
DAE Tools Project Documentation

Release 1.3.1

Dragan Nikolic

April 22, 2013

CONTENTS

1	Module pyCore	1
1.1	Overview	1
1.2	Autodifferentiation and equation evaluation tree support	1
1.3	Modelling concepts	4
2	Module pyActivity	23
2.1	Overview	23
3	Module pyDataReporting	27
3.1	Overview	27
4	Module pyIDAS	29
4.1	Overview	29
5	Module pyUnits	31
5.1	Overview	31
5.2	Classes	31
6	Indices and tables	33
	Python Module Index	35
	Index	37

MODULE PYCORE

1.1 Overview

Trt mrt.

1.2 Autodifferentiation and equation evaluation tree support

1.2.1 Classes

<code>adouble</code>	Class <code>adouble</code> operates on values/derivatives of domains, parameters and variables.
<code>adouble_array</code>	Class <code>adouble_array</code> operates on arrays of values/derivatives of domains, parameters and variables.
<code>daeCondition</code>	

class `pyCore.adouble`

Bases: `Boost.Python.instance`

Class `adouble` operates on values/derivatives of domains, parameters and variables. It supports basic mathematical operators (+, -, , /, *), comparison operators (<, <=, >, >=, ==, !=), and logical operators (and, or, not). Operands can be instances of `adouble` or float values.

Derivative

Derivative

GatherInfo

Internally used by the framework.

Node

Contains the equation evaluation node.

Value

Value

class `pyCore.adouble_array`

Bases: `Boost.Python.instance`

Class `adouble_array` operates on arrays of values/derivatives of domains, parameters and variables. It supports basic mathematical operators (+, -, , /, *). Operands can be instances of `adouble_array`, `adouble` or float values.

`__len__` ((`adouble_array`)*self*) → int :

Returns the size of the `adouble_array` object.

__getitem__ ((*adouble_array*)*self*, (*int*)*index*) → *adouble* :

Gets an *adouble* object at the specified index.

__setitem__ ((*adouble_array*)*self*, (*int*)*index*, (*adouble*)*value*) → *None* :

Sets an *adouble* object at the specified index.

GatherInfo

Used internally by the framework.

Node

Contains the equation evaluation node.

Resize ((*adouble_array*)*self*, (*int*)*newSize*) → *None* :

Resizes the *adouble_array* object to the new size.

items ((*object*)*arg1*) → *object* :

Returns an iterator over *adouble* items in *adouble_array* object.

class `pyCore.daeCondition`

Bases: `Boost.Python.instance`

__or__ ((*daeCondition*)*self*, (*daeCondition*)*right*) → *daeCondition*

Logical operator `or`

__and__ ((*daeCondition*)*self*, (*daeCondition*)*right*) → *daeCondition*

Logical operator `and`

EventTolerance

Expressions

RuntimeNode

SetupNode

1.2.2 Mathematical functions

<code>Exp</code>	<code>Exp((adouble_array)arg1) -> adouble_array</code>
<code>Log</code>	<code>Log((adouble_array)arg1) -> adouble_array</code>
<code>Log10</code>	<code>Log10((adouble_array)arg1) -> adouble_array</code>
<code>Sqrt</code>	<code>Sqrt((adouble_array)arg1) -> adouble_array</code>
<code>Sin</code>	<code>Sin((adouble_array)arg1) -> adouble_array</code>
<code>Cos</code>	<code>Cos((adouble_array)arg1) -> adouble_array</code>
<code>Tan</code>	<code>Tan((adouble_array)arg1) -> adouble_array</code>
<code>ASin</code>	<code>ASin((adouble_array)arg1) -> adouble_array</code>
<code>ACos</code>	<code>ACos((adouble_array)arg1) -> adouble_array</code>
<code>ATan</code>	<code>ATan((adouble_array)arg1) -> adouble_array</code>
<code>Sinh</code>	
<code>Cosh</code>	
<code>Tanh</code>	
<code>ASinh</code>	
<code>ACosh</code>	
<code>ATanh</code>	
<code>ATan2</code>	
<code>Ceil</code>	<code>Ceil((adouble_array)arg1) -> adouble_array</code>
<code>Floor</code>	<code>Floor((adouble_array)arg1) -> adouble_array</code>

Continued on next page

Table 1.2 – continued from previous page

Pow	Pow((adouble)arg1, (adouble)arg2) -> adouble
Abs	Abs((adouble_array)arg1) -> adouble_array
Min	Min((float)arg1, (adouble)arg2) -> adouble
Max	Max((float)arg1, (adouble)arg2) -> adouble

```

pyCore.Exp ((adouble)arg1) → adouble
    Exp( (adouble_array)arg1) -> adouble_array

pyCore.Log ((adouble)arg1) → adouble
    Log( (adouble_array)arg1) -> adouble_array

pyCore.Log10 ((adouble)arg1) → adouble
    Log10( (adouble_array)arg1) -> adouble_array

pyCore.Sqrt ((adouble)arg1) → adouble
    Sqrt( (adouble_array)arg1) -> adouble_array

pyCore.Sin ((adouble)arg1) → adouble
    Sin( (adouble_array)arg1) -> adouble_array

pyCore.Cos ((adouble)arg1) → adouble
    Cos( (adouble_array)arg1) -> adouble_array

pyCore.Tan ((adouble)arg1) → adouble
    Tan( (adouble_array)arg1) -> adouble_array

pyCore.ASin ((adouble)arg1) → adouble
    ASin( (adouble_array)arg1) -> adouble_array

pyCore.ACos ((adouble)arg1) → adouble
    ACos( (adouble_array)arg1) -> adouble_array

pyCore.ATan ((adouble)arg1) → adouble
    ATan( (adouble_array)arg1) -> adouble_array

pyCore.Sinh ((adouble)arg1) → adouble

pyCore.Cosh ((adouble)arg1) → adouble

pyCore.Tanh ((adouble)arg1) → adouble

pyCore.ASinh ((adouble)arg1) → adouble

pyCore.ACosh ((adouble)arg1) → adouble

pyCore.ATanh ((adouble)arg1) → adouble

pyCore.ATan2 ((adouble)arg1, (adouble)arg2) → adouble

pyCore.Ceil ((adouble)arg1) → adouble
    Ceil( (adouble_array)arg1) -> adouble_array

pyCore.Floor ((adouble)arg1) → adouble
    Floor( (adouble_array)arg1) -> adouble_array

pyCore.Pow ((adouble)arg1, (float)arg2) → adouble
    Pow( (adouble)arg1, (adouble)arg2) -> adouble
    Pow( (float)arg1, (adouble)arg2) -> adouble

pyCore.Abs ((adouble)arg1) → adouble
    Abs( (adouble_array)arg1) -> adouble_array

```

```
pyCore.Min ((adouble)arg1, (adouble)arg2) → adouble
    Min( (float)arg1, (adouble)arg2) -> adouble
    Min( (adouble)arg1, (float)arg2) -> adouble
    Min( (adouble_array)adarray) -> adouble

pyCore.Max ((adouble)arg1, (adouble)arg2) → adouble
    Max( (float)arg1, (adouble)arg2) -> adouble
    Max( (adouble)arg1, (float)arg2) -> adouble
    Max( (adouble_array)adarray) -> adouble
```

1.3 Modelling concepts

...

