MANUAL

INL/EXT-00-00000, GDE-000
Revision 0
Printed May 26, 2021

Tool for Economic AnaLysis (Teal) User Manual

Economics plugin for RAVEN

Aaron S. Epiney, Paul Talbot, Congjian Wang, Andrea Alfonsi, Elizabeth Worsham

Prepared by
Idaho National Laboratory
Idaho Falls, Idaho 83415

The Idaho National Laboratory is a multiprogram laboratory operated by
Battelle Energy Alliance for the United States Department of Energy
under DOE Idaho Operations Office. Contract DE-AC07-05ID14517.

Issued by the Idaho National Laboratory, operated for the United States Department of Energy by Battelle Energy Alliance.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

INL/EXT-00-00000, GDE-000
Revision 0
Printed May 26, 2021

Tool for Economic AnaLysis (Teal) User Manual

Economics plugin for RAVEN

Aaron S. Epiney, Paul Talbot, Congjian Wang, Andrea Alfonsi, Elizabeth Worsham

Contents

1 Introduction

TEAL (Tool for Economic AnaLysis) is RAVEN plugin aimed to contain and deploy economic analysis for RAVEN workflows.

It leverages the Uncertanty Quantification, Probabilistic Risk Assesment, Parameter Optimizzation and Data Anlysis framework RAVEN to deploy complex economic analyses.

TEAL enables the capability to compute the **NPV (Net Present Value)**, **IRR (Internal Rate of Return)** and the **PI (Profitability Index)** with RAVEN. Furthermore, it allows NPV, IRR or PI search, i.e. TEAL will compute a multiplicative value (for example the production cost) so that the NPV, IRR or PI has a desired value. The plugin allows for a generic definition of cash flows, which drivers are provided by RAVEN. Furthermore, TEAL includes flexible options to deal with taxes, inflation, discounting and offers capabilities to compute a combined cash flow for components with different component lives.

NPV: computes the NPV according to Eq. 1.

$$NPV = \sum_{y=0}^{N} \frac{CF_y}{(1 + DiscountRate)^y} \tag{1}$$

The sum runs over the years $y = 0$ to N. The net cash flows CF_y are the sum of all cash flows defined in the indicator block (see later for how to define these cash flows). N is the least common multiple (LCM) of all component life times involved. This guarantees that the NPV is computed for a time span so that all components reach their end of life in the same year. The individual component cash flows are repeated until the LCM is reached. For example, lets assume the calculation involves two components *Component1* and *Component2* with life times of 60 years and 40 years respectively. N will be 120 years where 2 successive *Component1* and 3 successive *Component2* will be build. For every 'building year', the cash flow for the last year (of the old component) and the year zero (for the newly built component) will be summed. Table 1 shows an example for illustration. The variable sent back to RAVEN, i.e. what needs to be added to the output data object is 'NPV'.

PI: computes the PI according to Eq. 2.

$$PI = \frac{NPV}{Initial\_investment} \tag{2}$$

where the NPV is calculated as explained above and the $Initial\_investment$ is the Total Net Cash flow at year zero, i.e. CF_0 in the example above. The variable sent back to RAVEN, i.e. what needs to be added to the output data object is 'PI'.

Table 1: Example cash flows for NPV calculation.

