
AIM-CU

Release 1.0.0

Smriti Prathapan, Berkman Sahiner, Dhaval Kadia, Ravi K. Samala

Jan 30, 2025

CONTENTS:

1	Code execution	3
2	Demo	5
2.1	Methods	5
2.2	CUSUM	6
2.3	ARLTheoretical	7
2.4	Utils	8
	Python Module Index	11
	Index	13

A CUSUM-based tool for AI Monitoring

AIM-CU is a statistical tool for AI monitoring using cumulative sum (AIM-CU). AIM-CU computes:

- The parameter choices for change-point detection based on an acceptable false alarm rate
- Detection delay estimates for a given displacement of the performance metric from the target for those parameter choices.

CODE EXECUTION

Clone AIM-CU repository.

```
git clone https://github.com/DIDSR/AIM-CU.git
```

Run the following commands to install required dependencies (Python = 3.10 is used).

```
apt-get -y install python3  
apt-get -y install pip  
cd AIM-CU  
pip install -r requirements.txt
```

Run AIM-CU.

```
python3 app.py
```

Open the URL <http://0.0.0.0:7860> that is running the AIM-CU locally.

AIM-CU can also be run through the demo available at <https://huggingface.co/spaces/didsr/AIM-CU>. If Space is paused, click on Restart button.

2.1 Methods

2.1.1 CUSUM parameters

Table 1: Table: CUSUM parameters

Parameter	Description
μ_{in}	The mean of the performance metric when the process is in-control, i.e., when there is no performance drift
ARL_0	Number of observations before the control chart signals a false detection
σ_{in}	The in-control standard deviation of the metric
ARL_1	Number of observations before the control chart signals a true detection
k	The normalized reference value, which is related to the magnitude of change that one is interested in detecting. $k = 0.5$ is the default choice for detecting a unit standard deviation change
S_hi	Cumulative sum of positive changes in the metric
h	The normalized threshold or control limit (default =4). This threshold determines when the control chart signals a detection
S_lo	Cumulative sum of negative changes in the metric

2.1.2 CUSUM chart

A two-sided CUSUM control chart computes the cumulative differences or deviations of individual observations from the target mean (or in-control mean, μ_{in}). The positive and negative cumulative sums are calculated:

$$S_{hi}(d) = \max(0, S_{hi}(d-1) + x_d - \hat{\mu}_{in} - K)$$

$$S_{lo}(d) = \max(0, S_{lo}(d-1) - x_d + \hat{\mu}_{in} - K)$$

where d denotes a unit of time, x_d is the value of quantity being monitored at time d , $\hat{\mu}_{in}$ is the in-control mean of x_d , and K is a “reference value” related to the magnitude of change that one is interested in detecting. S_{hi} and S_{lo} are the cumulative sum of positive and negative changes. To detect a change in the observed values from the in-control mean, the CUSUM scheme accumulates deviations that are K units away from the in-control mean. Let σ_{in} denote the in-control standard deviation of x_d .

2.2 CUSUM

Cumulative Sum (CUSUM)

@author: smriti.prathapan

class package.cusum.CUSUM

CUSUM class and its functionalities.

change_detection(*normalized_ref_value*: float = 0.5, *normalized_threshold*: float = 4) → None

Detects a change in the process.

Parameters

- **pre_change_days** (*int*) – Number of days for in-control phase.
- **normalized_ref_value** (*float*, *optional*) – Normalized reference value for detecting a unit standard deviation change in mean of the process. Defaults to 0.5.
- **normalized_threshold** (*float*, *optional*) – Normalized threshold. Defaults to 4.

compute_cusum(*x*: list[float], *mu_0*: float, *k*: float) → tuple[list[float], list[float], list[float]]

Compute CUSUM for the observations in x

Parameters

- **x** (*list[float]*) – Performance metric to be monitored
- **mu_0** (*float*) – In-control mean of the observations/performance metric
- **k** (*float*) – Reference value related to the magnitude of change that one is interested in detecting

Returns

Positive cumulative sum, negative cumulative sum, and CUSUM

Return type

tuple[list[float], list[float], list[float]]

initialize() → None

Initialize with the configuration file.

plot_cusum_plotly() → Figure

Plot CUSUM value using Plotly

Returns

CUSUM plot using Plotly graph object.

Return type

go.Figure

plot_input_metric_plotly() → Figure

Plot the input metric using Plotly.

Returns

Scatter plot as Plotly graph object.

Return type

go.Figure

set_df_metric_csv(*data_csv*: DataFrame) → None

Assign the performance metric data to be used for CUSUM.

Parameters

data_csv (*DataFrame or TextFileReader*) – A comma-separated values (csv) file is returned as two-dimensional data structure with labeled axes.

set_df_metric_default() → None

Read the provided performance metric data to be used for CUSUM for an example.

set_init_stats(*init_days: int*) → None

Use initial days to calculate in-control mean and standard deviation.