1.3.1 Enumerations

daeeDomainType
daeeParameterType
daeePortType
daeeDiscretizationMethod
daeeDomainBounds
daeeInitialConditionMode
daeeDomainIndexType
daeeRangeType
daeIndexRangeType
daeeOptimizationVariableType
daeeModelLanguage
daeeConstraintType
daeeUnaryFunctions
daeeBinaryFunctions
daeeSpecialUnaryFunctions
daeeLogicalUnaryOperator
daeeLogicalBinaryOperator
daeeConditionType
daeeActionType
daeeEquationType
daeeModelType

```
class pyCore.daeeDomainType
    Bases: Boost.Python.enum

    eArray = pyCore.daeeDomainType.eArray
    eDTUnknown = pyCore.daeeDomainType.eDTUnknown
    eDistributed = pyCore.daeeDomainType.eDistributed

class pyCore.daeeParameterType
    Bases: Boost.Python.enum
```



```
eBool = pyCore.daeParameterType.eBool
eInteger = pyCore.daeParameterType.eInteger
ePTUnknown = pyCore.daeParameterType.ePTUnknown
eReal = pyCore.daeParameterType.eReal

class pyCore.daePortType
    Bases: Boost.Python.enum

    eInletPort = pyCore.daePortType.eInletPort
    eOutletPort = pyCore.daePortType.eOutletPort
    eUnknownPort = pyCore.daePortType.eUnknownPort

class pyCore.daeDiscretizationMethod
    Bases: Boost.Python.enum

    eBFDM = pyCore.daeDiscretizationMethod.eBFDM
    eCFDM = pyCore.daeDiscretizationMethod.eCFDM
    eCustomDM = pyCore.daeDiscretizationMethod.eCustomDM
    eDMUnknown = pyCore.daeDiscretizationMethod.eDMUnknown
    eFFDM = pyCore.daeDiscretizationMethod.eFFDM

class pyCore.daeDomainBounds
    Bases: Boost.Python.enum

    eClosedClosed = pyCore.daeDomainBounds.eClosedClosed
    eClosedOpen = pyCore.daeDomainBounds.eClosedOpen
    eDBUnknown = pyCore.daeDomainBounds.eDBUnknown
    eLowerBound = pyCore.daeDomainBounds.eLowerBound
    eOpenClosed = pyCore.daeDomainBounds.eOpenClosed
    eOpenOpen = pyCore.daeDomainBounds.eOpenOpen
    eUpperBound = pyCore.daeDomainBounds.eUpperBound

class pyCore.daeInitialConditionMode
    Bases: Boost.Python.enum

    eAlgebraicValuesProvided = pyCore.daeInitialConditionMode.eAlgebraicValuesProvided
    eDifferentialValuesProvided = pyCore.daeInitialConditionMode.eDifferentialValuesProvided
    eICTUnknown = pyCore.daeInitialConditionMode.eICTUnknown
    eQuasySteadyState = pyCore.daeInitialConditionMode.eQuasySteadyState

class pyCore.daeDomainIndexType
    Bases: Boost.Python.enum

    eConstantIndex = pyCore.daeDomainIndexType.eConstantIndex
    eDITUnknown = pyCore.daeDomainIndexType.eDITUnknown
    eDomainIterator = pyCore.daeDomainIndexType.eDomainIterator
    eIncrementedDomainIterator = pyCore.daeDomainIndexType.eIncrementedDomainIterator
```

```
class pyCore.daeRangeType
    Bases: Boost.Python.enum

    eRaTUnknown = pyCore.daeRangeType.eRaTUnknown
    eRange = pyCore.daeRangeType.eRange
    eRangeDomainIndex = pyCore.daeRangeType.eRangeDomainIndex

class pyCore.daeIndexRangeType
    Bases: Boost.Python.enum

    eAllPointsInDomain = pyCore.daeIndexRangeType.eAllPointsInDomain
    eCustomRange = pyCore.daeIndexRangeType.eCustomRange
    eIRTUnknown = pyCore.daeIndexRangeType.eIRTUnknown
    eRangeOfIndexes = pyCore.daeIndexRangeType.eRangeOfIndexes

class pyCore.daeOptimizationVariableType
    Bases: Boost.Python.enum

    eBinaryVariable = pyCore.daeOptimizationVariableType.eBinaryVariable
    eContinuousVariable = pyCore.daeOptimizationVariableType.eContinuousVariable
    eIntegerVariable = pyCore.daeOptimizationVariableType.eIntegerVariable

class pyCore.daeModelLanguage
    Bases: Boost.Python.enum

    eCDAE = pyCore.daeModelLanguage.eCDAE
    eMLNone = pyCore.daeModelLanguage.eMLNone
    ePYDAE = pyCore.daeModelLanguage.ePYDAE

class pyCore.daeConstraintType
    Bases: Boost.Python.enum

    eEqualityConstraint = pyCore.daeConstraintType.eEqualityConstraint
    eInequalityConstraint = pyCore.daeConstraintType.eInequalityConstraint

class pyCore.daeUnaryFunctions
    Bases: Boost.Python.enum

    eAbs = pyCore.daeUnaryFunctions.eAbs
    eArcCos = pyCore.daeUnaryFunctions.eArcCos
    eArcSin = pyCore.daeUnaryFunctions.eArcSin
    eArcTan = pyCore.daeUnaryFunctions.eArcTan
    eCeil = pyCore.daeUnaryFunctions.eCeil
    eCos = pyCore.daeUnaryFunctions.eCos
    eExp = pyCore.daeUnaryFunctions.eExp
    eFloor = pyCore.daeUnaryFunctions.eFloor
    eLn = pyCore.daeUnaryFunctions.eLn
    eLog = pyCore.daeUnaryFunctions.eLog
    eSign = pyCore.daeUnaryFunctions.eSign
```

```
eSin = pyCore.daeUnaryFunctions.eSin
eSqrt = pyCore.daeUnaryFunctions.eSqrt
eTan = pyCore.daeUnaryFunctions.eTan
eUUnknown = pyCore.daeUnaryFunctions.eUUnknown

class pyCore.daeBinaryFunctions
    Bases: Boost.Python.enum

    eBUnknown = pyCore.daeBinaryFunctions.eBUnknown
    eDivide = pyCore.daeBinaryFunctions.eDivide
    eMax = pyCore.daeBinaryFunctions.eMax
    eMin = pyCore.daeBinaryFunctions.eMin
    eMinus = pyCore.daeBinaryFunctions.eMinus
    eMulti = pyCore.daeBinaryFunctions.eMulti
    ePlus = pyCore.daeBinaryFunctions.ePlus
    ePower = pyCore.daeBinaryFunctions.ePower

class pyCore.daeSpecialUnaryFunctions
    Bases: Boost.Python.enum

    eAverage = pyCore.daeSpecialUnaryFunctions.eAverage
    eMaxInArray = pyCore.daeSpecialUnaryFunctions.eMaxInArray
    eMinInArray = pyCore.daeSpecialUnaryFunctions.eMinInArray
    eProduct = pyCore.daeSpecialUnaryFunctions.eProduct
    eSUUnknown = pyCore.daeSpecialUnaryFunctions.eSUUnknown
    eSum = pyCore.daeSpecialUnaryFunctions.eSum

class pyCore.daeLogicalUnaryOperator
    Bases: Boost.Python.enum

    eNot = pyCore.daeLogicalUnaryOperator.eNot
    eUUnknown = pyCore.daeLogicalUnaryOperator.eUUnknown

class pyCore.daeLogicalBinaryOperator
    Bases: Boost.Python.enum

    eAnd = pyCore.daeLogicalBinaryOperator.eAnd
    eBUnknown = pyCore.daeLogicalBinaryOperator.eBUnknown
    eOr = pyCore.daeLogicalBinaryOperator.eOr

class pyCore.daeConditionType
    Bases: Boost.Python.enum

    eCTUnknown = pyCore.daeConditionType.eCTUnknown
    eEQ = pyCore.daeConditionType.eEQ
    eGT = pyCore.daeConditionType.eGT
    eGTEQ = pyCore.daeConditionType.eGTEQ
    eLT = pyCore.daeConditionType.eLT
```

```

    eLTEQ = pyCore.daeConditionType.eLTEQ
    eNotEQ = pyCore.daeConditionType.eNotEQ
class pyCore.daeActionType
    Bases: Boost.Python.enum

    eChangeState = pyCore.daeActionType.eChangeState
    eReAssignOrReInitializeVariable = pyCore.daeActionType.eReAssignOrReInitializeVariable
    eSendEvent = pyCore.daeActionType.eSendEvent
    eUnknownAction = pyCore.daeActionType.eUnknownAction
    eUserDefinedAction = pyCore.daeActionType.eUserDefinedAction
class pyCore.daeEquationType
    Bases: Boost.Python.enum

    eAlgebraic = pyCore.daeEquationType.eAlgebraic
    eETUnknown = pyCore.daeEquationType.eETUnknown
    eExplicitODE = pyCore.daeEquationType.eExplicitODE
    eImplicitODE = pyCore.daeEquationType.eImplicitODE
class pyCore.daeModelType
    Bases: Boost.Python.enum

    eDAE = pyCore.daeModelType.eDAE
    eMTUnknown = pyCore.daeModelType.eMTUnknown
    eODE = pyCore.daeModelType.eODE
    eSteadyState = pyCore.daeModelType.eSteadyState

```

1.3.2 Classes

daeVariableType	
daeDomain	
daeParameter	
daeVariable	
daeModel	Base model class.
daeSTN	
daeIF	
daeEquation	
daeState	
daeStateTransition	
daePort	
daeEventPort	
daePortConnection	
daeScalarExternalFunction	
daeVectorExternalFunction	
daeDomainIndex	
daeIndexRange	
daeArrayRange	
Continued on next page	

Table 1.4 – continued from previous page

<code>daeDEDI</code>
<code>daeAction</code>
<code>daeOptimizationVariable</code>
<code>daeObjectiveFunction</code>
<code>daeOptimizationConstraint</code>
<code>daeMeasuredVariable</code>
<code>daeEquationExecutionInfo</code>

class `pyCore.daeVariableType`

Bases: `Boost.Python.instance`

`__init__` *((object)arg1)* → None

`__init__` *((object)self, (str)name, (object)units, (float)lowerBound, (float)upperBound, (float)initialGuess, (float)absTolerance)* → None

AbsoluteTolerance

InitialGuess

LowerBound

Name

Units

UpperBound

class `pyCore.daeObject`

Bases: `Boost.Python.instance`

CanonicalName

Description

GetNameRelativeToParentModel *((daeObject)self)* → str

GetStrippedName *((daeObject)self)* → str

GetStrippedNameRelativeToParentModel *((daeObject)self)* → str

ID

Library

Model

Name

Version

class `pyCore.daeDomain`

Bases: `pyCore.daeObject`

`__init__` *((object)arg1)* → None

`__init__` *((object)self, (str)name, (daeModel)parentModel, (object)units [, (str)description=''])* → None

`__init__` *((object)self, (str)name, (daePort)parentPort, (object)units [, (str)description=''])* → None

`__getitem__` *((daeDomain)self, (int)index)* → adouble

`__call__` *((daeDomain)self, (int)index)* → adouble

CreateArray *((daeDomain)self, (int)noIntervals)* → None

```
CreateDistributed ((daeDomain)self, (daeDiscretizationMethod)discretizationMethod,  
                    (int)discretizationOrder, (int)numberOfIntervals, (float)lowerBound,  
                    (float)upperBound) → None
```

DiscretizationMethod

DiscretizationOrder

LowerBound

NumberOfIntervals

NumberOfPoints

Points

Type

Units

UpperBound

numpyPoints

```
class pyCore.daeParameter
```

Bases: `pyCore.daeObject`

```
GetValue ((daeParameter)self[, (int)index1[, ...[, (int)index8 ] ]]) → float
```

Gets the value of the parameter at the specified domain indexes. How many arguments `index1`, ..., `index8` are used depends on the number of domains that the parameter is distributed on.

```
GetQuantity ((daeParameter)self[, (int)index1[, ...[, (int)index8 ] ]]) → quantity
```

Gets the value of the parameter at the specified domain indexes as the `quantity` object (with value and units). How many arguments `index1`, ..., `index8` are used depends on the number of domains that the parameter is distributed on.

```
SetValue ((daeParameter)self[, (int)index1[, ...[, (int)index8 ] ]], (object)value) → None
```

Sets the value of the parameter at the specified domain indexes (as `float` or `quantity`). How many arguments `index1`, ..., `index8` are used depends on the number of domains that the parameter is distributed on.

```
SetValues ((daeParameter)self, (float)values) → None
```

Sets all values of the parameter (as `float` or `quantity`).

```
array ((daeParameter)self[, (object)index1[, ...[, (object)index8 ] ]]) → adouble_array
```

Gets the array of parameter's values at the specified domain indexes (used to build equation residuals only). How many arguments `index1`, ..., `index8` are used depends on the number of domains that the parameter is distributed on. Argument types can be one of the following:

- `daeIndexRange` object
- plain integer (to select a single index from a domain)
- python `list` (to select a list of indexes from a domain)
- python `slice` (to select a range of indexes from a domain: `start_index`, `end_index`, `step`)
- character `'*'` (to select all points from a domain)
- integer `-1` (to select all points from a domain)
- empty python list `[]` (to select all points from a domain)

```
__call__ ((daeParameter)self[, (int)index1[, ...[, (int)index8 ] ]]) → adouble
```

Gets the value of the parameter at the specified domain indexes (used to build equation residuals only).