	Compo 1		Compo 2		
Year	Comp. lifetime	Cash Flow (year)	Compo. Lifetime	Cash Flow (year)	Total Net Cash flow (CF_y)
0	0	CF_0^{comp1}	0	CF_0^{comp2}	$CF_0^{comp1} + CF_0^{comp2}$
1	1	CF_1^{comp1}	1	CF_1^{comp2}	$CF_1^{comp1} + CF_1^{comp2}$
...					
39	39	CF_{39}^{comp1}	39	CF_{39}^{comp2}	$CF_{39}^{comp1} + CF_{39}^{comp2}$
40	40	CF_{40}^{comp1}	40 and 0	CF_{40}^{comp2} $+CF_0^{comp2}$	$CF_{40}^{comp1} + CF_{40}^{comp2}$ $+CF_0^{comp2}$
41	41	CF_{41}^{comp1}	1	CF_1^{comp2}	$CF_{41}^{comp1} + CF_1^{comp2}$
...					
59	59	CF_{59}^{comp1}	19	CF_{19}^{comp2}	$CF_{59}^{comp1} + CF_{19}^{comp2}$
60	60 and 0	CF_{60}^{comp1} $+CF_0^{comp1}$	20	CF_{20}^{comp2}	$CF_{60}^{comp1} + CF_0^{comp1}$ $+CF_{20}^{comp2}$
61	1	CF_1^{comp1}	21	CF_{21}^{comp2}	$CF_1^{comp1} + CF_{21}^{comp2}$
...					
79	19	CF_{19}^{comp1}	39	CF_{39}^{comp2}	$CF_{19}^{comp1} + CF_{39}^{comp2}$
80	20	CF_{20}^{comp1}	40 and 0	CF_{40}^{comp2} $+CF_0^{comp2}$	$CF_{20}^{comp1} + CF_{40}^{comp2}$ $+CF_0^{comp2}$
81	21	CF_{21}^{comp1}	1	CF_1^{comp2}	$CF_{21}^{comp1} + CF_1^{comp2}$
...					
119	59	CF_{59}^{comp1}	39	CF_{39}^{comp2}	$CF_{59}^{comp1} + CF_{39}^{comp2}$
120	60	CF_{60}^{comp1}	40	CF_{40}^{comp2}	$CF_{60}^{comp1} + CF_{40}^{comp2}$

IRR: computes the IRR according to Eq. 3.

$$0 = \sum_{y=0}^{N} \frac{CF_y}{(1+IRR)^y} \tag{3}$$

Same as for the NPV, the sum runs over the years $y = 0$ to N. The net cash flows CF_y are the sum of all cash flows defined in the indicator block (see explanation of NPV above for details). N is the least common multiple (LCM) of all component life times involved. The variable sent back to RAVEN, i.e. what needs to be added to the output data object is 'IRR'.

NPV_search: The NPV search finds a multiplier 'x' that multiplies some of the cash flows, so that the NPV has a desired value (defined by the `target` attribute). The equation solved is shown in Eq. 4.

$$'target' = \sum_{y=0}^{N} \frac{CF_y^{dep\_on\_x}}{(1+DiscountRate)^y} x + \sum_{y=0}^{N} \frac{CF_y^{not\_dep\_on\_x}}{(1+DiscountRate)^y} \tag{4}$$

2 Input Structure

In the following sections we describe the input structure in a general sense, with details in following chapters.

2.1 XML Input

TEAL makes use of the eXtensible Markup Language (XML) for its input structure, similar to RAVEN. XML is made up of nodes, which have parameters and subnodes. For example:

```
<node_tag par_name="par_value">
  <sub_tag sub_par_name="sub_par_value">sub_value</sub_tag>
</node_tag>
```

The node's name (or tag) opens the XML element. In the example, we have two nodes, named **<node_tag>** and **<sub_tag>**. The node **<node_tag>** has a parameter with name **par_name**. The parameter **par_name** has the value **'par_value'**. Similarly, the **<sub_tag>** node has a parameter and corresponding value. The **<sub_tag>** further has a value itself given by **'sub_value'**.

We will use the terminology **<node>**, **<subnode>**, **parameter**, and **'value'** to describe the input structure of HERON.

2.2 Structure

The TEAL XML Input makes use of two main nodes within the root node **<HERON>**:

- **<Global>**, in which general features of the desired solve are described, including general economics to apply, simulation properties such as project length and time stepping, and so forth.
- **<Component>**, in which the cashflows and economics of each physical component are defined.

Details for the two nodes used in TEAL analyses are enumerated the following sections.

3 Installation

This section of the manual describes how to install the TEAL plug-in and use it from within RAVEN [1].

3.1 Installing the plug-in

The TEAL plugin is distributed with RAVEN. If you just want to use the plugin and are not interested in developing additional features or bug-fixes for it, just make sure that you have access to the plugin repository and you will automatically get it when installing RAVEN. For more information, please refere to the RAVEN manual.

3.2 Accessing the plug-in from within RAVEN

The plugin can be accessed as a special subtype of the External model. The syntax is given in Listing 1.

