

---

# **EqParse Documentation**

***Release***

**Haroon Arshad**

December 16, 2014



## CONTENTS

<b>1</b>	<b>eqparse package</b>	<b>3</b>
1.1	eqparse Package . . . . .	3
1.2	baseparse Class . . . . .	3
1.3	baseparse module . . . . .	3
1.4	cppparser module . . . . .	5
1.5	createlibrary module . . . . .	5
1.6	matlabparser module . . . . .	6
1.7	smc_helper_functions module . . . . .	7
1.8	timer module . . . . .	7
1.9	xppautparser module . . . . .	7
1.10	Module contents . . . . .	7
	<b>Python Module Index</b>	<b>9</b>
	<b>Index</b>	<b>11</b>



Contents:



## EQPARSE PACKAGE

### 1.1 eqparse Package

The equation parse package is a universal parser from a single-input file describing a larger ODE system to different programming languages. Multiple file output of the same language can easily be created and other parser modules can easily be implemented for other language outputs with some python knowledge `eqparse import module initialise`

### 1.2 baseparse Class

```
eqparse.__init__.baseparse  
    alias of eqparse.baseparse
```

### 1.3 baseparse module

```
class eqparse.baseparse.BaseParse  
    Bases: object
```

The main objective of `BaseParse` is to organise universal functions and member variables that are inherited in other class parser modules. Current supplied parsers include the `MatlabParser`, `CppParser` and `XPPautParser` as child modules.

Initialise universal variables usually important in most or all language syntax of the following. if a module inherits this class, the following variables usually need to be redefined according to its language syntax. The following parameters are automatically defined as `__init__` is called

#### Parameters

- **l\_enclose** – The left character to access a piece of memory is usually a left square bracket (C++) or a normal left bracket (MATLAB)
- **r\_enclose** – Similar to :param l\_enclose:
- **comment\_prefix** – character used for single-line comment
- **vec\_counter\_start** – starting number to access first element of a container/array (default: 0)
- **container\_type** – a list of container types used
- **PERM\_I\_LIST** – the input file contain preconditioned statements in csv input file such as if statements or power function.

- **operations** – the required result during parse of the :param PERM\_I\_LIST: list to the respective language syntax

**close\_file** (*file\_n\_str*)

**end** ()

**get\_directory** ()

**get\_index** (*index\_to\_get*)

**get\_list** (*mKey*, *mLib=None*, *mPrefix=''*, *mPostfix=''*, *mIndex=None*)

get a list of a particular property of the variables. The list of properties depends on a key which associates a particular property of a variables library (e.g. comment, rhs).

#### Parameters

- **mKey** – the property type of a variable
- **mLib** – library to use e.g self.data (normal format) self.data\_vec (vectorised format) (default - self.data)
- **mPrefix** – pre-fix the returned property with a string.
- **mPostfix** – post-fix the returned property with a string.

**get\_names** ()

**get\_only\_names** (*oNames*, *mIndex*)

**initialise\_library** (*lib*)

In order to initialise library

**Parameters lib** – The created library from the `createlibrary` class

**Returns** void

**new\_dependency\_index** (*index\_to\_order*, *key\_to\_order*, *is\_return=True*)

**new\_index** (\**argv*)

**new\_names** (*oNames*, *mIndex*, *specifier=['', '']*, *return\_it=True*)

**new\_names\_restricted** (*oNames*, *mIndex*, *specifier=['', '']*, *return\_it=True*)

**new\_order** (\**argv*)

**open\_file** (*file\_n\_str\_key*)

**order\_index** (*index\_to\_order*, *order\_by=None*)

**pattern\_change** (*gIndex*, *orig*, \**argv*)

**pattern\_write** (*w\_file*, *w\_data\_tuple*, *w\_str\_tuple*, *w\_lib=None*, *mIndex=None*, *end\_wrr\_='n'*,  
*beg\_wrr\_=''*)

Write to file output each of the required variables with a similar coded pattern

#### Parameters

- **w\_file** (*str*) – file to write to specified by string used in `open_file` function
- **w\_data\_tuple** (*list*) – combination of strings (if same for each variable) and lists (each with same size) that will be printed in consecutive order
- **w\_str\_tuple** (*str*) –
- **w\_lib** – post-fix the returned property with a string.
- **mIndex** (*list*) – specify index of variables to go through.