How many arguments `index1`, ..., `index8` are used depends on the number of domains that the parameter is distributed on.

DistributeOnDomain `((daeParameter)self, (daeDomain)domain) → None`

Domains

GetDomainsIndexesMap `((daeParameter)self, (int)indexBase) → dict`

NumberOfPoints

ReportingOn

Units

numpyValues

class `pyCore.daeVariable`

Bases: `pyCore.daeObject`

GetValue `((daeVariable)self[, (int)index1[, ...[, (int)index8]]]) → float`

Gets the value of the variable at the specified domain indexes. How many arguments `index1`, ..., `index8` are used depends on the number of domains that the variable is distributed on.

GetQuantity `((daeVariable)self[, (int)index1[, ...[, (int)index8]]]) → quantity`

Gets the value of the variable at the specified domain indexes as the `quantity` object (with value and units). How many arguments `index1`, ..., `index8` are used depends on the number of domains that the variable is distributed on.

SetValue `((daeVariable)self[, (int)index1[, ...[, (int)index8]]], (object)value) → None`

Sets the value of the variable at the specified domain indexes (as `float` or `quantity`). How many arguments `index1`, ..., `index8` are used depends on the number of domains that the variable is distributed on.

SetValues `((daeVariable)self, (object)values) → None`

Sets all values of the variable (as `float` or `quantity`).

AssignValue `((daeVariable)self[, (int)index1[, ...[, (int)index8]]], (object)value) → None`

AssignValues `((daeVariable)self, (object)values) → None`

ReAssignValue `((daeVariable)self[, (int)index1[, ...[, (int)index8]]], (object)value) → None`

ReAssignValues `((daeVariable)self, (object)values) → None`

SetInitialCondition `((daeVariable)self[, (int)index1[, ...[, (int)index8]]], (object)initialCondition) → None`

SetInitialConditions `((daeVariable)self, (object)initialConditions) → None`

ReSetInitialCondition `((daeVariable)self[, (int)index1[, ...[, (int)index8]]], (object)initialCondition) → None`

ReSetInitialConditions `((daeVariable)self, (object)initialConditions) → None`

SetInitialGuess `((daeVariable)self[, (int)index1[, ...[, (int)index8]]], (object)initialGuess) → None`

SetInitialGuesses `((daeVariable)self, (object)initialGuesses) → None`

SetAbsoluteTolerances `((daeVariable)self, (object)tolerances) → None`

array `((daeVariable)self[, (object)index1[, ...[, (object)index8]]]) → adouble_array`

Gets the array of variable's values at the specified domain indexes (used to build equation residuals only). How many arguments `index1`, ..., `index8` are used depends on the number of domains that the variable is distributed on. Argument types are the same as those described in `pyCore.daeParameter.array()`

d_array ((*daeVariable*)self[, (*object*)index1[, ...[, (*object*)index8]]]) → *adouble_array*

Gets the array of partial derivatives at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on. Argument types are the same as those described in `pyCore.daeParameter.array()`.

d2_array ((*daeVariable*)self[, (*object*)index1[, ...[, (*object*)index8]]]) → *adouble_array*

Gets the array of partial derivatives of the second order at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on. Argument types are the same as those described in `pyCore.daeParameter.array()`.

dt_array ((*daeVariable*)self[, (*object*)index1[, ...[, (*object*)index8]]]) → *adouble_array*

Gets the array of time derivatives at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on. Argument types are the same as those described in `pyCore.daeParameter.array()`.

__call__ ((*daeVariable*)self[, (*int*)index1[, ...[, (*int*)index8]]]) → *adouble*

Gets the value of the variable at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on.

d ((*daeVariable*)self, (*daeDomain*)domain[, (*int*)index1[, ...[, (*int*)index8]]]) → *adouble*

Gets the partial derivative of the variable at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on.

d2 ((*daeVariable*)self, (*daeDomain*)domain[, (*int*)index1[, ...[, (*int*)index8]]]) → *adouble*

Gets the partial derivative of second order of the variable at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on.

dt ((*daeVariable*)self[, (*int*)index1[, ...[, (*int*)index8]]]) → *adouble*

Gets the time derivative of the variable at the specified domain indexes (used to build equation residuals only). How many arguments *index1*, ..., *index8* are used depends on the number of domains that the variable is distributed on.

DistributeOnDomain ((*daeVariable*)self, (*daeDomain*)domain) → *None*

Domains

GetDomainsIndexesMap ((*daeVariable*)arg1, (*int*)self) → *dict*

NumberOfPoints

OverallIndex

ReportingOn

VariableType

numpyIDs

numpyValues

class `pyCore.daeModel`

Bases: `pyCore.daeObject`

Base model class.

__init__ ((*object*)arg1) → *None*

__init__ ((object)self, (str)name [, (daeModel)parent=0 [, (str)description='']] -> None :
 Constructor...

ComponentArrays

A list of arrays of components in the model.

Components

A list of components in the model.

ConnectEventPorts ((daeModel)self, (daeEventPort)portFrom, (daeEventPort)portTo) -> None :
 Connects two event ports.

ConnectPorts ((daeModel)self, (daePort)portFrom, (daePort)portTo) -> None :
 Connects two ports.

CreateEquation ((daeModel)self, (str)name[, (str)description='[, (float)scaling=1.0]']) -> daeEquation :
 Creates a new equation. Used to add equations to models or states in state transition networks

DeclareEquations ((daeModel)self) -> None :

User-defined function where all model equations and state transition networks are declared. Must be always implemented in derived classes.

DeclareEquations((daeModel)self) -> None

Domains

A list of domains in the model.

ELSE ((daeModel)self) -> None :
 Adds the last state to a reversible state transition network.

ELSE_IF ((daeModel)self, (daeCondition)condition[, (float)eventTolerance=0.0]) -> None :
 Adds a new state to a reversible state transition network.

END_IF ((daeModel)self) -> None :
 Finalises a reversible state transition network.

END_STN ((daeModel)self) -> None :

Equations

A list of equations in the model.

EventPorts

A list of event ports in the model.

Export ((daeModel)self, (str)content, (daeModelLanguage)language, (daeModelExportContext)modelExportContext) -> None :

ExportObjects ((daeModel)self, (list)objects, (daeModelLanguage)language) -> str :

IF ((daeModel)self, (daeCondition)condition[, (float)eventTolerance=0.0]) -> None :
 Creates a reversible state transition network and adds the first state.

InitialConditionMode

A mode used to calculate initial conditions ...

IsModelDynamic

Boolean flag that determines whether the model is dynamic or steady-state.

ModelType

A type of the model ().

A list of OnEvent actions in the model.

A list of parameters in the model.

A list of arrays of ports in the model.

A list of port connections in the model.

A list of ports in the model.

A list of state transition networks in the model.

Switches the reporting of the model variables/parameters to the data reporter on or off.

A list of variables in the model.

ActiveState

```

DistributeOnDomain ((daeEquation)arg1, (daeDomain)arg2, (daeDomainBounds)arg3) →
    daeDEDI
    DistributeOnDomain((daeEquation)arg1, (daeDomain)arg2, (list)arg3) -> daeDEDI

```

DistributedEquationDomainInfos

EquationExecutionInfos

EquationType

Residual

Scaling

class `pyCore.daeState`
Bases: `pyCore.daeObject`

Equations

NestedSTNs

StateTransitions

class `pyCore.daeStateTransition`
Bases: `pyCore.daeObject`

Actions

Condition

class `pyCore.daePort`
Bases: `pyCore.daeObject`

Domains

Export `((daePort)arg1, (str)arg2, (daeModelLanguage)arg3, (daeModelExportContext)arg4) → None`

Parameters

SetReportingOn `((daePort)arg1, (bool)arg2) → None`

Type

Variables

class `pyCore.daeEventPort`
Bases: `pyCore.daeObject`

EventData

Events

ReceiveEvent `((daeEventPort)arg1, (float)arg2) → None`

RecordEvents

SendEvent `((daeEventPort)arg1, (float)arg2) → None`

Type

class `pyCore.daePortConnection`
Bases: `pyCore.daeObject`

Equations

PortFrom

PortTo

class `pyCore.daeScalarExternalFunction`
Bases: `Boost.Python.instance`

__call__ `((daeScalarExternalFunction)arg1) → adouble`

Calculate ((*daeScalarExternalFunction*)arg1, (*tuple*)arg2, (*dict*)arg3) → object
Calculate((daeScalarExternalFunction)arg1, (tuple)arg2, (dict)arg3) -> None

Name

class pyCore.**daeVectorExternalFunction**

Bases: Boost.Python.instance

__call__ ((*daeVectorExternalFunction*)arg1) → adouble_array

Calculate ((*daeVectorExternalFunction*)arg1, (*tuple*)arg2, (*dict*)arg3) → list
Calculate((daeVectorExternalFunction)arg1, (tuple)arg2, (dict)arg3) -> None