Listing 1: Call TEAL.CashFlow from RAVEN input.

```
<ExternalModel name="Cash_Flow" subType="TEAL.CashFlow">
  <variables> Input and output variables needed by TEAL
     </variables>
  <ExternalXML node="Economics" xmlToLoad="Cash_Flow_input.xml"/>
</ExternalModel>
```

3.3 Running the plugin as a stand-alone python code

In addition to accessing the plug-in from within RAVEN, it can also be run as a stand-alone python program. This is useful for example for testing. However, since TEAL is still a RAVEN plugin, RAVEN needs to be installed and the plugin needs to be in the plugin-folder for that to work.

Assuming RAVEN is installed and the plugin is in the proper directory (execution will generate an error if its not), one can run it using the command shown in Listing 2.

Listing 2: TEAL run as stand-alone python code

```
~/raven --> python plugins/TEAL/src/CashFlow_ExtMode.py -h
usage: Cash_Flow.py [-h] -iXML inp_file -iINP inp_file -o out_file

Run RAVEN TEAL plugin as stand-alone code
```

```
arguments:
  -h, --help        show this help message and exit
  -iXML inp_file    XML TEAL input file name
  -iINP inp_file    TEAL input file name with the input
                    variable list
  -o out_file       Output file name
```

The XML TEAL input is the one that contains the component and cash flow definitions, i.e. the one in **xmlToLoad** in Listing 1.

The TEAL input file contains the variables that RAVEN provides when using the TEAL as a plugin, i.e. the cash flow drivers and multipliers. Listing 3 shows an example of the TEAL input file for the cash flow definitions given in Listing 4.

Listing 3: TEAL run as stand-alone python code

```
Cfdriver1 5.5
multiplier1 1.0
Cfdriver2 10.8
multiplier2 2.0
```

4 TEAL for RAVEN

A generalized module within the TEAL software (called TEAL.CashFlow) for economic analysis within RAVEN has been developed [2]. The module is able to compute the NPV (Net Present Value), the IRR (Internal Rate of Return) and the PI (Profitability Index). Furthermore, it is possible to do an NPV, IRR or PI search, i.e. CashFlow will compute a multiplicative value (for example the production cost) so that the NPV, IRR or PI has a desired value (for details see 1, NPV_search). This CashFlow module has been written using the script language Python. The Python code can be used as an "external model" in RAVEN (for installation and usage instructions, see 3).

The input of **TEAL.CashFlow** is an XML file. An example of the input structure is given in Listing 4. The following section will discuss the different keywords in the input and describe how they are used in the **TEAL.CashFlow** module.

Listing 4: Economics input example.

```
<Economics verbosity='0'>
    <Global>
        <Indicator name='IRR,NPV_search,NPV' target='0'>
            Component1|Cfname1
            Component1|Cfname2
            ...
        </Indicator>
        <DiscountRate>0.08</DiscountRate>
        <tax>0.392</tax>
        <inflation>0.04</inflation>
        <ProjectTime>100</ProjectTime> <!-- optional -->
    </Global>

    <Component name='Component1'>
        <Life_time>20</Life_time>
        <StartTime>10</StartTime> <!-- optional -->
        <Repetitions>3</Repetitions> <!-- optional -->
        <tax>0.3</tax> <!-- optional -->
        <inflation>0.07</inflation> <!-- optional -->
        <CashFlows>
            <Capex name='Cfname1' tax='false' inflation='none'
               multiply='multiplier1' mult_target='false'>
                <driver>Cfdriver1</driver>
                <alpha>-4000000000</alpha>
                <reference>1000000000</reference>
                <X>1.0</X>
            </Capex>
```

```
            <Recurring name='Cfname2' tax='false'
               inflation='none' multiply='multiplier2'
               mult_target='true'>
                 ...
            </Recurring>
            ...
        </CashFlows>
    </Component>

    <Component name='Component2'>
        ...
    </Component>
    ...
</Economics>
```

As one can see, all the specifications of the **TEAL.CashFlow** module are given in the **<Economics>** block. The block accepts an attribute called **verbosity**, which can range from 0 to 100, 0 meaning maximum debug verbosity and 100 meaning errors only. Setting the verbosity to 50 will output (in addition to errors) the NPV, IRR, PI or NPV_mult. Inside the **<Economics>** block, there are two types of blocks: **<Global>** and **<Component>**.

