
agentMET4FOF Documentation

Bang Xiang Yong

Feb 21, 2020

Contents

1	agentMET4FOF agents	3
2	agentMET4FOF streams	9
3	Indices and tables	11
4	References	13
	Bibliography	15
	Python Module Index	17
	Index	19

agentMET4FOF is a Python software package developed at the [Institute for Manufacturing](#) of the University of Cambridge (UK) as part of the European joint Research Project [EMPIR 17IND12 Met4FoF](#).

For the *agentMET4FOF* homepage go to [GitHub](#).

agentMET4FOF is written in Python 3.

Contents:

agentMET4FOF agents

class agentMET4FOF.agents.**AgentMET4FOF** (*name=""*, *host=None*, *serializer=None*, *transport=None*, *attributes=None*)

Base class for all agents with specific functions to be overridden/supplied by user.

Behavioural functions for users to provide are `init_parameters`, `agent_loop` and `on_received_message`. Communicative functions are `bind_output`, `unbind_output` and `send_output`.

agent_loop ()

User defined method for the agent to execute for *loop_wait* seconds specified either in *self.loop_wait* or explicitly via `'init_agent_loop(loop_wait)'`

To start a new loop, call `init_agent_loop(loop_wait)` on the agent Example of usage is to check the *current_state* of the agent and send data periodically

before_loop ()

This action is executed before initiating the loop

bind_output (*output_agent*)

Forms Output connection with another agent. Any call on `send_output` will reach this newly binded agent

Adds the agent to its list of Outputs.

Parameters `output_agent` (`AgentMET4FOF`) – Agent to be binded to this agent's output channel

convert_to_plotly (*matplotlib_fig*)

Internal method to convert matplotlib figure to plotly figure

Parameters `matplotlib_fig` (`plt.Figure`) – Matplotlib figure to be converted

handle_process_data (*message*)

Internal method to handle incoming message before calling user-defined `on_received_message` method.

If *current_state* is either Stop or Reset, it will terminate early before entering `on_received_message`

init_agent_loop (*loop_wait=1.0*)

Initiates the agent loop, which iterates every `'loop_wait'` seconds

Stops every timers and initiate a new loop.

Parameters `loop_wait` (*int*) – The wait between each iteration of the loop

init_parameters ()

User provided function to initialize parameters of choice.

on_init ()

Internal initialization to setup the agent: mainly on setting the dictionary of Inputs, Outputs, PubAddr.

Calls user-defined `init_parameters()` upon finishing.

Inputs

Dictionary of Agents connected to its input channels. Messages will arrive from agents in this dictionary. Automatically updated when `bind_output()` function is called

Type dict

Outputs

Dictionary of Agents connected to its output channels. Messages will be sent to agents in this dictionary. Automatically updated when `bind_output()` function is called

Type dict

PubAddr_alias

Name of Publish address socket

Type str

PubAddr

Publish address socket handle

Type str

AgentType

Name of class

Type str

current_state

Current state of agent. Can be used to define different states of operation such as “Running”, “Idle”, “Stop”, etc.. Users will need to define their own flow of handling each type of `self.current_state` in the `agent_loop`

Type str

loop_wait

The interval to wait between loop. Call `init_agent_loop` to restart the timer or set the value of `loop_wait` in `init_parameters` when necessary.

Type int

on_received_message (*message*)

User-defined method and is triggered to handle the message passed by Input.

Parameters `message` (*Dictionary*) – The message received is in form
{‘from’:agent_name, ‘data’: data, ‘senderType’: agent_class, ‘channel’:channel_name}
agent_name is the name of the Input agent which sent the message data is the actual content of the message

pack_data (*data*, *channel*=‘data’)

Internal method to pack the data content into a dictionary before sending out.

Parameters

- **data** (*argument*) – Data content to be packed before sending out to agents.
- **channel** (*str*) – Key of dictionary which stores data

Returns Packed message data

Return type dict of the form {'from':agent_name, 'data': data, 'senderType': agent_class, 'channel':channel_name}.

reset ()

This method will be called on all agents when the global *reset_agents* is called by the AgentNetwork and when the Reset button is clicked on the dashboard.

Method to reset the agent's states and parameters. User can override this method to reset the specific parameters.

send_output (data, channel='default')

Sends message data to all connected agents in self.Outputs.

Output connection can first be formed by calling bind_output. By default calls pack_data(data) before sending out. Can specify specific channel as opposed to default 'data' channel.

Parameters

- **data** (*argument*) – Data content to be sent out
- **channel** (*str*) – Key of *message* dictionary which stores data

Returns message

Return type dict of the form {'from':agent_name, 'data': data, 'senderType': agent_class, 'channel':channel_name}.

send_plot (fig=<Figure size 640x480 with 0 Axes>)

Sends plot to agents connected to this agent's Output channel.