Name

class pyCore.**daeDomainIndex**

Bases: Boost.Python.instance

__init__ ((*object*)arg1) → None

__init__((object)self, (int)index) -> None

__init__((object)self, (daeDEDI)dedi) -> None

__init__((object)self, (daeDEDI)dedi, (int)increment) -> None

__init__((object)self, (daeDomainIndex)domainIndex) -> None

DEDI

Increment

Index

Type

class pyCore.**daeIndexRange**

Bases: Boost.Python.instance

__init__ ((*object*)arg1) → None

__init__((object)self, (daeDomain)domain) -> None

__init__((object)arg1, (daeDomain)arg2, (list)arg3) -> object

__init__((object)self, (daeDomain)domain, (int)startIndex, (int)endIndex, (int)step) -> None

Domain

EndIndex

NoPoints

StartIndex

Step

Type

class pyCore.**daeArrayRange**

Bases: Boost.Python.instance

__init__ ((*object*)arg1) → None

__init__((object)self, (daeDomainIndex)domainIndex) -> None

__init__((object)self, (daeIndexRange)indexRange) -> None

DomainIndex

NoPoints

Range

Type

class `pyCore.daeDEDI`

Bases: `pyCore.daeObject`

`__call__` ((*daeDEDI*)*self*) → adouble

Domain

DomainBounds

DomainPoints

class `pyCore.daeAction`

Bases: `pyCore.daeObject`

Execute ((*daeAction*)*arg1*) → None

Execute((*daeAction*)*arg1*) -> None

RuntimeNode

STN

SendEventPort

SetupNode

StateTo

Type

VariableWrapper

class `pyCore.daeOptimizationVariable`

Bases: `pyCore.daeOptimizationVariable_t`

LowerBound

Name

StartingPoint

Type

UpperBound

Value

class `pyCore.daeObjectiveFunction`

Bases: `pyCore.daeObjectiveFunction_t`

Gradients

Name

Residual

Value

class `pyCore.daeOptimizationConstraint`

Bases: `pyCore.daeOptimizationConstraint_t`

Gradients

Name

Residual

Type

Value

```
class pyCore.daeMeasuredVariable
    Bases: pyCore.daeMeasuredVariable_t
```

Gradients

Name

Residual

Value

```
class pyCore.daeEquationExecutionInfo
    Bases: Boost.Python.instance
```

EquationType

Node

VariableIndexes

1.3.3 Logging

daeLog_t
daeBaseLog
daeFileLog
daeStdOutLog
daeTCPILog
daeTCPILogServer

```
class pyCore.daeLog_t
    Bases: Boost.Python.instance

    DecreaseIndent ((daeLog_t)arg1, (int)arg2) → None
        DecreaseIndent( (daeLog_t)arg1, (int)arg2) -> None

    ETA

    Enabled

    IncreaseIndent ((daeLog_t)arg1, (int)arg2) → None
        IncreaseIndent( (daeLog_t)arg1, (int)arg2) -> None

    Indent

    IndentString

    JoinMessages ((daeLog_t)arg1, (str)arg2) → str
        JoinMessages( (daeLog_t)arg1, (str)arg2) -> None

    Message ((daeLog_t)arg1, (str)arg2, (int)arg3) → None
        Message( (daeLog_t)arg1, (str)arg2, (int)arg3) -> None

    PercentageDone

    PrintProgress

    Progress
```

```

class pyCore.daeBaseLog
    Bases: pyCore.daeLog_t

    DecreaseIndent ((daeBaseLog)arg1, (int)arg2) → None
    IncreaseIndent ((daeBaseLog)arg1, (int)arg2) → None
    Message ((daeBaseLog)arg1, (str)arg2, (int)arg3) → None
        Message( (daeBaseLog)arg1, (str)arg2, (int)arg3) -> None
    SetProgress ((daeBaseLog)arg1, (float)arg2) → None
        SetProgress( (daeBaseLog)arg1, (float)arg2) -> None

class pyCore.daeFileLog
    Bases: pyCore.daeBaseLog

    Message ((daeFileLog)arg1, (str)arg2, (int)arg3) → None
        Message( (daeFileLog)arg1, (str)arg2, (int)arg3) -> None

class pyCore.daeStdOutLog
    Bases: pyCore.daeBaseLog

    Message ((daeStdOutLog)arg1, (str)arg2, (int)arg3) → None
        Message( (daeStdOutLog)arg1, (str)arg2, (int)arg3) -> None

class pyCore.daeTCPIPLog
    Bases: pyCore.daeBaseLog

    Message ((daeTCPIPLog)arg1, (str)arg2, (int)arg3) → None
        Message( (daeTCPIPLog)arg1, (str)arg2, (int)arg3) -> None

class pyCore.daeTCPIPLogServer
    Bases: Boost.Python.instance

    MessageReceived ((daeTCPIPLogServer)arg1, (str)arg2) → None
        MessageReceived( (daeTCPIPLogServer)arg1, (str)arg2) -> None

```

1.3.4 Functions

d
dt
Time
Constant
Constant((object)value) -> adouble
Array
Sum
Product
Integral
Average

```

pyCore.d ((adouble)arg1, (daeDomain)ad) → adouble
pyCore.dt ((adouble)ad) → adouble
pyCore.Time () → adouble
pyCore.Constant ((float)value) → adouble
    Constant( (object)value) -> adouble
pyCore.Array ((list)values) → adouble_array

```

```
pyCore.Sum((adouble_array)adarray) → adouble
pyCore.Product((adouble_array)adarray) → adouble
pyCore.Integral((adouble_array)adarray) → adouble
pyCore.Average((adouble_array)adarray) → adouble
```

1.3.5 Auxiliary classes

daeVariableWrapper
daeConfig

```
class pyCore.daeVariableWrapper
    Bases: Boost.Python.instance
    __init__ ((object)self, (daeVariable)variable[, (str)name='']) → None
    __init__ ((object)self, (adouble)ad[, (str)name='']) -> None

    DomainIndexes
    Name
    OverallIndex
    Value
    Variable
    VariableType

class pyCore.daeConfig
    Bases: Boost.Python.instance
    __contains__ ((daeConfig)self, (object)propertyPath) → object
    __getitem__ ((daeConfig)self, (object)propertyPath) → object
    __setitem__ ((daeConfig)self, (object)propertyPath, (object)value) → None
    GetBoolean ((daeConfig)self, (str)propertyPath[, (bool)defaultValue]) → bool
    GetFloat ((daeConfig)self, (str)propertyPath[, (float)defaultValue]) → float
    GetInteger ((daeConfig)self, (str)propertyPath[, (int)defaultValue]) → int
    GetString ((daeConfig)self, (str)propertyPath[, (str)defaultValue]) → str
    Reload ((daeConfig)self) → None
    SetBoolean ((daeConfig)self, (str)propertyPath, (bool)value) → None
    SetFloat ((daeConfig)self, (str)propertyPath, (float)value) → None
    SetInteger ((daeConfig)self, (str)propertyPath, (int)value) → None
    SetString ((daeConfig)self, (str)propertyPath, (str)value) → None
    has_key ((daeConfig)self, (object)propertyPath) → object
```

1.3.6 Auxiliary functions

daeGetConfig
daeVersion
daeVersionMajor
daeVersionMinor
daeVersionBuild

`pyCore.daeGetConfig()` → object

`pyCore.daeVersion([(bool)includeBuild=False])` → str

`pyCore.daeVersionMajor()` → int

`pyCore.daeVersionMinor()` → int

`pyCore.daeVersionBuild()` → int

1.3.7 Global constants

<code>cnAlgebraic</code>	<code>int(x[, base]) -> integer</code>
<code>cnDifferential</code>	<code>int(x[, base]) -> integer</code>
<code>cnAssigned</code>	<code>int(x[, base]) -> integer</code>

`pyCore.cnAlgebraic = 0`
`int(x[, base]) -> integer`

Convert a string or number to an integer, if possible. A floating point argument will be truncated towards zero (this does not include a string representation of a floating point number!) When converting a string, use the optional base. It is an error to supply a base when converting a non-string. If base is zero, the proper base is guessed based on the string content. If the argument is outside the integer range a long object will be returned instead.

`pyCore.cnDifferential = 1`
`int(x[, base]) -> integer`

Convert a string or number to an integer, if possible. A floating point argument will be truncated towards zero (this does not include a string representation of a floating point number!) When converting a string, use the optional base. It is an error to supply a base when converting a non-string. If base is zero, the proper base is guessed based on the string content. If the argument is outside the integer range a long object will be returned instead.

`pyCore.cnAssigned = 2`
`int(x[, base]) -> integer`

Convert a string or number to an integer, if possible. A floating point argument will be truncated towards zero (this does not include a string representation of a floating point number!) When converting a string, use the optional base. It is an error to supply a base when converting a non-string. If base is zero, the proper base is guessed based on the string content. If the argument is outside the integer range a long object will be returned instead.

MODULE PYACTIVITY

2.1 Overview

Trt mrt.

daeSimulation
daeOptimization

2.1.1 daeSimulation

```

class pyActivity.daeSimulation
    Bases: pyActivity.daeSimulation_t
    AbsoluteTolerances
    ActivityAction
    CleanupSetupData ((daeSimulation)arg1) → None
        CleanupSetupData( (daeSimulation)arg1) -> None
    Constraints
    CreateEqualityConstraint ((daeSimulation)arg1, (str)arg2) → object
    CreateInequalityConstraint ((daeSimulation)arg1, (str)arg2) → object
    CurrentTime
    DAEsolver
    DataReporter
    Finalize ((daeSimulation)arg1) → None
    IndexMappings
    InitialConditionMode
    InitialDerivatives
    InitialValues
    Initialize ((daeSimulation)arg1, (object)arg2, (object)arg3, (object)arg4[,
        (bool)CalculateSensitivities=False]) → None
    InputVariables
  
```

Integrate ((*daeSimulation*)arg1, (*daeStopCriterion*)arg2[, (*bool*)ReportDataAroundDiscontinuities=True
]) → float

IntegrateForTimeInterval ((*daeSimulation*)arg1, (float)arg2[, (*bool*)ReportDataAroundDiscontinuities=True]) → float

IntegrateUntilTime ((*daeSimulation*)arg1, (float)arg2, (*daeStopCriterion*)arg3[, (*bool*)ReportDataAroundDiscontinuities=True]) → float