**rasterise\_dict\_modifiers** (*refi*)

**replace\_operations** (*ops=None*)

**search\_and\_replace** (*origContainer, repNames, origNames=None, return\_it=True*)

Search and replace the variables of a container of equations (normally the RHS or initial condition) to specified new names. All containers should have same length and coincide to variable name properties

#### Parameters

- **origContainer** – data of formular/equations containing variables one wants to change
- **repNames** – the names one wants to replace to parallel to :param *origNames*:
- **origNames** – the original names to replace (default - original variable names)

**vectorise\_name** (*mIndex, specifier, names\_list=None, return\_it=True*)

**write** (*fileAttr, strAttr, end\_wrr\_='n', beg\_wrr\_=''*)

**write\_comment** (*fileAttr, strAttr, end\_wrr\_='n', beg\_wrr\_=''*)

## 1.4 cppparser module

**class** eqparse.cppparser.**CppParser** (*Lib*)

Bases: eqparse.baseparse.BaseParse

**cpp\_original** ()

## 1.5 createlibrary module

create Library of variables : Read saved model file (current formats: csv, )

**class** eqparse.createlibrary.**CreateLibrary** (*g\_abbrev\_title, name\_of\_file\_list*)

Bases: object

Before calling a parser module, one must call this class with the csv files to be parsed, organised and managed into a dictionary for easy look-up and readability for the different parsers. Currently this module only accepts CSV files. In the future anticipate other possible formats such as XML and possibly a GUI.

Examples are included that parse example ode problems into several formats

OK

**add\_index** (*define\_dict\_keys, cond\_on={}, cond\_not\_on={}*)

**add\_variable** (*mDict*)

**change\_variable** (*mname, mval*)

**complete** ()

**copy\_lib** ()

**copy\_lib\_vec** ()

**create\_data** ()

**dup2** (*n*)

**function\_order** (*\*argv*)

```
get_index_dependency (m_key_to_order='init-value', resp_order_val='name', cond_remove={},  
                      cond_include={})
```

```
set_directory ()
```

Set directory where file specific parser files are saved to, the directory can be a relative path and not necessarily an absolute path

**Parameters** *dir\_str* – string of path to directory

```
table = []
```

```
eqparse.createlibrary.change_a_type (Lib, vName, vKey, vToo)
```

## 1.6 matlabparser module

```
class eqparse.matlabparser.MatlabParser (Lib)
```

Bases: `eqparse.baseparse.BaseParse`

To create matlab parsed file

```
FSA_cvodes ()
```

```
FSA_ode ()
```

Create FSA matlab ODE file (ode.m) to be used with `multi_runfile` and `multi_runfile_slider` modules

```
inset_runfile_slider (newSubDirectory='', file_ext='', no_guess=None, yes_guess=None)
```

```
multi_odefile (newSubDirectory='', file_ext='', no_guess=None, yes_guess=None,  
              wr_modifiers='')
```

Create matlab ODE file (ode.m) to be used with `multi_runfile` and `multi_runfile_slider` modules

```
multi_paramfile (newSubDirectory='', file_ext='', no_guess=None, yes_guess=None)
```

```
multi_runfile (newSubDirectory='', file_ext='', no_guess=None, yes_guess=None,  
              wr_modifiers='')
```

Create run file (run.m) that uses the ode file created from running alongside with `multi_odefile` module. This format is currently set for MATLAB defined implicit ode function - ode15s.

**Parameters**

- *newSubDirectory* –
- *file\_ext* –
- *no\_guess* –
- *yes\_guess* –
- *wr\_modifiers* –

```
multi_runfile_slider (newSubDirectory='', file_ext='', no_guess=None, yes_guess=None)
```

```
p_est_main (param_to_est)
```

```
solve_single ()
```

generate single MATLAB (single\_generic.m) file to solve equations. Runs the Euler finite difference scheme to model with dt=5 (ms)

```
supp_initialise_data ()
```

```
supp_input_ic ()
```

```
vectorised_to_readable()
```

## 1.7 smc\_helper\_functions module

```
eqparse.smc_helper_functions.error(err_str)
```

## 1.8 timer module

```
class eqparse.timer.Timer
```

```
    finish(m_str=None)
```

```
    start()
```

## 1.9 xppautparser module

```
class eqparse.xppautparser.XppautParser(Lib)
```

```
    Bases: eqparse.baseparse.BaseParse
```

```
    parse_one(concateq_unordered=[])
```

```
    printarray(c, specific=None)
```

```
    set_temp_ic_name()
```

```
    truncate_ics()
```

```
    xpp_search_and_define()
```

## 1.10 Module contents

```
eqparse import module initialise
```



**e**

eqparse, 7  
eqparse.\_\_init\_\_, 3  
eqparse.baseparse, 3  
eqparse.cppparser, 5  
eqparse.createlibrary, 5  
eqparse.matlabparser, 6  
eqparse.smc\_helper\_functions, 7  
eqparse.timer, 7  
eqparse.xppautparser, 7



## A

add\_index() (eqparse.createlibrary.CreateLibrary method), 5  
 add\_variable() (eqparse.createlibrary.CreateLibrary method), 5

## B

BaseParse (class in eqparse.baseparse), 3  
 baseparse (in module eqparse.\_\_init\_\_), 3

