
pytransport Documentation

Release 1.3.0

Royal Holloway

Feb 27, 2019

CONTENTS

1	Licence & Disclaimer	3
1.1	Licence	3
2	Authorship	5
3	Installation	7
3.1	Requirements	7
3.2	Installation	7
4	Conversion	9
5	Module Contents	11
5.1	pytransport.Data module	11
5.2	pytransport.Elements module	12
5.3	pytransport.Reader module	13
5.4	pytransport._General module	13
6	Indices and tables	15
	Python Module Index	17
	Index	19

pytransport is a set of classes and functions to load MADX output as well as prepare Transport models. The package overall functions as a holder for any code required to load and manipulate Transport output data.

LICENCE & DISCLAIMER

pytransport copyright (c) Royal Holloway, University of London, 2017. All rights reserved.

1.1 Licence

This software is provided “AS IS” and any express or limit warranties, including, but not limited to, implied warranties of merchantability, of satisfactory quality, and fitness for a particular purpose or use are disclaimed. In no event shall Royal Holloway, University of London be liable for any direct, indirect, incidental, special, exemplary, or consequential damages arising in any way out of the use of this software, even if advised of the possibility of such damage.

AUTHORSHIP

The following people have contributed to pytransport:

- William Shields
- Jochem Snuerink
- Laurie Nevay
- Stuart Walker

INSTALLATION

3.1 Requirements

- pytransport was developed for the Python 2.7 series.

pytransport depends on the following Python packages not included with Python:

- matplotlib
- numpy
- scipy
- pymadx
- pybdsim

3.2 Installation

A *setup.py* file required for a correct python installation is currently under development.

Currently, we recommend the user clones the source repository and exports the parent directory to their PYTHONPATH environmental variable. This will allow Python to find pytransport.:

```
pwd
/Users/nevay/physics/refs
git clone http://bitbucket.org/jairhul/pytransport
ls
> pytransport
export PYTHONPATH=/Users/nevay/physics/refs

python
>>> import pytransport # no errors!
```


CONVERSION

Conversion.

MODULE CONTENTS

This documentation is automatically generated by scanning all the source code. Parts may be incomplete.

pytransport - Royal Holloway utility to manipulate TRANSPORT data and models.

Authors:

- William Shields
- Jochem Snuverink

Copyright Royal Holloway, University of London 2019.

5.1 pytransport.Data module

Data

Data containers used in converting from Transport to gmad or madx.

Classes: BDSData - a list of data read from Transport files. ConversionData - a class for holding data during conversion.

class pytransport.Data.BDSData (*args, **kwargs)

Bases: list

General class representing simple 2 column data.

Inherits python list. It's a list of tuples with extra columns of 'name' and 'units'.

ConcatenateMachine (*args)

This is used to concatenate machines.

Filter (booleanarray)

Filter the data with a booleanarray. Where true, will return that event in the data.

Return type is BDSData

GetColumn (columnstring)

Return a numpy array of the values in columnstring in order as they appear in the beamline

GetItemTuple (index)

Get a specific entry in the data as a tuple of values rather than a dictionary.

IndexFromNearestS (S)

IndexFromNearestS(S)

return the index of the beamline element closest to S

Only works if "SStart" column exists in data

MatchValue (*parametername, matchvalue, tolerance*)

This is used to filter the instance of the class based on matching a parameter withing a certain tolerance.

```
>>> a = pytransport.Data.Load("myfile.txt")
>>> a.MatchValue("S", 0.3, 0.0004)
```

this will match the “S” variable in instance “a” to the value of 0.3 within +- 0.0004.

You can therefore used to match any parameter.

Return type is BDSAsciiData

MergeDuplicatesAtSameS ()

Merge duplicate entries at the same s position. This is to prevent having multiple entries at the same s in situations such as poleface rotations. Will merge zero-length items into finite length items.

NameFromNearestS (S)

```
class pytransport.Data.ConversionData (inputfile, machine, options=None, particle='proton',
                                         debug=False, distrType='gauss', gmad=True,
                                         gmadDir='gmad', madx=False, madxDir='madx',
                                         auto=True, dontSplit=False, keepName=False,
                                         combineDrifts=False, outlog=True)
```

Class used as data container object in Transport2Gmad / Transport2Madx conversion. Required input: - input-file: string, inputfile name - machine: either pybdsim.Builder.Machine or pymadx.Builder.Machine instance.

Note: if used as a holder for conversion to gmad, options must be supplied a pybdsim.Options.Options instance.

This class will hold ALL conversion related data, some stored in member variables which are separate containers for: conversion related properties (user input arguments), beam properties, and machine properties.