This method is different from send_output which will be sent to through the 'plot' channel to be handled.

Parameters **fig** (*Figure*) – Can be either matplotlib figure or plotly figure

Returns The message format is {'from'

Return type agent_name, 'plot': data, 'senderType': agent_class}.

stop_agent_loop ()

Stops agent_loop from running. Note that the agent will still be responding to messages

unbind_output (output_agent)

Remove existing output connection with another agent. This reverses the bind_output method

Parameters **output_agent** (*AgentMET4FOF*) – Agent binded to this agent's output channel

update_data_memory (message)

Updates data stored in *self.memory* with the received message

Checks if sender agent has sent any message before If it did,then append, otherwise create new entry for it

Parameters **message** (*dict*) – Standard message format specified by AgentMET4FOF class

```
class agentMET4FOF.agents.AgentNetwork (ip_addr='127.0.0.1', port=3333, connect=False, dashboard_modules=True,
                                         dashboard_update_interval=3,
                                         log_filename='log_file.csv')
```

Object for starting a new Agent Network or connect to an existing Agent Network specified by ip & port

Provides function to add agents, (un)bind agents, query agent network state, set global agent states Interfaces with an internal _AgentController which is hidden from user

```
add_agent (name=' ', agentType=<class 'agentMET4FOF.agents.AgentMET4FOF'>, log_mode=True)
```

Instantiates a new agent in the network.

Parameters

- **name** (*str*) – Unique name of agent. If left empty, the name will be automatically set to its class name. There cannot be more than one agent with the same name.
- **agentType** (*AgentMET4FOF*) – Agent class to be instantiated in the network.
- **log_mode** (*bool*) – Default is True. Determines if messages will be logged to background Logger Agent.

Returns AgentMET4FOF

Return type Newly instantiated agent

agents ()

Returns all agent names connected to Agent Network.

Returns list

Return type names of all agents

bind_agents (source, target)

Binds two agents communication channel in a unidirectional manner from *source* Agent to *target* Agent

Any subsequent calls of *source.send_output()* will reach *target* Agent's message queue.

Parameters

- **source** (*AgentMET4FOF*) – Source agent whose Output channel will be binded to *target*
- **target** (*AgentMET4FOF*) – Target agent whose Input channel will be binded to *source*

connect (ip_addr='127.0.0.1', port=3333, verbose=True)**Parameters**

- **ip_addr** (*str*) – IP Address of server to connect to
- **port** (*int*) – Port of server to connect to

get_agent (agent_name)

Returns a particular agent connected to Agent Network.

Parameters agent_name (*str*) – Name of agent to search for in the network

set_agents_state (filter_agent=None, state='Idle')

Blanket operation on all agents to set their *current_state* attribute to given state

Can be used to define different states of operation such as “Running”, “Idle”, “Stop”, etc.. Users will need to define their own flow of handling each type of *self.current_state* in the *agent_loop*

Parameters

- **filter_agent** (*str*) – (Optional) Filter name of agents to set the states
- **state** (*str*) – State of agents to set

set_running_state (filter_agent=None)

Blanket operation on all agents to set their *current_state* attribute to “Running”

Users will need to define their own flow of handling each type of *self.current_state* in the *agent_loop*

Parameters filter_agent (*str*) – (Optional) Filter name of agents to set the states

set_stop_state (filter_agent=None)

Blanket operation on all agents to set their *current_state* attribute to “Stop”

Users will need to define their own flow of handling each type of *self.current_state* in the *agent_loop*

Parameters **filter_agent** (*str*) – (Optional) Filter name of agents to set the states

shutdown ()

Shutowns the entire agent network and all agents

start_server (*ip_addr='127.0.0.1', port=3333*)

Parameters

- **ip_addr** (*str*) – IP Address of server to start
- **port** (*int*) – Port of server to start

unbind_agents (*source, target*)

Unbinds two agents communication channel in a unidirectional manner from *source* Agent to *target* Agent

This is the reverse of *bind_agents()*

Parameters

- **source** (*AgentMET4FOF*) – Source agent whose Output channel will be unbinded from *target*
- **target** (*AgentMET4FOF*) – Target agent whose Input channel will be unbinded from *source*

class agentMET4FOF.agents.**DataStreamAgent** (*name="", host=None, serializer=None, transport=None, attributes=None*)

Able to simulate generation of datastream by loading a given DataStreamMET4FOF object.

