**JSON format input file:**

**“Task”:** to describe the task in hand, we consider three sub-hashtables: “type”, “input” and “method”

**“type”:** this is the type of problem we are addressing. In this case is "reverseEngineering"

**“input”:** here we introduce the general input data that any PDS reverse engineering problem will need, namely: Number of variables, Characteristic of polynomial ring and the list of input time series:

* It starts with one field that allow us to introduce comments regarding the file. The first field is “description” to describe the file.
* Number of variables indicated as “numberVariables”. This is a non-zero positive integer number.
* The cardinality of the field for the PDSs. This is indicated as “fieldCardinality”. At the moment this has to be a prime number.
* The input time series data indicated as “timeSeriesData”.
* “timeSeriesData” it is organized as an array of time series data.
* Each one of the time series is described as object “matrix”.
* Each one of the “matrix” objects -that describe an experiment-, we specify:
  + The actual time series data. This is represented as a matrix were columns represent the variables and rows are the ordered time steps.
  + “Name” which refers to the name designated to that experiment
  + “index” this used for Knockout data to specify which variable is knocked out. For example if the 3rd variable is knocked out, then we write the as “index”: [3]. Notice that we can have in one experiment more than one variable knocked out . For example, if variables 4 and 5 are knocked out, then we describe such KO experiment as “index”: [4,5]. When we refer to wildtype data, we simply write an empty“index”: [].

**“method”:** here is were we describe each one of our methods

* "id": "REACT", this will be the unique identifier of each reverse engineering method.
* "description": "evolutionary algorithm", this will be a brief description of the reverse engineering in question
* "parameters": ["your parameter 1","your parameter 2"]
* },