5 CashFlows

5.1 Global

Exactly one **<Global>** block has to be provided. The **<Global>** block does not have any attributes.

The **<Global>** node recognizes the following subnodes:

- **<Indicator>**: ***comma-separated strings***, List of cash flows considered in the computation of the economic indicator. See later for the definition of the cash flows. Only cash flows listed here are considered, additional cash flows defined, but not listed are ignored. The **<Indicator>** node recognizes the following parameters:
  - **name**: ***comma-separated strings, required***, The names of the economic indicators that should be computed. So far, **'NPV'**, **'NPV_search'**, **'IRR'** and **'PI'** are supported. More than one indicator can be asked for. The **name** attribute can contain a comma-separated list as shown in the example in Listing reflst:InputExample.
    Note on IRR and PI search: It should be noted that although the only search keyword allowed in **name** is **NPV_search**, it is possible to perform IRR and PI searches as well.
    - To do an IRR search, the DiscountRate is set to the desired IRR and a NPV search with the target of '0' is performed.
    - To perform a PI search, an NPV search can be performed where the target PI is multiplied with the initial investment.
  - **target**: ***float, optional***, Target value for the NPV search, i.e. **'0'** will look for 'x' so that $NPV(x) = 0$.

  The **<Indicator>** node recognizes the following subnodes:

  - **<DiscountRate>**: ***float***, The discount rate used to compute the NPV and PI. Not used for the computation of the IRR (although it must be input).
  - **<tax>**: ***float***, The standard tax rate used to compute the taxes if no other tax rate is specified in the componet blocks. This is a required input. If a tax rate is specified inside a component block, the componet will use that tax rate. If no tax rate is specified in a component, this standard tax rate is used for the component. See later in the definition of the cash flows for more details how the tax rate is used.
  - **<inflation>**: ***float***, The standard inflation rate used to compute the inflation if no other inflation rate is specified in the componet blocks. This is a required input. If a inflation rate is specified inside a component block, the componet will use that inflation rate. If no inflation rate is specified in a component, this standard inflation rate is used for the component. See later in the definition of the cash flows

- **<ProjectTime>**: ***integer***, This is a optional input. If it is included in the input, the global project time is not the LCM of all components (see **<Indicator>** attribute **name** for more information), but the time indicated here.

5.2 Component

The user can define as many **<Component>** blocks as needed. A component is typically a part of the system that has the same lifetime and the same cash flows, i.e. for example a gas turbine, a battery or a nuclear plant. Each component needs to have a **name** attribute that is unique. Each **<Component>** has to have one **<Life_time>** block and as many **<CashFlow>** blocks as needed.

The **<Component>** node recognizes the following parameters:

- **name**: ***string, required***, The unique name of the component.

The **<Component>** node recognizes the following subnodes:

- **<Life_time>**: ***integer***, The lifetime of the component in years. This is used to compute the least common multiple (LCM) of all components involved in the computation of the economics indicator. For more details see NPV, IRR and PI explanations above.
- **<StartTime>**: ***integer***, This is a optional input. If this input is specified for one or more components, the **<Global>** input **<ProjectTime>** is required. This input specifies the year in which this component is going to be build for the first time, i.e. is going to be included in the cash flows. The default is 0 and the componet is build at the start of the project, i.e. at project year 0. For example, if the **<ProjectTime>** is 100 years, and for this component, the **<StartTime>** is 20 years, the cash flows for this component are going to be zero for years 0 to 19 of the project. Year 20 of the project will be year 0 of this component and so on (project year 21 will be component year 1 etc.).
- **<Repetitions>**: ***integer***, This is a optional input. If this input is specified for one or more components, the **<Global>** input **<ProjectTime>** is required. This input specifies the number of times this component is going to be rebuilt. The default is 0, which indicates that the component is going to be rebuild indefinitely until the project end (**<ProjectTime>**) is reached. Lets assume the **<ProjectTime>** is 100 years, and the component **<Life_time>** is 20 years. Specifying 3 repetitions of this component will build 3 components in succession, at years 0, 20 and 40. For years 61 to 100 of the project, the cash flows for this component will be zero.
- **<tax>**: ***float***, This is a optional input. If the tax rate is specified here, i.e. inside the component block, the componet will use this tax rate. If no tax rate is specified in the component, the standard tax rate from the **<Global>** block is used for the component.