LoadInitializationValues ((*daeSimulation*)arg1, (*str*)arg2) → None

Log

MeasuredVariables

Model

ModelParameters

NextReportingTime

NumberOfEquations

NumberOfObjectiveFunctions

ObjectiveFunction

ObjectiveFunctions

OptimizationVariables

Pause ((*daeSimulation*)arg1) → None

ReRun ((*daeSimulation*)arg1) → None

RegisterData ((*daeSimulation*)arg1, (*str*)arg2) → None

Reinitialize ((*daeSimulation*)arg1) → None

RelativeTolerance

ReportData ((*daeSimulation*)arg1, (float)arg2) → None

ReportingInterval

ReportingTimes

Reset ((*daeSimulation*)arg1) → None

Resume ((*daeSimulation*)arg1) → None

Run ((*daeSimulation*)arg1) → None
Run((*daeSimulation*)arg1) -> None

SetBinaryOptimizationVariable ((*daeSimulation*)arg1, (*object*)arg2, (*bool*)arg3) → object
SetBinaryOptimizationVariable((*daeSimulation*)arg1, (*object*)arg2, (*bool*)arg3) -> object

SetContinuousOptimizationVariable ((*daeSimulation*)arg1, (*object*)arg2, (float)arg3,
(float)arg4, (float)arg5) → object
SetContinuousOptimizationVariable((*daeSimulation*)arg1, (*object*)arg2, (float)arg3, (float)arg4,
(float)arg5) -> object

SetInputVariable ((*daeSimulation*)arg1, (*object*)arg2) → object
SetInputVariable((*daeSimulation*)arg1, (*object*)arg2) -> object

```

SetIntegerOptimizationVariable ((daeSimulation)arg1, (object)arg2, (int)arg3, (int)arg4,
                                   (int)arg5) → object
    SetIntegerOptimizationVariable( (daeSimulation)arg1, (object)arg2, (int)arg3, (int)arg4, (int)arg5) -> ob-
    ject

SetMeasuredVariable ((daeSimulation)arg1, (object)arg2) → object
    SetMeasuredVariable( (daeSimulation)arg1, (object)arg2) -> object

SetModelParameter ((daeSimulation)arg1, (object)arg2, (float)arg3, (float)arg4, (float)arg5) → ob-
    ject
    SetModelParameter( (daeSimulation)arg1, (object)arg2, (float)arg3, (float)arg4, (float)arg5) -> object

SetUpOptimization ((daeSimulation)arg1) → None
    SetUpOptimization( (daeSimulation)arg1) -> None

SetUpParameterEstimation ((daeSimulation)arg1) → None
    SetUpParameterEstimation( (daeSimulation)arg1) -> None

SetUpParametersAndDomains ((daeSimulation)arg1) → None
    SetUpParametersAndDomains( (daeSimulation)arg1) -> None

SetUpSensitivityAnalysis ((daeSimulation)arg1) → None
    SetUpSensitivityAnalysis( (daeSimulation)arg1) -> None

SetUpVariables ((daeSimulation)arg1) → None
    SetUpVariables( (daeSimulation)arg1) -> None

SimulationMode

SolveInitial ((daeSimulation)arg1) → None

StoreInitializationValues ((daeSimulation)arg1, (str)arg2) → None

TimeHorizon

TotalNumberOfVariables

VariableTypes

__init__ ((object)arg1) → None
__instance_size__ = 440
__module__ = 'pyActivity'
__reduce__ ()
m
model

```

2.1.2 daeOptimization

```

class pyActivity.daeOptimization
    Bases: pyActivity.daeOptimization_t

    Finalize ((daeOptimization)arg1) → None

    Initialize ((daeOptimization)arg1, (daeSimulation_t)arg2, (object)arg3, (object)arg4, (object)arg5,
                (object)arg6) → None

    Run ((daeOptimization)arg1) → None

    __init__ ((object)arg1) → None
    __instance_size__ = 88

```

```
__module__ = 'pyActivity'  
__reduce__()
```

MODULE PYDATAREPORTING

3.1 Overview

Trt mrt.

`pyDataReporting`

3.1.1 `daeDataReporter_t`

```
class pyDataReporting.daeDataReporter_t
    Bases: Boost.Python.instance

    Connect ((daeDataReporter_t)arg1, (str)arg2, (str)arg3) → bool
        Connect( (daeDataReporter_t)arg1, (str)arg2, (str)arg3) -> None

    Disconnect ((daeDataReporter_t)arg1) → bool
        Disconnect( (daeDataReporter_t)arg1) -> None

    EndOfData ((daeDataReporter_t)arg1) → bool
        EndOfData( (daeDataReporter_t)arg1) -> None

    EndRegistration ((daeDataReporter_t)arg1) → bool
        EndRegistration( (daeDataReporter_t)arg1) -> None

    IsConnected ((daeDataReporter_t)arg1) → bool
        IsConnected( (daeDataReporter_t)arg1) -> None

    RegisterDomain ((daeDataReporter_t)arg1, (daeDataReporterDomain)arg2) → bool
        RegisterDomain( (daeDataReporter_t)arg1, (daeDataReporterDomain)arg2) -> None

    RegisterVariable ((daeDataReporter_t)arg1, (daeDataReporterVariable)arg2) → bool
        RegisterVariable( (daeDataReporter_t)arg1, (daeDataReporterVariable)arg2) -> None

    SendVariable ((daeDataReporter_t)arg1, (daeDataReporterVariableValue)arg2) → bool
        SendVariable( (daeDataReporter_t)arg1, (daeDataReporterVariableValue)arg2) -> None

    StartNewResultSet ((daeDataReporter_t)arg1, (float)arg2) → bool
        StartNewResultSet( (daeDataReporter_t)arg1, (float)arg2) -> None

    StartRegistration ((daeDataReporter_t)arg1) → bool
        StartRegistration( (daeDataReporter_t)arg1) -> None
```


MODULE PYIDAS

4.1 Overview

Trt mrt.

daeIDAS

4.1.1 daeIDAS

```
class pyIDAS.daeIDAS
    Bases: pyIDAS.daeDAESolver_t
    SaveMatrixAsXPM((daeIDAS)arg1, (str)arg2) → None
    SetLASolver((daeIDAS)arg1, (daeIDASLASolverType)arg2) → None
    SetLASolver( (daeIDAS)arg1, (object)arg2) -> None
```


MODULE PYUNITS

5.1 Overview

Trt mrt.

5.2 Classes

<code>unit</code>
<code>quantity</code>

5.2.1 unit

```
class pyUnits.unit
    Bases: Boost.Python.instance
    baseUnit
    unitDictionary
```

5.2.2 quantity

```
class pyUnits.quantity
    Bases: Boost.Python.instance
    scaleTo ((quantity)arg1, (object)arg2) → quantity
    units
    value
    valueInSIUnits
```


INDICES AND TABLES

- *genindex*
- *modindex*
- *search*

PYTHON MODULE INDEX

p

- `pyActivity`, [23](#)
- `pyCore`, [1](#)
- `pyDataReporting`, [27](#)
- `pyIDAS`, [29](#)
- `pyUnits`, [31](#)

INDEX

Symbols

`__and__()` (pyCore.daeCondition method), 2
`__call__()` (pyCore.daeDEDI method), 17
`__call__()` (pyCore.daeDomain method), 9
`__call__()` (pyCore.daeParameter method), 10
`__call__()` (pyCore.daeScalarExternalFunction method), 15
`__call__()` (pyCore.daeVariable method), 12
`__call__()` (pyCore.daeVectorExternalFunction method), 16
`__contains__()` (pyCore.daeConfig method), 20
`__getitem__()` (pyCore.adouble_array method), 1
`__getitem__()` (pyCore.daeConfig method), 20
`__getitem__()` (pyCore.daeDomain method), 9
`__init__()` (pyActivity.daeOptimization method), 25
`__init__()` (pyActivity.daeSimulation method), 25
`__init__()` (pyCore.daeArrayRange method), 16
`__init__()` (pyCore.daeDomain method), 9
`__init__()` (pyCore.daeDomainIndex method), 16
`__init__()` (pyCore.daeIndexRange method), 16
`__init__()` (pyCore.daeModel method), 12
`__init__()` (pyCore.daeVariableType method), 9
`__init__()` (pyCore.daeVariableWrapper method), 20
`__instance_size__` (pyActivity.daeOptimization attribute), 25
`__instance_size__` (pyActivity.daeSimulation attribute), 25
`__len__()` (pyCore.adouble_array method), 1
`__module__` (pyActivity.daeOptimization attribute), 26
`__module__` (pyActivity.daeSimulation attribute), 25
`__or__()` (pyCore.daeCondition method), 2
`__reduce__()` (pyActivity.daeOptimization method), 26
`__reduce__()` (pyActivity.daeSimulation method), 25
`__setitem__()` (pyCore.adouble_array method), 2
`__setitem__()` (pyCore.daeConfig method), 20

A

`Abs()` (in module pyCore), 3
`AbsoluteTolerance` (pyCore.daeVariableType attribute), 9
`AbsoluteTolerances` (pyActivity.daeSimulation attribute), 23

`ACos()` (in module pyCore), 3
`ACosh()` (in module pyCore), 3
`Actions` (pyCore.daeStateTransition attribute), 15
`ActiveState` (pyCore.daeSTN attribute), 14
`ActivityAction` (pyActivity.daeSimulation attribute), 23
`adouble` (class in pyCore), 1
`adouble_array` (class in pyCore), 1
`Array()` (in module pyCore), 19
`array()` (pyCore.daeParameter method), 10
`array()` (pyCore.daeVariable method), 11
`ASin()` (in module pyCore), 3
`ASinh()` (in module pyCore), 3
`AssignValue()` (pyCore.daeVariable method), 11
`AssignValues()` (pyCore.daeVariable method), 11
`ATan()` (in module pyCore), 3
`ATan2()` (in module pyCore), 3
`ATanh()` (in module pyCore), 3
`Average()` (in module pyCore), 20

B

`baseUnit` (pyUnits.unit attribute), 31

C

`Calculate()` (pyCore.daeScalarExternalFunction method), 15
`Calculate()` (pyCore.daeVectorExternalFunction method), 16
`CanonicalName` (pyCore.daeObject attribute), 9
`Ceil()` (in module pyCore), 3
`CleanUpSetupData()` (pyActivity.daeSimulation method), 23
`cnAlgebraic` (in module pyCore), 21
`cnAssigned` (in module pyCore), 21
`cnDifferential` (in module pyCore), 21
`ComponentArrays` (pyCore.daeModel attribute), 13
`Components` (pyCore.daeModel attribute), 13
`Condition` (pyCore.daeStateTransition attribute), 15
`Connect()` (pyDataReporting.daeDataReporter_t method), 27
`ConnectEventPorts()` (pyCore.daeModel method), 13
`ConnectPorts()` (pyCore.daeModel method), 13