Can be used in incremental training or batch training mode. To simulate batch training mode, set *pretrain_size=-1*, otherwise, set *pretrain_size* and *batch_size* for the respective See *DataStreamMET4FOF* on loading your own data set as a data stream.

agent_loop ()

User defined method for the agent to execute for *loop_wait* seconds specified either in *self.loop_wait* or explicitly via 'init_agent_loop(loop_wait)'

To start a new loop, call *init_agent_loop(loop_wait)* on the agent Example of usage is to check the *current_state* of the agent and send data periodically

init_parameters (*stream=DataStreamMET4FOF(), pretrain_size=None, batch_size=1, loop_wait=1, randomize=False*)

Parameters

- **stream** (*DataStreamMET4FOF*) – A DataStreamMET4FOF object which provides the sample data
- **pretrain_size** (*int*) – The number of sample data to send through in the first loop cycle, and subsequently, the *batch_size* will be used
- **batch_size** (*int*) – The number of sample data to send in every loop cycle
- **loop_wait** (*int*) – The duration to wait (seconds) at the end of each loop cycle before going into the next cycle
- **randomize** (*bool*) – Determines if the dataset should be shuffled before streaming

reset ()

This method will be called on all agents when the global *reset_agents* is called by the AgentNetwork and when the Reset button is clicked on the dashboard.

Method to reset the agent's states and parameters. User can override this method to reset the specific parameters.

class agentMET4FOF.agents.**MonitorAgent** (*name=""*, *host=None*, *serializer=None*, *transport=None*, *attributes=None*)

Unique Agent for storing plots and data from messages received from input agents.

The dashboard searches for Monitor Agents' *memory* and *plots* to draw the graphs "plot" channel is used to receive base64 images from agents to plot on dashboard

memory

Dictionary of format *{agent1_name : agent1_data, agent2_name : agent2_data}*

Type dict

plots

Dictionary of format *{agent1_name : agent1_plot, agent2_name : agent2_plot}*

Type dict

init_parameters ()

User provided function to initialize parameters of choice.

on_received_message (*message*)

Handles incoming data from 'default' and 'plot' channels.

Stores 'default' data into *self.memory* and 'plot' data into *self.plots*

Parameters *message* (*dict*) – Acceptable channel values are 'default' or 'plot'

reset ()

This method will be called on all agents when the global *reset_agents* is called by the AgentNetwork and when the Reset button is clicked on the dashboard.

Method to reset the agent's states and parameters. User can override this method to reset the specific parameters.

update_plot_memory (*message*)

Updates plot figures stored in *self.plots* with the received message

Parameters *message* (*dict*) – Standard message format specified by AgentMET4FOF class
Message['data'] needs to be base64 image string and can be nested in dictionary for multiple plots Only the latest plot will be shown kept and does not keep a history of the plots.

agentMET4FOF streams

class agentMET4FOF.streams.CosineGenerator (num_cycles=1000)

class agentMET4FOF.streams.DataStreamMET4FOF

Class for creating finite datastream for ML with x as inputs and y as target Data can be fetched sequentially using *next_sample()* or all at once *all_samples()*

For sensors data: The format shape for 2D data stream (num_samples, n_sensors) The format shape for 3D data stream (num_samples, sample_length , n_sensors)

all_samples ()

Returns all the samples in the data stream

Returns samples

Return type dict of the form {'x': current_sample_x, 'y': current_sample_y}

has_more_samples ()

Checks if stream has more samples.

Returns True if stream has more samples.

Return type Boolean

next_sample (batch_size=1)

Fetches the samples from the data stream and advances the internal pointer *current_idx*

Parameters **batch_size** (*int*) – number of batches to get from data stream

Returns samples

Return type dict of the form {'x': current_sample_x, 'y': current_sample_y}

prepare_for_use ()

Prepare the stream for use. Can be the reading of a file, or the generation of a function, or anything necessary for the stream to work after its initialization.

Notes

Every time a stream is created this function has to be called.

reset()

Resets the estimator to its initial state.

Returns

Return type self

class agentMET4FOF.streams.**SineGenerator** (*num_cycles=1000*)

agentMET4FOF.streams.**extract_x_y** (*message*)

Extracts features & target from *message*['data'] with expected structure such as :

1. tuple - (x,y)
2. dict - {'x':x_data,'y':y_data}

Handle data structures of dictionary to extract features & target

CHAPTER 3

Indices and tables

- `genindex`
- `modindex`
- `search`

CHAPTER 4

References

Bibliography

- [Bang2019] Bang X. Yong, A. Brintrup Multi Agent System for Machine Learning Under Uncertainty in Cyber Physical Manufacturing System, 9th Workshop on Service Oriented, Holonic and Multi-agent Manufacturing Systems for Industry of the Future

a

`agentMET4FOF.agents`, 3
`agentMET4FOF.streams`, 9

A

`add_agent()` (*agentMET4FOF.agents.AgentNetwork* method), 5
`agent_loop()` (*agentMET4FOF.agents.AgentMET4FOF* method), 3
`agent_loop()` (*agentMET4FOF.agents.DataStreamAgent* method), 7
`AgentMET4FOF` (*class in agentMET4FOF.agents*), 3
`agentMET4FOF.agents` (*module*), 3
`agentMET4FOF.streams` (*module*), 9
`AgentNetwork` (*class in agentMET4FOF.agents*), 5
`agents()` (*agentMET4FOF.agents.AgentNetwork* method), 6
`AgentType` (*agentMET4FOF.agents.AgentMET4FOF* attribute), 4
`all_samples()` (*agentMET4FOF.streams.DataStreamMET4FOF* method), 9