- **<inflation>**: ***float***, This is a optional input. If the inflation rate is specified here, i.e. inside the component block, the componet will use this inflation rate. If no inflation rate is specified in the component, the standard inflation rate from the **<Global>** block is used for the component.

- **<CashFlows>**: The user can define any number of 'cash flows' for a component. Each cash flow is of the form given in Eq. 5 where y is the year from 0 (capital investment) to the end of the **<Life_time>** of the component.

$$CF\_y = mult \cdot \alpha\_y \left(\frac{driver\_y}{ref} \right)^X \tag{5}$$

  The **<CashFlows>** node recognizes the following subnodes:

  - **<Capex>**: The cash flow for capital expenditures The **<Capex>** node recognizes the following parameters:

    - **name**: ***string, required***, The name of the Cash flow. Has to be unique across all components. This is the name that can be listed in the **<Indicator>** node of the **<Global>** block.
    - **tax**: ***[True, Yes, 1, False, No, 0, t, y, 1, f, n, 0], optional***, Can be **true** or **false**. If it is **true**, the cash flow is multiplied by $(1 - tax)$, where tax is the tax rate given in **<tax>** in the **<Global>** block. As an example, the cash flow of *comp2* for year 119 in Listing 4 would become $CF^{comp2}\_39(1 - tax)$. If a cash flow with **tax**=**true** is the driver of another cash flow, the cash flow without the tax is used as driver for the new cash flow. The limitation of having a global tax rate will be lifted in future version of the **TEAL.CashFlow** module. It is planned to have the possibility to input different tax rates for each component, since they might be in different tax regions.
    - **inflation**: ***[real, none], optional***, Can be **real, nominal** or **none**. If it is **real**, the cash flow is multiplied by $(1 + inflation)^{-y}$. If it is **nominal**, the cash flow is multiplied by $(1+inflation)^{y}$. In both cases, inflation is given by **<inflation>** in the **<Global>** block. Furthermore, y goes from year 0 (capital investment) to the LCM of all component lifetimes. This means that the cash flows as expressed in Listing 4 are multiplied with the infloation seen from today, i.e. the cash flow for *comp2* for year 119 assuming it includes **real** inflation would be $CF^{comp2}\_39(1 + inflation)^{-119}$ If a cash flow with **inflation** equal **real** or **nominal** is the driver of another cash flow, the cash flow without the inflation is used as driver for the new cash flow.
    - **mult_target**: ***[True, Yes, 1, False, No, 0, t, y, 1, f, n, 0], optional***, Can be **true** or **false**. If **true**, it means that this cash flow multiplies the search variable 'x' as explained in the NPV_search option above. If the NPV_search option is used, al least one cash flow has to have **mult_target**=**true**.

- **multiply**: ***string, optional***, This is an optional attribute. This can be the name of any scalar variable passed in from RAVEN. This number is $mult$ in Eq. 5 that multiplies the cash flow.

The **<Capex>** node recognizes the following subnodes:

- **<driver>**: ***comma-separated strings, integers, and floats***, The $driver$ in Eq. 5 of the cash flow. This can be any variable passed in from RAVEN or the name of another cash flow. If it is passed in from RAVEN, it has to be either a scalar or a vector with length **<Life_time>** + 1. If its a scalar, all $driver\_y$ in Eq. 5 are the same for all years of the project life. If it is a vector instead, each year of the project **<Life_time>** will have its corresponding value for the driver. If the driver is another cash flow, the project **<Life_time>** of the component to which the driving cash flow belongs has to be the same than the project
- **<alpha>**: ***comma-separated strings, integers, and floats***, α_y multiplier of the cash flow (see Eq. 5). Similar to **<driver>**, can be either scalar or vector. If a vector, exactly **<Life_time>**$+1$ values are expected. One for $y = 0$ to $y =$**<Life_time>**. If a scalar, we assume alpha is zero for all years of the lifetime of the component except the year zero (the provided scalar value will be used for year zero), which is the construction year.
- **<reference>**: ***float***, The ref value of the cash flow (see Eq. 5).
- **<X>**: ***float***, The X exponent (economy of scale factor) of the cash flow (see Eq. 5).
- **<depreciation>**: ***comma-separated strings, integers, and floats***, INSERT The **<depreciation>** node recognizes the following parameters:
  - **scheme**: ***[MACRS, custom], required***, – no description yet –