Constant() (in module pyCore), 19
Constraints (pyActivity.daeSimulation attribute), 23
Cos() (in module pyCore), 3
Cosh() (in module pyCore), 3
CreateArray() (pyCore.daeDomain method), 9
CreateDistributed() (pyCore.daeDomain method), 9
CreateEqualityConstraint() (pyActivity.daeSimulation method), 23
CreateEquation() (pyCore.daeModel method), 13
CreateInequalityConstraint() (pyActivity.daeSimulation method), 23
CurrentTime (pyActivity.daeSimulation attribute), 23

D

d() (in module pyCore), 19
d() (pyCore.daeVariable method), 12
d2() (pyCore.daeVariable method), 12
d2_array() (pyCore.daeVariable method), 12
d_array() (pyCore.daeVariable method), 12
daeAction (class in pyCore), 17
daeArrayRange (class in pyCore), 16
daeBaseLog (class in pyCore), 18
daeCondition (class in pyCore), 2
daeConfig (class in pyCore), 20
daeDataReporter_t (class in pyDataReporting), 27
daeDEDI (class in pyCore), 17
daeDomain (class in pyCore), 9
daeDomainIndex (class in pyCore), 16
daeActionType (class in pyCore), 8
daeBinaryFunctions (class in pyCore), 7
daeConditionType (class in pyCore), 7
daeConstraintType (class in pyCore), 6
daeDiscretizationMethod (class in pyCore), 5
daeDomainBounds (class in pyCore), 5
daeDomainIndexType (class in pyCore), 5
daeDomainType (class in pyCore), 4
daeEquationType (class in pyCore), 8
daeInitialConditionMode (class in pyCore), 5
daeLogicalBinaryOperator (class in pyCore), 7
daeLogicalUnaryOperator (class in pyCore), 7
daeModelLanguage (class in pyCore), 6
daeModelType (class in pyCore), 8
daeOptimizationVariableType (class in pyCore), 6
daeParameterType (class in pyCore), 4
daePortType (class in pyCore), 5
daeEquation (class in pyCore), 14
daeEquationExecutionInfo (class in pyCore), 18
daeRangeType (class in pyCore), 5
daeSpecialUnaryFunctions (class in pyCore), 7
daeUnaryFunctions (class in pyCore), 6
daeEventPort (class in pyCore), 15
daeFileLog (class in pyCore), 19
daeGetConfig() (in module pyCore), 21
daeIDAS (class in pyIDAS), 29

daeIF (class in pyCore), 14
daeIndexRange (class in pyCore), 16
daeIndexRangeType (class in pyCore), 6
daeLog_t (class in pyCore), 18
daeMeasuredVariable (class in pyCore), 18
daeModel (class in pyCore), 12
daeObject (class in pyCore), 9
daeObjectiveFunction (class in pyCore), 17
daeOptimization (class in pyActivity), 25
daeOptimizationConstraint (class in pyCore), 17
daeOptimizationVariable (class in pyCore), 17
daeParameter (class in pyCore), 10
daePort (class in pyCore), 15
daePortConnection (class in pyCore), 15
daeScalarExternalFunction (class in pyCore), 15
daeSimulation (class in pyActivity), 23
DAESolver (pyActivity.daeSimulation attribute), 23
daeState (class in pyCore), 15
daeStateTransition (class in pyCore), 15
daeStdOutLog (class in pyCore), 19
daeSTN (class in pyCore), 14
daeTCPIPLog (class in pyCore), 19
daeTCPIPLogServer (class in pyCore), 19
daeVariable (class in pyCore), 11
daeVariableType (class in pyCore), 9
daeVariableWrapper (class in pyCore), 20
daeVectorExternalFunction (class in pyCore), 16
daeVersion() (in module pyCore), 21
daeVersionBuild() (in module pyCore), 21
daeVersionMajor() (in module pyCore), 21
daeVersionMinor() (in module pyCore), 21
DataReporter (pyActivity.daeSimulation attribute), 23
DeclareEquations() (pyCore.daeModel method), 13
DecreaseIndent() (pyCore.daeBaseLog method), 19
DecreaseIndent() (pyCore.daeLog_t method), 18
DEDI (pyCore.daeDomainIndex attribute), 16
Derivative (pyCore.adouble attribute), 1
Description (pyCore.daeObject attribute), 9
Disconnect() (pyDataReporting.daeDataReporter_t method), 27
DiscretizationMethod (pyCore.daeDomain attribute), 10
DiscretizationOrder (pyCore.daeDomain attribute), 10
DistributedEquationDomainInfos (pyCore.daeEquation attribute), 14
DistributeOnDomain() (pyCore.daeEquation method), 14
DistributeOnDomain() (pyCore.daeParameter method), 11
DistributeOnDomain() (pyCore.daeVariable method), 12
Domain (pyCore.daeDEDI attribute), 17
Domain (pyCore.daeIndexRange attribute), 16
DomainBounds (pyCore.daeDEDI attribute), 17
DomainIndex (pyCore.daeArrayRange attribute), 16
DomainIndexes (pyCore.daeVariableWrapper attribute), 20

DomainPoints (pyCore.daeDEDI attribute), 17
 Domains (pyCore.daeModel attribute), 13
 Domains (pyCore.daeParameter attribute), 11
 Domains (pyCore.daePort attribute), 15
 Domains (pyCore.daeVariable attribute), 12
 dt() (in module pyCore), 19
 dt() (pyCore.daeVariable method), 12
 dt_array() (pyCore.daeVariable method), 12

E

eAbs (pyCore.daeUnaryFunctions attribute), 6
 eAlgebraic (pyCore.daeEquationType attribute), 8
 eAlgebraicValuesProvided (pyCore.daeInitialConditionMode attribute), 5
 eAllPointsInDomain (pyCore.daeIndexRangeType attribute), 6
 eAnd (pyCore.daeLogicalBinaryOperator attribute), 7
 eArcCos (pyCore.daeUnaryFunctions attribute), 6
 eArcSin (pyCore.daeUnaryFunctions attribute), 6
 eArcTan (pyCore.daeUnaryFunctions attribute), 6
 eArray (pyCore.daeDomainType attribute), 4
 eAverage (pyCore.daeSpecialUnaryFunctions attribute), 7
 eBFDM (pyCore.daeDiscretizationMethod attribute), 5
 eBFUnknown (pyCore.daeBinaryFunctions attribute), 7
 eBinaryVariable (pyCore.daeOptimizationVariableType attribute), 6
 eBool (pyCore.daeParameterType attribute), 4
 eBOUnknown (pyCore.daeLogicalBinaryOperator attribute), 7
 eCDAE (pyCore.daeModelLanguage attribute), 6
 eCeil (pyCore.daeUnaryFunctions attribute), 6
 eCFDM (pyCore.daeDiscretizationMethod attribute), 5
 eChangeState (pyCore.daeActionType attribute), 8
 eClosedClosed (pyCore.daeDomainBounds attribute), 5
 eClosedOpen (pyCore.daeDomainBounds attribute), 5
 eConstantIndex (pyCore.daeDomainIndexType attribute), 5
 eContinuousVariable (pyCore.daeOptimizationVariableType attribute), 6
 eCos (pyCore.daeUnaryFunctions attribute), 6
 eCTUnknown (pyCore.daeConditionType attribute), 7
 eCustomDM (pyCore.daeDiscretizationMethod attribute), 5
 eCustomRange (pyCore.daeIndexRangeType attribute), 6
 eDAE (pyCore.daeModelType attribute), 8
 eDBUnknown (pyCore.daeDomainBounds attribute), 5
 eDifferentialValuesProvided (pyCore.daeInitialConditionMode attribute), 5
 eDistributed (pyCore.daeDomainType attribute), 4
 eDITUnknown (pyCore.daeDomainIndexType attribute), 5
 eDivide (pyCore.daeBinaryFunctions attribute), 7
 eDMUnknown (pyCore.daeDiscretizationMethod attribute), 5
 eDomainIterator (pyCore.daeDomainIndexType attribute), 5
 eDTUnknown (pyCore.daeDomainType attribute), 4
 eEQ (pyCore.daeConditionType attribute), 7
 eEqualityConstraint (pyCore.daeConstraintType attribute), 6
 eETUnknown (pyCore.daeEquationType attribute), 8
 eExp (pyCore.daeUnaryFunctions attribute), 6
 eExplicitODE (pyCore.daeEquationType attribute), 8
 eFFDM (pyCore.daeDiscretizationMethod attribute), 5
 eFloor (pyCore.daeUnaryFunctions attribute), 6
 eGT (pyCore.daeConditionType attribute), 7
 eGTEQ (pyCore.daeConditionType attribute), 7
 eICTUnknown (pyCore.daeInitialConditionMode attribute), 5
 eImplicitODE (pyCore.daeEquationType attribute), 8
 eIncrementedDomainIterator (pyCore.daeDomainIndexType attribute), 5
 eInequalityConstraint (pyCore.daeConstraintType attribute), 6
 eInletPort (pyCore.daePortType attribute), 5
 eInteger (pyCore.daeParameterType attribute), 5
 eIntegerVariable (pyCore.daeOptimizationVariableType attribute), 6
 eIRTUnknown (pyCore.daeIndexRangeType attribute), 6
 eLn (pyCore.daeUnaryFunctions attribute), 6
 eLog (pyCore.daeUnaryFunctions attribute), 6
 eLowerBound (pyCore.daeDomainBounds attribute), 5
 ELSE() (pyCore.daeModel method), 13
 ELSE_IF() (pyCore.daeModel method), 13
 eLT (pyCore.daeConditionType attribute), 7
 eLTEQ (pyCore.daeConditionType attribute), 7
 eMax (pyCore.daeBinaryFunctions attribute), 7
 eMaxInArray (pyCore.daeSpecialUnaryFunctions attribute), 7
 eMin (pyCore.daeBinaryFunctions attribute), 7
 eMinInArray (pyCore.daeSpecialUnaryFunctions attribute), 7
 eMinus (pyCore.daeBinaryFunctions attribute), 7
 eMLNone (pyCore.daeModelLanguage attribute), 6
 eMTUnknown (pyCore.daeModelType attribute), 8
 eMulti (pyCore.daeBinaryFunctions attribute), 7
 Enabled (pyCore.daeLog_t attribute), 18
 END_IF() (pyCore.daeModel method), 13
 END_STN() (pyCore.daeModel method), 13
 EndIndex (pyCore.daeIndexRange attribute), 16
 EndOfData() (pyDataReporting.daeDataReporter_t method), 27