Parameters

init_days (*int, optional*) – Initial days when observations are considered stable. Defaults to 30.

set_timeline(*data: ndarray*) → None

Set the timeline of observations.

Parameters

data (*np.ndarray*) – Data of the metric values across the observations.

2.3 ARLTheoretical

ARLTheoretical

@author: smriti.prathapan

package.ARLTheoretical.get_ARL_1(*h: float, shift_in_mean: list[float], dict_ARL0_k: OrderedDict*) → DataFrame

Get the ARL1 along with k values.

Parameters

- **h** (*float*) – Normalized threshold.
- **shift_in_mean** (*list[float]*) – List of the values of shift in mean.
- **dict_ARL0_k** (*OrderedDict*) – Data dictionary of ARL0 and k

Returns

Table for ARL1 and k values.

Return type

pd.DataFrame

package.ARLTheoretical.get_ARL_1_h_mu1_k(*h: float, k: float, mu1: float*) → float

Calculate ARL_1 with given Shift in Mean (mu1) and k.

Parameters

- **h** (*float*) – Normalized threshold.
- **k** (*float*) – Normalized reference value.
- **mu1** (*float*) – Intended shift in mean.

Returns

Detection delay (ARL1).

Return type

float

`package.ARLTheoretical.get_ref_value(h: float, list_ARL_0: list[float]) → tuple[DataFrame, OrderedDict]`
 provides normalized reference values k for provided list of ARL0, given the value of normalized threshold h.

Parameters

- **h** (*float*) – Normalized threshold.
- **list_ARL_0** (*list*) – List of ARL0 values.

Returns

Dataframe of ARL0 and k, Data dictionary of ARL0 and k; where k is normalized reference value.

Return type

tuple[pd.DataFrame, OrderedDict]

`package.ARLTheoretical.get_ref_value_k(h: float, ARL_0: float) → float`

Calculation for the reference value for given h and ARL_0.

Parameters

- **h** (*float*) – Normalized threshold.
- **ARL_0** (*float*) – ARL0 value.

Returns

Normalized reference value k.

Return type

float

2.4 Utils

Utilities to handle different operations

`package.utils.get_greatable_as_html(df: DataFrame) → GT`

Get the great_table as HTML from Pandas dataframe.

Parameters

df (*pd.DataFrame*) – Dataframe to render as a table.

Returns

Table in HTML format.

Return type

gt.GT

`package.utils.populate_summary_table_ARL0_k(summary_table_df_ARL0_k: DataFrame, h) → GT`

Populate ARLTheoretical.summary_table_df_ARL0_k.

Parameters

- **summary_table_df_ARL0_k** (*pd.DataFrame*) – Dataframe of ARL0 and its respective values of k.
- **h** (*float*) – Normalized threshold.

Returns

Table of ARL0 and k in HTML format.

Return type

gt.GT

`package.utils.populate_summary_table_ARL1_k(summary_table_df_ARL1_k: DataFrame, dict_ARL0_k: OrderedDict, h) → GT`

Populate Multiindex table specific for ARLTheoretical.summary_table_df_ARL1_k

Parameters

- **summary_table_df_ARL1_k** (*pd.DataFrame*) – Dataframe with ARL1 and k values.
- **dict_ARL0_k** (*OrderedDict*) – Data Dictionary with the mapping between ARL0 and k.
- **h** (*float*) – Normalized threshold.

Returns

Table for ARL1 and k in HTML format.

Return type

gt.GT

PYTHON MODULE INDEX

p

`package.ARLTheoretical`, 7
`package.cusum`, 6
`package.utils`, 8

C

`change_detection()` (*package.cusum.CUSUM method*), 6
`compute_cusum()` (*package.cusum.CUSUM method*), 6
`CUSUM` (*class in package.cusum*), 6

G

`get_ARL_1()` (*in module package.ARLTheoretical*), 7
`get_ARL_1_h_mu1_k()` (*in module package.ARLTheoretical*), 7
`get_greatable_as_html()` (*in module package.utils*), 8
`get_ref_value()` (*in module package.ARLTheoretical*), 7
`get_ref_value_k()` (*in module package.ARLTheoretical*), 8

I

`initialize()` (*package.cusum.CUSUM method*), 6

M

`module`
`package.ARLTheoretical`, 7
`package.cusum`, 6
`package.utils`, 8

P

`package.ARLTheoretical`
`module`, 7
`package.cusum`
`module`, 6
`package.utils`
`module`, 8
`plot_cusum_plotly()` (*package.cusum.CUSUM method*), 6
`plot_input_metric_plotly()` (*package.cusum.CUSUM method*), 6
`populate_summary_table_ARL0_k()` (*in module package.utils*), 8
`populate_summary_table_ARL1_k()` (*in module package.utils*), 8

S

`set_df_metric_csv()` (*package.cusum.CUSUM method*), 6
`set_df_metric_default()` (*package.cusum.CUSUM method*), 7
`set_init_stats()` (*package.cusum.CUSUM method*), 7
`set_timeline()` (*package.cusum.CUSUM method*), 7