AddBeam ()

Function to prepare the beam and add to the machine.

AddOptions ()

Function to set the Options for a BDSIM machine.

ResetMachine ()

Delete the machine and set to be the empty machine copied at class instantiation.

5.2 pytransport.Elements module

Functions to help convert elements from Transport to gmad/madx. Code inherited in classes in pybdsim.Convert.Transport2Gmad and pymadx.Convert.Transport2Madx.

Classes: Elements - a class used to convert different element types.

```
class pytransport.Elements.Elements (transportData)
```

Class for converting different Transport element types into gmad/madx format.

Required: transportData, dtype = pytransport.Data.ConversionData

Acceleration (*linedict*)

A Function that writes the properties of an acceleration element Only RF added for gmad, not for madx!

ChangeBend (*linedict*)

Function to change the direction of the dipole bend. Can be a direction other than horizontal (i.e != n*pi).

Collimator (*linedict*)

A Function that writes the properties of a collimator element Only added for gmad, not for madx!

Correction (*linedict*)**DefineBeam** (*linedict*)**Dipole** (*linedict*)**Drift** (*linedict*)**Printline** (*linedict*)**Quadrupole** (*linedict*)**Sextupole** (*linedict*)**Solenoid** (*linedict*)**SpecialInput** (*linedict*)**TransformUpdate** (*linedict*)**UnitChange** (*linedict*)

Function to change the units (scaling) of various parameters.

5.3 pytransport.Reader module

Reader

Readers for loading Transport input files and output files. Can extract the individual lattice, optics, and fitting sections.

Classes: Reader - a list of data read from Transport files. ConversionData - a class for holding data during conversion.

`pytransport.Reader.GetFitsSection (inputFile)`

Function to get the fit routine data from the standard transport output. Returns two lists, the first with the direct output from the fitting data, the second with the first line of each element in the output data, which contains the element parameters with their fitted values.

`pytransport.Reader.GetLattice (inputFile)`

Function to extract the lattice from a standard output file.

`pytransport.Reader.GetOptics (inputFile, inputType=None)`

Extract the optics from a Transport output file.

`pytransport.Reader.GetResultsFromFitting (inputFile)`

5.4 pytransport._General module

General utilities for day to day housekeeping

Classes: _Writer - a class used for writing any output during conversion.

`pytransport._General.CheckDirExists (directory)``pytransport._General.CheckIsAddition (line, filetype='input')`

Function to check if a BEAM line of TRANSPORT code is a beam definition or r.m.s addition.

`pytransport._General.CheckIsOutput (inputfile)`

Function to check if a file is a standard TRANSPORT output file. Based upon existence of the lines:

```
"0   XXX"
```

being present, which represents the TRANSPORT indicator card line. X can be 0, 1, 2. Default is 0.

`pytransport._General.CheckIsSentinel (line)`

`pytransport._General.CheckSingleLineOutputApplied (inputfile)`

Function to check if the control element that print element output in a single line was successfully applied. Check needed as not all versions of TRANSPORT can run this type code.

`pytransport._General.ConvertBunchLength (transport, bunch_length)`

Function to convert bunch length unit in TRANSPORT into seconds.

`pytransport._General.FindEndOfLine (line)`

`pytransport._General.GetComment (line)`

Function to extract a comment from a line. Will only find one comment per line.

`pytransport._General.GetElementData (line)`

`pytransport._General.GetFaceRotationAngles (data, linenum)`

`pytransport._General.GetIndicator (data)`

Function to read the indicator number. Must be 0, 1, or 2, where: 0 is a new lattice 1 is for fitting with the first lattice (from a 0 indicator file) 2 if for a second fitting which suppresses the first fitting.

`pytransport._General.GetLabel (line)`

Function to get element label from code line.

`pytransport._General.GetPreamble (data)`

Function to read any preamble at the start of the TRANSPORT file.

`pytransport._General.GetTypeNum (line)`

Function to extract the element type number (type code). Written because element types can contain alphabetical characters when fits are used, e.g: 5.0A. Only the number is required, the use of fitting does not need to be known.

`pytransport._General.JoinSplitLines (linenum, lattice)`

`pytransport._General.OutputFitsToRegistry (transport, outputdata)`

`pytransport._General.ProcessFits (fits)`

`pytransport._General.RemoveFileExt (inputfile)`

Remove the file extension from the input file name. Only works on known extensions.

`pytransport._General.RemoveIllegals (line)`

Function to remove " and stray characters from lines.

`pytransport._General.RemoveLabel (line)`

Function to remove the label from a line.

`pytransport._General.RemoveSpaces (line)`

`pytransport._General.ScaleToMeters (transport, quantity)`

Function to scale quantity (string) to meters, returns conversion factor.