EndRegistration() (pyDataReporting.daeDataReporter_t method), 27

eNot (pyCore.daeLogicalUnaryOperator attribute), 7

eNotEQ (pyCore.daeConditionType attribute), 8

eODE (pyCore.daeModelType attribute), 8

eOpenClosed (pyCore.daeDomainBounds attribute), 5

eOpenOpen (pyCore.daeDomainBounds attribute), 5

eOr (pyCore.daeLogicalBinaryOperator attribute), 7

eOutletPort (pyCore.daePortType attribute), 5

ePlus (pyCore.daeBinaryFunctions attribute), 7

ePower (pyCore.daeBinaryFunctions attribute), 7

eProduct (pyCore.daeSpecialUnaryFunctions attribute), 7

ePTUnknown (pyCore.daeParameterType attribute), 5

ePYDAE (pyCore.daeModelLanguage attribute), 6

eQuasySteadyState (pyCore.daeInitialConditionMode attribute), 5

EquationExecutionInfos (pyCore.daeEquation attribute), 15

Equations (pyCore.daeModel attribute), 13

Equations (pyCore.daePortConnection attribute), 15

Equations (pyCore.daeState attribute), 15

EquationType (pyCore.daeEquation attribute), 15

EquationType (pyCore.daeEquationExecutionInfo attribute), 18

eRange (pyCore.daeRangeType attribute), 6

eRangeDomainIndex (pyCore.daeRangeType attribute), 6

eRangeOfIndexes (pyCore.daeIndexRangeType attribute), 6

eRaTUnknown (pyCore.daeRangeType attribute), 6

eReal (pyCore.daeParameterType attribute), 5

eReAssignOrReInitializeVariable (pyCore.daeActionType attribute), 8

eSendEvent (pyCore.daeActionType attribute), 8

eSign (pyCore.daeUnaryFunctions attribute), 6

eSin (pyCore.daeUnaryFunctions attribute), 6

eSqrt (pyCore.daeUnaryFunctions attribute), 7

eSteadyState (pyCore.daeModelType attribute), 8

eSUUnknown (pyCore.daeSpecialUnaryFunctions attribute), 7

eSum (pyCore.daeSpecialUnaryFunctions attribute), 7

ETA (pyCore.daeLog_t attribute), 18

eTan (pyCore.daeUnaryFunctions attribute), 7

eUUnknown (pyCore.daeUnaryFunctions attribute), 7

eUnknownAction (pyCore.daeActionType attribute), 8

eUnknownPort (pyCore.daePortType attribute), 5

eUOUnknown (pyCore.daeLogicalUnaryOperator attribute), 7

eUpperBound (pyCore.daeDomainBounds attribute), 5

eUserDefinedAction (pyCore.daeActionType attribute), 8

EventData (pyCore.daeEventPort attribute), 15

EventPorts (pyCore.daeModel attribute), 13

Events (pyCore.daeEventPort attribute), 15

EventTolerance (pyCore.daeCondition attribute), 2

Execute() (pyCore.daeAction method), 17

Exp() (in module pyCore), 3

Export() (pyCore.daeModel method), 13

Export() (pyCore.daePort method), 15

ExportObjects() (pyCore.daeModel method), 13

Expressions (pyCore.daeCondition attribute), 2

F

Finalize() (pyActivity.daeOptimization method), 25

Finalize() (pyActivity.daeSimulation method), 23

Floor() (in module pyCore), 3

G

GatherInfo (pyCore.adouble attribute), 1

GatherInfo (pyCore.adouble_array attribute), 2

GetBoolean() (pyCore.daeConfig method), 20

GetDomainsIndexesMap() (pyCore.daeParameter method), 11

GetDomainsIndexesMap() (pyCore.daeVariable method), 12

GetFloat() (pyCore.daeConfig method), 20

GetInteger() (pyCore.daeConfig method), 20

GetNameRelativeToParentModel() (pyCore.daeObject method), 9

GetQuantity() (pyCore.daeParameter method), 10

GetQuantity() (pyCore.daeVariable method), 11

GetString() (pyCore.daeConfig method), 20

GetStrippedName() (pyCore.daeObject method), 9

GetStrippedNameRelativeToParentModel() (pyCore.daeObject method), 9

GetValue() (pyCore.daeParameter method), 10

GetValue() (pyCore.daeVariable method), 11

Gradients (pyCore.daeMeasuredVariable attribute), 18

Gradients (pyCore.daeObjectiveFunction attribute), 17

Gradients (pyCore.daeOptimizationConstraint attribute), 17

H

has_key() (pyCore.daeConfig method), 20

I

ID (pyCore.daeObject attribute), 9

IF() (pyCore.daeModel method), 13

IncreaseIndent() (pyCore.daeBaseLog method), 19

IncreaseIndent() (pyCore.daeLog_t method), 18

Increment (pyCore.daeDomainIndex attribute), 16

Indent (pyCore.daeLog_t attribute), 18

IndentString (pyCore.daeLog_t attribute), 18

Index (pyCore.daeDomainIndex attribute), 16

IndexMappings (pyActivity.daeSimulation attribute), 23

InitialConditionMode (pyActivity.daeSimulation attribute), 23

[InitialConditionMode](#) (pyCore.daeModel attribute), 13
[InitialDerivatives](#) (pyActivity.daeSimulation attribute), 23
[InitialGuess](#) (pyCore.daeVariableType attribute), 9
[Initialize\(\)](#) (pyActivity.daeOptimization method), 25
[Initialize\(\)](#) (pyActivity.daeSimulation method), 23
[InitialValues](#) (pyActivity.daeSimulation attribute), 23
[InputVariables](#) (pyActivity.daeSimulation attribute), 23
[Integral\(\)](#) (in module pyCore), 20
[Integrate\(\)](#) (pyActivity.daeSimulation method), 23
[IntegrateForTimeInterval\(\)](#) (pyActivity.daeSimulation method), 24
[IntegrateUntilTime\(\)](#) (pyActivity.daeSimulation method), 24
[IsConnected\(\)](#) (pyDataReporting.daeDataReporter_t method), 27
[IsModelDynamic](#) (pyCore.daeModel attribute), 13
[items\(\)](#) (pyCore.adouble_array method), 2

J

[JoinMessages\(\)](#) (pyCore.daeLog_t method), 18

L

[Library](#) (pyCore.daeObject attribute), 9
[LoadInitializationValues\(\)](#) (pyActivity.daeSimulation method), 24
[Log](#) (pyActivity.daeSimulation attribute), 24
[Log\(\)](#) (in module pyCore), 3
[Log10\(\)](#) (in module pyCore), 3
[LowerBound](#) (pyCore.daeDomain attribute), 10
[LowerBound](#) (pyCore.daeOptimizationVariable attribute), 17
[LowerBound](#) (pyCore.daeVariableType attribute), 9

M

[m](#) (pyActivity.daeSimulation attribute), 25
[Max\(\)](#) (in module pyCore), 4
[MeasuredVariables](#) (pyActivity.daeSimulation attribute), 24
[Message\(\)](#) (pyCore.daeBaseLog method), 19
[Message\(\)](#) (pyCore.daeFileLog method), 19
[Message\(\)](#) (pyCore.daeLog_t method), 18
[Message\(\)](#) (pyCore.daeStdOutLog method), 19
[Message\(\)](#) (pyCore.daeTCPIPLog method), 19
[MessageReceived\(\)](#) (pyCore.daeTCPIPLogServer method), 19
[Min\(\)](#) (in module pyCore), 3
[Model](#) (pyActivity.daeSimulation attribute), 24
[model](#) (pyActivity.daeSimulation attribute), 25
[Model](#) (pyCore.daeObject attribute), 9
[ModelParameters](#) (pyActivity.daeSimulation attribute), 24
[ModelType](#) (pyCore.daeModel attribute), 13

N

[Name](#) (pyCore.daeMeasuredVariable attribute), 18
[Name](#) (pyCore.daeObject attribute), 9
[Name](#) (pyCore.daeObjectiveFunction attribute), 17
[Name](#) (pyCore.daeOptimizationConstraint attribute), 17
[Name](#) (pyCore.daeOptimizationVariable attribute), 17
[Name](#) (pyCore.daeScalarExternalFunction attribute), 16
[Name](#) (pyCore.daeVariableType attribute), 9
[Name](#) (pyCore.daeVariableWrapper attribute), 20
[Name](#) (pyCore.daeVectorExternalFunction attribute), 16
[NestedSTNs](#) (pyCore.daeState attribute), 15
[NextReportingTime](#) (pyActivity.daeSimulation attribute), 24
[Node](#) (pyCore.adouble attribute), 1
[Node](#) (pyCore.adouble_array attribute), 2
[Node](#) (pyCore.daeEquationExecutionInfo attribute), 18
[NoPoints](#) (pyCore.daeArrayRange attribute), 16
[NoPoints](#) (pyCore.daeIndexRange attribute), 16
[npIDs](#) (pyCore.daeVariable attribute), 12
[npPoints](#) (pyCore.daeDomain attribute), 10
[npValues](#) (pyCore.daeParameter attribute), 11
[npValues](#) (pyCore.daeVariable attribute), 12
[NumberOfEquations](#) (pyActivity.daeSimulation attribute), 24
[NumberOfIntervals](#) (pyCore.daeDomain attribute), 10
[NumberOfObjectiveFunctions](#) (pyActivity.daeSimulation attribute), 24
[NumberOfPoints](#) (pyCore.daeDomain attribute), 10
[NumberOfPoints](#) (pyCore.daeParameter attribute), 11
[NumberOfPoints](#) (pyCore.daeVariable attribute), 12

