed run
- **n_ticks** (`int`, optional) – Number of x-axis ticks. Useful when a large number of parameters are bring compared, as the axes can become crowded if the number of ticks is too high.
- **d** (`str`, optional) – Distance measure. One of 'euclidean', 'manhattan', 'MAE', 'MSE'.

Note: Should be given as a raw string if latex is used i.e. *r'MAE'*.

- **verbose** (boolean, optional) – Boolean to indicate verbosity. Default is False.

Returns **g** – Seaborn pairgrid object is returned in case of further formatting.

Return type seaborn.PairGrid

bayescmd.results_handling.**plot_repeated_outputs**(*df*, *model_name*, *parameters*, *input_path*, *inputs*, *openopt_path*, *targets*, *n_repeats*, *frac*, *distance='euclidean'*, *zero_flag=None*)

Generate model output and distances multiple times.

Parameters

- **model_name** (str) – Name of model. Should match the modeldef file for model being generated i.e. model_name of 'model'' should have a modeldef file 'model1.modeldef'.
- **parameters** (list of str) – List of parameters for which posteriors are being investigated.
- **input_path** (str) – Path to the true data file
- **inputs** (list of str) – List of model inputs.
- **targets** (list of str) – List of model outputs against which the model is being optimised.
- **n_repeats** – Number of times to generate output data
- **frac** (float) – Fraction of results to consider. Should be given as a percentage i.e. 1=1%, 0.1=0.1%
- **distance** (str) – Distance measure. One of 'euclidean', 'manhattan', 'MAE', 'MSE'.
- **zero_flag** (*dict*) – Dictionary of form target(str): bool, where bool indicates whether to zero that target.

  Note: zero_flag keys should match targets list.

Returns

Return type None

bayescmd.results_handling.**run_model**(*model*)

Run a BCMD Model.

Parameters **model** (*bayescmd.bcmdModel.ModelBCMD*) – An initialised instance of a ModelBCMD class.

Returns **output** – Dictionary of parsed model output.

Return type dict

bayescmd.results_handling.**scatter_dist_plot**(*df*, *params*, *frac*, *n_ticks=6*, *d='euclidean'*, *verbose=False*)

Plot distribution of parameters as a scatter PairPlot.

Parameters

- **df** (pandas.DataFrame) – Dataframe of distances and parameters, generated using *data_import()*
- **params** (list of str) – List of model parameters to compare pairwise.

- **frac** (float) – Fraction of results to consider. Should be given as a percentage i.e. 1=1%, 0.1=0.1%
- **n_ticks** (int, optional) – Number of x-axis ticks. Useful when a large number of parameters are bring compared, as the axes can become crowded if the number of ticks is too high.
- **d** (str, optional) – Distance measure. One of 'euclidean', 'manhattan', 'MAE', 'MSE'.

  Note: Should be given as a raw string if latex is used i.e. *r'MAE'*.
- **verbose** (boolean, optional) – Boolean to indicate verbosity. Default is False.

Returns **g** – Seaborn pairgrid object is returned in case of further formatting.

Return type `seaborn.PairGrid`

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

b

bayescmd.abc, 5
bayescmd.abc.distances, 5
bayescmd.bcmdModel.bcmd_model, 3
bayescmd.bcmdModel.input_creation, 3
bayescmd.results_handling, 11
bayescmd.util, 11

INDEX

B

BAYESCMD (in module bayescmd.results_handling), 11
bayescmd.abc (module), 5
bayescmd.abc.distances (module), 5
bayescmd.bcmdModel.bcmd_model (module), 3
bayescmd.bcmdModel.input_creation (module), 3
bayescmd.results_handling (module), 11
bayescmd.util (module), 11

C

check_for_key() (in module bayescmd.abc.distances), 5
create_default_input() (bayescmd.bcmdModel.ModelBCMD method), 3
create_initialised_input() (bayescmd.bcmdModel.ModelBCMD method), 3

D

data_import() (in module bayescmd.results_handling), 11
data_merge() (in module bayescmd.results_handling), 12
default_creation() (bayescmd.bcmdModel.InputCreator method), 4
diag_kde_plot() (in module bayescmd.results_handling), 12
DISTANCES (in module bayescmd.abc.distances), 5

E

Error, 5
euclidean_dist() (in module bayescmd.abc.distances), 5

F

f_out (bayescmd.bcmdModel.input_creation.InputCreator attribute), 4
filename (bayescmd.bcmdModel.input_creation.InputCreator attribute), 4
findBaseDir() (in module bayescmd.util), 11
frac_calculator() (in module bayescmd.results_handling), 12

G

get_distance() (in module bayescmd.abc.distances), 6
get_output() (in module bayescmd.results_handling), 12

H

histogram_plot() (in module bayescmd.results_handling), 13

I

initialised_creation() (bayescmd.bcmdModel.InputCreator method), 4
input_file_write() (bayescmd.bcmdModel.InputCreator method), 4
InputCreator (class in bayescmd.bcmdModel), 3
inputs (bayescmd.bcmdModel.input_creation.InputCreator attribute), 4

K

kde_plot() (in module bayescmd.results_handling), 13

M

manhattan_dist() (in module bayescmd.abc.distances), 6
mean_absolute_error_dist() (in module bayescmd.abc.distances), 6
mean_square_error_dist() (in module bayescmd.abc.distances), 7
ModelBCMD (class in bayescmd.bcmdModel), 3

O

output_parse() (bayescmd.bcmdModel.ModelBCMD method), 3
outputs (bayescmd.bcmdModel.input_creation.InputCreator attribute), 4

P

params (bayescmd.bcmdModel.input_creation.InputCreator attribute), 4
plot_repeated_outputs() (in module bayescmd.results_handling), 14

R

round_sig() (in module bayescmd.util), 11
run_model() (in module bayescmd.results_handling), 14

S

scatter_dist_plot() (in module bayescmd.results_handling), 14

T

times (bayescmd.bcmdModel.input_creation.InputCreator attribute), 4

W

write_default_input() (bayescmd.bcmdModel.ModelBCMD method), 3
write_initialised_input() (bayescmd.bcmdModel.ModelBCMD method), 3

Z

zero_array() (in module bayescmd.abc.distances), 7
ZeroArrayError, 5