- **<Recurring>**: The cash flow for recurring cost, such as operation and maintenance cost. The **<Recurring>** node recognizes the following parameters:

  - **name**: ***string, required***, The name of the Cash flow. Has to be unique across all components. This is the name that can be listed in the **<Indicator>** node of the **<Global>** block.
  - **tax**: ***[True, Yes, 1, False, No, 0, t, y, 1, f, n, 0], optional***, Can be **true** or **false**. If it is **true**, the cash flow is multiplied by $(1 - tax)$, where tax is the tax rate given in **<tax>** in the **<Global>** block. As an example, the cash flow of *comp2* for year 119 in Listing 4 would become $CF^{comp2}\_39(1 - tax)$. If a cash flow with **tax**=**true** is the driver of another cash flow, the cash flow without the tax is used as driver for the new cash flow. The limitation of having a global tax rate will be lifted in future version of the **TEAL.CashFlow** module. It is planned to have the possibility to input different tax rates for each component, since they might be in different tax regions.
  - **inflation**: ***[real, none], optional***, Can be **real, nominal** or **none**. If it is **real**, the cash flow is multiplied by $(1 + inflation)^{-y}$. If it is **nominal**, the cash flow is

multiplied by $(1 + inflation)^y$. In both cases, inflation is given by **<inflation>** in the **<Global>** block. Furthermore, y goes from year 0 (capital investment) to the LCM of all component lifetimes. This means that the cash flows as expressed in Listing 4 are multiplied with the infloation seen from today, i.e. the cash flow for *comp2* for year 119 assuming it includes **real** inflation would be $CF^{comp2}\_39(1 + inflation)^{-119}$ If a cash flow with **inflation** equal **real** or **nominal** is the driver of another cash flow, the cash flow without the inflation is used as driver for the new cash flow.

- **mult_target**: ***[True, Yes, 1, False, No, 0, t, y, 1, f, n, 0], optional***, Can be **true** or **false**. If **true**, it means that this cash flow multiplies the search variable 'x' as explained in the NPV_search option above. If the NPV_search option is used, al least one cash flow has to have **mult_target**=**true**.
- **multiply**: ***string, optional***, This is an optional attribute. This can be the name of any scalar variable passed in from RAVEN. This number is $mult$ in Eq. 5 that multiplies the cash flow.

The **<Recurring>** node recognizes the following subnodes:

- **<driver>**: ***comma-separated strings, integers, and floats***, The $driver$ in Eq. 5 of the cash flow. This can be any variable passed in from RAVEN or the name of another cash flow. If it is passed in from RAVEN, it has to be either a scalar or a vector with length **<Life_time>** + 1. If its a scalar, all $driver\_y$ in Eq. 5 are the same for all years of the project life. If it is a vector instead, each year of the project **<Life_time>** will have its corresponding value for the driver. If the driver is another cash flow, the project **<Life_time>** of the component to which the driving cash flow belongs has to be the same than the project
- **<alpha>**: ***comma-separated strings, integers, and floats***, α_y multiplier of the cash flow (see Eq. 5). Similar to **<driver>**, can be either scalar or vector. If a vector, exactly **<Life_time>**$+1$ values are expected. One for $y = 0$ to $y =$**<Life_time>**. If a scalar, we assume alpha is zero for all years of the lifetime of the component except the year zero (the provided scalar value will be used for year zero), which is the construction year.

References

[1] C. Rabiti, A. Alfonsi, J. Cogliati, D. Mandelli, R. Kinoshita, S. Sen, C. Wang, and J. Chen, "Raven user manual," Tech. Rep. INL/EXT-15-34123, March 2017.

[2] A. Epiney, C. Rabiti, A. Alfonsi, P. Talbot, and F. Ganda, "Report on the economic optimization of a demonstration case for a static n-r hes configuration using raven," Tech. Rep. INL/EXT-17-41915, April 2017.


INL
Idaho National Laboratory