O

[ObjectiveFunction](#) (pyActivity.daeSimulation attribute), 24
[ObjectiveFunctions](#) (pyActivity.daeSimulation attribute), 24
[ON_CONDITION\(\)](#) (pyCore.daeModel method), 13
[ON_EVENT\(\)](#) (pyCore.daeModel method), 14
[OnEventActions](#) (pyCore.daeModel attribute), 14
[OptimizationVariables](#) (pyActivity.daeSimulation attribute), 24
[OverallIndex](#) (pyCore.daeVariable attribute), 12
[OverallIndex](#) (pyCore.daeVariableWrapper attribute), 20

P

[Parameters](#) (pyCore.daeModel attribute), 14
[Parameters](#) (pyCore.daePort attribute), 15
[Pause\(\)](#) (pyActivity.daeSimulation method), 24
[PercentageDone](#) (pyCore.daeLog_t attribute), 18
[Points](#) (pyCore.daeDomain attribute), 10
[PortArrays](#) (pyCore.daeModel attribute), 14
[PortConnections](#) (pyCore.daeModel attribute), 14
[PortFrom](#) (pyCore.daePortConnection attribute), 15

Ports (pyCore.daeModel attribute), 14
PortTo (pyCore.daePortConnection attribute), 15
Pow() (in module pyCore), 3
PrintProgress (pyCore.daeLog_t attribute), 18
Product() (in module pyCore), 20
Progress (pyCore.daeLog_t attribute), 18
pyActivity (module), 23
pyCore (module), 1
pyDataReporting (module), 27
pyIDAS (module), 29
pyUnits (module), 31

Q

quantity (class in pyUnits), 31

R

Range (pyCore.daeArrayRange attribute), 16
ReAssignValue() (pyCore.daeVariable method), 11
ReAssignValues() (pyCore.daeVariable method), 11
ReceiveEvent() (pyCore.daeEventPort method), 15
RecordEvents (pyCore.daeEventPort attribute), 15
RegisterData() (pyActivity.daeSimulation method), 24
RegisterDomain() (pyDataReporting.daeDataReporter_t method), 27
RegisterVariable() (pyDataReporting.daeDataReporter_t method), 27
Reinitialize() (pyActivity.daeSimulation method), 24
RelativeTolerance (pyActivity.daeSimulation attribute), 24
Reload() (pyCore.daeConfig method), 20
ReportData() (pyActivity.daeSimulation method), 24
ReportingInterval (pyActivity.daeSimulation attribute), 24
ReportingOn (pyCore.daeParameter attribute), 11
ReportingOn (pyCore.daeVariable attribute), 12
ReportingTimes (pyActivity.daeSimulation attribute), 24
ReRun() (pyActivity.daeSimulation method), 24
Reset() (pyActivity.daeSimulation method), 24
ReSetInitialCondition() (pyCore.daeVariable method), 11
ReSetInitialConditions() (pyCore.daeVariable method), 11
Residual (pyCore.daeEquation attribute), 15
Residual (pyCore.daeMeasuredVariable attribute), 18
Residual (pyCore.daeObjectiveFunction attribute), 17
Residual (pyCore.daeOptimizationConstraint attribute), 17
Resize() (pyCore.adouble_array method), 2
Resume() (pyActivity.daeSimulation method), 24
Run() (pyActivity.daeOptimization method), 25
Run() (pyActivity.daeSimulation method), 24
RuntimeNode (pyCore.daeAction attribute), 17
RuntimeNode (pyCore.daeCondition attribute), 2

S

SaveMatrixAsXPM() (pyIDAS.daeIDAS method), 29
SaveModelReport() (pyCore.daeModel method), 14
SaveRuntimeModelReport() (pyCore.daeModel method), 14
scaleTo() (pyUnits.quantity method), 31
Scaling (pyCore.daeEquation attribute), 15
SendEvent() (pyCore.daeEventPort method), 15
SendEventPort (pyCore.daeAction attribute), 17
SendVariable() (pyDataReporting.daeDataReporter_t method), 27
SetAbsoluteTolerances() (pyCore.daeVariable method), 11
SetBinaryOptimizationVariable() (pyActivity.daeSimulation method), 24
SetBoolean() (pyCore.daeConfig method), 20
SetContinuousOptimizationVariable() (pyActivity.daeSimulation method), 24
SetFloat() (pyCore.daeConfig method), 20
SetInitialCondition() (pyCore.daeVariable method), 11
SetInitialConditions() (pyCore.daeVariable method), 11
SetInitialGuess() (pyCore.daeVariable method), 11
SetInitialGuesses() (pyCore.daeVariable method), 11
SetInputVariable() (pyActivity.daeSimulation method), 24
SetInteger() (pyCore.daeConfig method), 20
SetIntegerOptimizationVariable() (pyActivity.daeSimulation method), 24
SetLASolver() (pyIDAS.daeIDAS method), 29
SetMeasuredVariable() (pyActivity.daeSimulation method), 25
SetModelParameter() (pyActivity.daeSimulation method), 25
SetProgress() (pyCore.daeBaseLog method), 19
SetReportingOn() (pyCore.daeModel method), 14
SetReportingOn() (pyCore.daePort method), 15
SetString() (pyCore.daeConfig method), 20
SetupNode (pyCore.daeAction attribute), 17
SetupNode (pyCore.daeCondition attribute), 2
SetUpOptimization() (pyActivity.daeSimulation method), 25
SetUpParameterEstimation() (pyActivity.daeSimulation method), 25
SetUpParametersAndDomains() (pyActivity.daeSimulation method), 25
SetUpSensitivityAnalysis() (pyActivity.daeSimulation method), 25
SetUpVariables() (pyActivity.daeSimulation method), 25
SetValue() (pyCore.daeParameter method), 10
SetValue() (pyCore.daeVariable method), 11
SetValues() (pyCore.daeParameter method), 10
SetValues() (pyCore.daeVariable method), 11
SimulationMode (pyActivity.daeSimulation attribute), 25
Sin() (in module pyCore), 3

[Sinh\(\)](#) (in module `pyCore`), 3
[SolveInitial\(\)](#) (`pyActivity.daeSimulation` method), 25
[Sqrt\(\)](#) (in module `pyCore`), 3
[StartIndex](#) (`pyCore.daeIndexRange` attribute), 16
[StartingPoint](#) (`pyCore.daeOptimizationVariable` attribute), 17
[StartNewResultSet\(\)](#) (`pyDataReporting.daeDataReporter_t` method), 27
[StartRegistration\(\)](#) (`pyDataReporting.daeDataReporter_t` method), 27
[STATE\(\)](#) (`pyCore.daeModel` method), 14
[States](#) (`pyCore.daeSTN` attribute), 14
[StateTo](#) (`pyCore.daeAction` attribute), 17
[StateTransitions](#) (`pyCore.daeState` attribute), 15
[Step](#) (`pyCore.daeIndexRange` attribute), 16
[STN](#) (`pyCore.daeAction` attribute), 17
[STN\(\)](#) (`pyCore.daeModel` method), 14
[STNs](#) (`pyCore.daeModel` attribute), 14
[StoreInitializationValues\(\)](#) (`pyActivity.daeSimulation` method), 25
[Sum\(\)](#) (in module `pyCore`), 19
[SWITCH_TO\(\)](#) (`pyCore.daeModel` method), 14

T

[Tan\(\)](#) (in module `pyCore`), 3
[Tanh\(\)](#) (in module `pyCore`), 3
[Time\(\)](#) (in module `pyCore`), 19
[TimeHorizon](#) (`pyActivity.daeSimulation` attribute), 25
[TotalNumberOfVariables](#) (`pyActivity.daeSimulation` attribute), 25
[Type](#) (`pyCore.daeAction` attribute), 17
[Type](#) (`pyCore.daeArrayRange` attribute), 17
[Type](#) (`pyCore.daeDomain` attribute), 10
[Type](#) (`pyCore.daeDomainIndex` attribute), 16
[Type](#) (`pyCore.daeEventPort` attribute), 15
[Type](#) (`pyCore.daeIndexRange` attribute), 16
[Type](#) (`pyCore.daeOptimizationConstraint` attribute), 17
[Type](#) (`pyCore.daeOptimizationVariable` attribute), 17
[Type](#) (`pyCore.daePort` attribute), 15

U

[unit](#) (class in `pyUnits`), 31
[unitDictionary](#) (`pyUnits.unit` attribute), 31
[Units](#) (`pyCore.daeDomain` attribute), 10
[Units](#) (`pyCore.daeParameter` attribute), 11
[Units](#) (`pyCore.daeVariableType` attribute), 9
[units](#) (`pyUnits.quantity` attribute), 31
[UpperBound](#) (`pyCore.daeDomain` attribute), 10
[UpperBound](#) (`pyCore.daeOptimizationVariable` attribute), 17
[UpperBound](#) (`pyCore.daeVariableType` attribute), 9

V

[Value](#) (`pyCore.adouble` attribute), 1

[Value](#) (`pyCore.daeMeasuredVariable` attribute), 18
[Value](#) (`pyCore.daeObjectiveFunction` attribute), 17
[Value](#) (`pyCore.daeOptimizationConstraint` attribute), 18
[Value](#) (`pyCore.daeOptimizationVariable` attribute), 17
[Value](#) (`pyCore.daeVariableWrapper` attribute), 20
[value](#) (`pyUnits.quantity` attribute), 31
[valueInSIUnits](#) (`pyUnits.quantity` attribute), 31
[Variable](#) (`pyCore.daeVariableWrapper` attribute), 20
[VariableIndexes](#) (`pyCore.daeEquationExecutionInfo` attribute), 18
[Variables](#) (`pyCore.daeModel` attribute), 14
[Variables](#) (`pyCore.daePort` attribute), 15
[VariableType](#) (`pyCore.daeVariable` attribute), 12
[VariableType](#) (`pyCore.daeVariableWrapper` attribute), 20
[VariableTypes](#) (`pyActivity.daeSimulation` attribute), 25
[VariableWrapper](#) (`pyCore.daeAction` attribute), 17
[Version](#) (`pyCore.daeObject` attribute), 9