B

`before_loop()` (*agentMET4FOF.agents.AgentMET4FOF* method), 3
`bind_agents()` (*agentMET4FOF.agents.AgentNetwork* method), 6
`bind_output()` (*agentMET4FOF.agents.AgentMET4FOF* method), 3

C

`connect()` (*agentMET4FOF.agents.AgentNetwork* method), 6
`convert_to_plotly()` (*agentMET4FOF.agents.AgentMET4FOF* method), 3
`CosineGenerator` (*class in agentMET4FOF.streams*), 9

`current_state` (*agentMET4FOF.agents.AgentMET4FOF* attribute), 4

D

`DataStreamAgent` (*class in agentMET4FOF.agents*), 7
`DataStreamMET4FOF` (*class in agentMET4FOF.streams*), 9

E

`extract_x_y()` (*in module agentMET4FOF.streams*), 10

G

`get_agent()` (*agentMET4FOF.agents.AgentNetwork* method), 6

H

`handle_process_data()` (*agentMET4FOF.agents.AgentMET4FOF* method), 3
`has_more_samples()` (*agentMET4FOF.streams.DataStreamMET4FOF* method), 9

I

`init_agent_loop()` (*agentMET4FOF.agents.AgentMET4FOF* method), 3
`init_parameters()` (*agentMET4FOF.agents.AgentMET4FOF* method), 4
`init_parameters()` (*agentMET4FOF.agents.DataStreamAgent* method), 7
`init_parameters()` (*agentMET4FOF.agents.MonitorAgent* method), 8

Inputs (*agentMET4FOF.agents.AgentMET4FOF attribute*), 4

L

loop_wait (*agentMET4FOF.agents.AgentMET4FOF attribute*), 4

M

memory (*agentMET4FOF.agents.MonitorAgent attribute*), 8

MonitorAgent (*class in agentMET4FOF.agents*), 8

N

next_sample() (*agentMET4FOF.streams.DataStreamMET4FOF method*), 9

O

on_init() (*agentMET4FOF.agents.AgentMET4FOF method*), 4

on_received_message() (*agentMET4FOF.agents.AgentMET4FOF method*), 4

on_received_message() (*agentMET4FOF.agents.MonitorAgent method*), 8

Outputs (*agentMET4FOF.agents.AgentMET4FOF attribute*), 4

P

pack_data() (*agentMET4FOF.agents.AgentMET4FOF method*), 4

plots (*agentMET4FOF.agents.MonitorAgent attribute*), 8

prepare_for_use() (*agentMET4FOF.streams.DataStreamMET4FOF method*), 9

PubAddr (*agentMET4FOF.agents.AgentMET4FOF attribute*), 4

PubAddr_alias (*agentMET4FOF.agents.AgentMET4FOF attribute*), 4

R

reset() (*agentMET4FOF.agents.AgentMET4FOF method*), 5

reset() (*agentMET4FOF.agents.DataStreamAgent method*), 7

reset() (*agentMET4FOF.agents.MonitorAgent method*), 8

reset() (*agentMET4FOF.streams.DataStreamMET4FOF method*), 10

S

send_output() (*agentMET4FOF.agents.AgentMET4FOF method*), 5

send_plot() (*agentMET4FOF.agents.AgentMET4FOF method*), 5

set_agents_state() (*agentMET4FOF.agents.AgentNetwork method*), 6

set_running_state() (*agentMET4FOF.agents.AgentNetwork method*), 6

set_stop_state() (*agentMET4FOF.agents.AgentNetwork method*), 6

shutdown() (*agentMET4FOF.agents.AgentNetwork method*), 7

SineGenerator (*class in agentMET4FOF.streams*), 10

start_server() (*agentMET4FOF.agents.AgentNetwork method*), 7

stop_agent_loop() (*agentMET4FOF.agents.AgentMET4FOF method*), 5

U

unbind_agents() (*agentMET4FOF.agents.AgentNetwork method*), 7

unbind_output() (*agentMET4FOF.agents.AgentMET4FOF method*), 5

update_data_memory() (*agentMET4FOF.agents.AgentMET4FOF method*), 5

update_plot_memory() (*agentMET4FOF.agents.MonitorAgent method*), 8