`pytransport._General.UpdateEnergyFromMomentum (transport, momentum)`

Function to calculate (from momentum): Total Energy Kinetic Energy Momentum Lorentz factor (gamma) Velocity (beta) Magnetic rigidity (brho)

`pytransport._General.UpdateMomentumFromEnergy (transport, k_energy)`

Function to calculate (from kinetic energy): Total Energy Kinetic Energy Momentum Lorentz factor (gamma) Velocity (beta) Magnetic rigidity (brho)

INDICES AND TABLES

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

PYTHON MODULE INDEX

p

- `pytransport`, [11](#)
- `pytransport._General`, [13](#)
- `pytransport.Data`, [11](#)
- `pytransport.Elements`, [12](#)
- `pytransport.Reader`, [13](#)

A

Acceleration() (pytransport.Elements.Elements method), 12
 AddBeam() (pytransport.Data.ConversionData method), 12
 AddOptions() (pytransport.Data.ConversionData method), 12

B

BDSDData (class in pytransport.Data), 11

C

ChangeBend() (pytransport.Elements.Elements method), 12
 CheckDirExists() (in module pytransport._General), 13
 CheckIsAddition() (in module pytransport._General), 13
 CheckIsOutput() (in module pytransport._General), 13
 CheckIsSentinel() (in module pytransport._General), 14
 CheckSingleLineOutputApplied() (in module pytransport._General), 14
 Collimator() (pytransport.Elements.Elements method), 12
 ConcatenateMachine() (pytransport.Data.BDSData method), 11
 ConversionData (class in pytransport.Data), 12
 ConvertBunchLength() (in module pytransport._General), 14
 Correction() (pytransport.Elements.Elements method), 12

D

DefineBeam() (pytransport.Elements.Elements method), 13
 Dipole() (pytransport.Elements.Elements method), 13
 Drift() (pytransport.Elements.Elements method), 13

E

Elements (class in pytransport.Elements), 12

F

Filter() (pytransport.Data.BDSData method), 11
 FindEndOfLine() (in module pytransport._General), 14

G

GetColumn() (pytransport.Data.BDSData method), 11
 GetComment() (in module pytransport._General), 14
 GetElementData() (in module pytransport._General), 14
 GetFaceRotationAngles() (in module pytransport._General), 14
 GetFitsSection() (in module pytransport.Reader), 13
 GetIndicator() (in module pytransport._General), 14
 GetItemTuple() (pytransport.Data.BDSData method), 11
 GetLabel() (in module pytransport._General), 14
 GetLattice() (in module pytransport.Reader), 13
 GetOptics() (in module pytransport.Reader), 13
 GetPreamble() (in module pytransport._General), 14
 GetResultsFromFitting() (in module pytransport.Reader), 13
 GetTypeNum() (in module pytransport._General), 14

I

IndexFromNearestS() (pytransport.Data.BDSData method), 11

J

JoinSplitLines() (in module pytransport._General), 14

M

MatchValue() (pytransport.Data.BDSData method), 11
 MergeDuplicatesAtSameS() (pytransport.Data.BDSData method), 12

N

`NameFromNearestS()` (*pytransport.Data.BDSData method*), 12

O

`OutputFitsToRegistry()` (*in module pytransport._General*), 14

P

`Printline()` (*pytransport.Elements.Elements method*), 13

`ProcessFits()` (*in module pytransport._General*), 14

`pytransport` (*module*), 11

`pytransport._General` (*module*), 13

`pytransport.Data` (*module*), 11

`pytransport.Elements` (*module*), 12

`pytransport.Reader` (*module*), 13

Q

`Quadrupole()` (*pytransport.Elements.Elements method*), 13

R

`RemoveFileExt()` (*in module pytransport._General*), 14

`RemoveIllegals()` (*in module pytransport._General*), 14

`RemoveLabel()` (*in module pytransport._General*), 14

`RemoveSpaces()` (*in module pytransport._General*), 14

`ResetMachine()` (*pytransport.Data.ConversionData method*), 12

S

`ScaleToMeters()` (*in module pytransport._General*), 14

`Sextupole()` (*pytransport.Elements.Elements method*), 13

`Solenoid()` (*pytransport.Elements.Elements method*), 13

`SpecialInput()` (*pytransport.Elements.Elements method*), 13

T

`TransformUpdate()` (*pytransport.Elements.Elements method*), 13

U

`UnitChange()` (*pytransport.Elements.Elements method*), 13

`UpdateEnergyFromMomentum()` (*in module pytransport._General*), 14

`UpdateMomentumFromEnergy()` (*in module pytransport._General*), 14