## C

change\_a\_type() (in module eqparse.createlibrary), 6  
 change\_variable() (eqparse.createlibrary.CreateLibrary method), 5  
 close\_file() (eqparse.baseparse.BaseParse method), 4  
 complete() (eqparse.createlibrary.CreateLibrary method), 5  
 copy\_lib() (eqparse.createlibrary.CreateLibrary method), 5  
 copy\_lib\_vec() (eqparse.createlibrary.CreateLibrary method), 5  
 cpp\_original() (eqparse.cppparser.CppParser method), 5  
 CppParser (class in eqparse.cppparser), 5  
 create\_data() (eqparse.createlibrary.CreateLibrary method), 5  
 CreateLibrary (class in eqparse.createlibrary), 5

## D

dup2() (eqparse.createlibrary.CreateLibrary method), 5

## E

end() (eqparse.baseparse.BaseParse method), 4  
 eqparse (module), 7  
 eqparse.\_\_init\_\_ (module), 3  
 eqparse.baseparse (module), 3  
 eqparse.cppparser (module), 5  
 eqparse.createlibrary (module), 5  
 eqparse.matlabparser (module), 6  
 eqparse.smc\_helper\_functions (module), 7  
 eqparse.timer (module), 7  
 eqparse.xppautparser (module), 7  
 error() (in module eqparse.smc\_helper\_functions), 7

## F

finish() (eqparse.timer.Timer method), 7  
 FSA\_cvodes() (eqparse.matlabparser.MatlabParser method), 6  
 FSA\_ode() (eqparse.matlabparser.MatlabParser method), 6  
 function\_order() (eqparse.createlibrary.CreateLibrary method), 5

## G

get\_directory() (eqparse.baseparse.BaseParse method), 4  
 get\_index() (eqparse.baseparse.BaseParse method), 4  
 get\_index\_dependency() (eqparse.createlibrary.CreateLibrary method), 5  
 get\_list() (eqparse.baseparse.BaseParse method), 4  
 get\_names() (eqparse.baseparse.BaseParse method), 4  
 get\_only\_names() (eqparse.baseparse.BaseParse method), 4

## I

initialise\_library() (eqparse.baseparse.BaseParse method), 4  
 inset\_runfile\_slider() (eqparse.matlabparser.MatlabParser method), 6

## M

MatlabParser (class in eqparse.matlabparser), 6  
 multi\_odefile() (eqparse.matlabparser.MatlabParser method), 6  
 multi\_paramfile() (eqparse.matlabparser.MatlabParser method), 6  
 multi\_runfile() (eqparse.matlabparser.MatlabParser method), 6  
 multi\_runfile\_slider() (eqparse.matlabparser.MatlabParser method), 6

## N

new\_dependancy\_index() (eqparse.baseparse.BaseParse method), 4  
 new\_index() (eqparse.baseparse.BaseParse method), 4

`new_names()` (eqparse.baseparse.BaseParse method), 4  
`new_names_restricted()` (eqparse.baseparse.BaseParse method), 4  
`new_order()` (eqparse.baseparse.BaseParse method), 4

## O

`open_file()` (eqparse.baseparse.BaseParse method), 4  
`order_index()` (eqparse.baseparse.BaseParse method), 4

## P

`p_est_main()` (eqparse.matlabparser.MatlabParser method), 6  
`parse_one()` (eqparse.xppautparser.XppautParser method), 7  
`pattern_change()` (eqparse.baseparse.BaseParse method), 4  
`pattern_write()` (eqparse.baseparse.BaseParse method), 4  
`printarray()` (eqparse.xppautparser.XppautParser method), 7

## R

`rasterise_dict_modifiers()` (eqparse.baseparse.BaseParse method), 4  
`replace_operations()` (eqparse.baseparse.BaseParse method), 5

## S

`search_and_replace()` (eqparse.baseparse.BaseParse method), 5  
`set_directory()` (eqparse.createlibrary.CreateLibrary method), 6  
`set_temp_ic_name()` (eqparse.xppautparser.XppautParser method), 7  
`solve_single()` (eqparse.matlabparser.MatlabParser method), 6  
`start()` (eqparse.timer.Timer method), 7  
`supp_initialise_data()` (eqparse.matlabparser.MatlabParser method), 6  
`supp_input_ic()` (eqparse.matlabparser.MatlabParser method), 6

## T

`table` (eqparse.createlibrary.CreateLibrary attribute), 6  
`Timer` (class in eqparse.timer), 7  
`truncate_ics()` (eqparse.xppautparser.XppautParser method), 7

## V

`vectorise_name()` (eqparse.baseparse.BaseParse method), 5  
`vectorised_to_readable()` (eqparse.matlabparser.MatlabParser method), 6

## W

`write()` (eqparse.baseparse.BaseParse method), 5  
`write_comment()` (eqparse.baseparse.BaseParse method), 5

## X

`xpp_search_and_define()` (eqparse.xppautparser.XppautParser method), 7  
`XppautParser` (class in eqparse.xppautparser), 7