

# DareDare

API Documentation

November 23, 2008

## Contents

<b>Contents</b>	<b>1</b>
<b>1 Module compiler.compiler</b>	<b>2</b>
1.1 Functions . . . . .	2
1.2 Variables . . . . .	2
1.3 Class Compiler . . . . .	5
1.3.1 Methods . . . . .	5
1.3.2 Class Variables . . . . .	5
<b>2 Module daredare</b>	<b>6</b>
2.1 Functions . . . . .	6
2.2 Variables . . . . .	6
<b>3 Module misc.calculus</b>	<b>10</b>
3.1 Functions . . . . .	10
3.2 Variables . . . . .	10
<b>4 Module model.model</b>	<b>14</b>
4.1 Variables . . . . .	14
4.2 Class Model . . . . .	14
4.2.1 Methods . . . . .	14
4.2.2 Class Variables . . . . .	15
<b>Index</b>	<b>16</b>

# 1 Module *compiler.compiler*

Version: 0.6.0

## 1.1 Functions

### **rand(...)**

Return an array of the given dimensions which is initialized to random numbers from a uniform distribution in the range [0,1).

`rand(d0, d1, ..., dn) -> random values`

Note: This is a convenience function. If you want an interface that takes a tuple as the first argument use `numpy.random.random_sample(shape_tuple)`.

### **randn(...)**

Returns zero-mean, unit-variance Gaussian random numbers in an array of shape (d0, d1, ..., dn).

`randn(d0, d1, ..., dn) -> random values`

Note: This is a convenience function. If you want an interface that takes a tuple as the first argument use `numpy.random.standard_normal(shape_tuple)`.

## 1.2 Variables

Name	Description
ALLOW_THREADS	<b>Value:</b> 1
BUFSIZE	<b>Value:</b> 10000
C	<b>Value:</b> <sympy.core.basic.ClassesRegistry instance at 0x96e56cc>
CLIP	<b>Value:</b> 0
Catalan	<b>Value:</b> Catalan
E	<b>Value:</b> E
ERR_CALL	<b>Value:</b> 3
ERR_DEFAULT	<b>Value:</b> 0
ERR_DEFAULT2	<b>Value:</b> 2084
ERR_IGNORE	<b>Value:</b> 0
ERR_LOG	<b>Value:</b> 5
ERR_PRINT	<b>Value:</b> 4
ERR_RAISE	<b>Value:</b> 2
ERR_WARN	<b>Value:</b> 1
EulerGamma	<b>Value:</b> EulerGamma

*continued on next page*

Name	Description
FLOATING_POINT_SUPPORT	Value: 1
FPE_DIVIDEBYZERO	Value: 1
FPE_INVALID	Value: 8
FPE_OVERFLOW	Value: 2
FPE_UNDERFLOW	Value: 4
False_	Value: False
GoldenRatio	Value: GoldenRatio
I	Value: I
Inf	Value: inf
Infinity	Value: inf
MAXDIMS	Value: 32
NAN	Value: nan
NINF	Value: -inf
NZERO	Value: -0.0
NaN	Value: nan
PINF	Value: inf
PZERO	Value: 0.0
RAISE	Value: 2
S	Value: S
SHIFT_DIVIDEBYZERO	Value: 0
SHIFT_INVALID	Value: 9
SHIFT_OVERFLOW	Value: 3
SHIFT_UNDERFLOW	Value: 6
ScalarType	Value: (<type 'int'>, <type 'float'>, <type 'complex'>, <type 'l...
True_	Value: True
UFUNC_BUFSIZE_DEFAULT	Value: 10000
UFUNC_PYVALS_NAME	Value: 'UFUNC_PYVALS'
WRAP	Value: 1
absolute	Value: <ufunc 'absolute'>
arccosh	Value: <ufunc 'arccosh'>
arcsinh	Value: <ufunc 'arcsinh'>
arctan	Value: <ufunc 'arctan'>
arctan2	Value: <ufunc 'arctan2'>
bitwise_and	Value: <ufunc 'bitwise_and'>
bitwise_not	Value: <ufunc 'invert'>
bitwise_or	Value: <ufunc 'bitwise_or'>
bitwise_xor	Value: <ufunc 'bitwise_xor'>
c_	Value: <numpy.lib.index.tricks.CClass object at 0x9aef10c>
cast	Value: {<type 'numpy.int64'>: <function <lambda> at 0x99c5374>, ...
ceil	Value: <ufunc 'ceil'>
conj	Value: <ufunc 'conjugate'>
degrees	Value: <ufunc 'degrees'>
divide	Value: <ufunc 'divide'>
e	Value: 2.71828182846
equal	Value: <ufunc 'equal'>
expm1	Value: <ufunc 'expm1'>

*continued on next page*

Name	Description
<code>fabs</code>	Value: <code>&lt;ufunc 'fabs'&gt;</code>
<code>floor_divide</code>	Value: <code>&lt;ufunc 'floor_divide'&gt;</code>
<code>fmod</code>	Value: <code>&lt;ufunc 'fmod'&gt;</code>
<code>frexp</code>	Value: <code>&lt;ufunc 'frexp'&gt;</code>
<code>greater</code>	Value: <code>&lt;ufunc 'greater'&gt;</code>
<code>greater_equal</code>	Value: <code>&lt;ufunc 'greater_equal'&gt;</code>
<code>hypot</code>	Value: <code>&lt;ufunc 'hypot'&gt;</code>
<code>index_exp</code>	Value: <code>&lt;numpy.lib.index.tricks.IndexExpression object at 0x9aef28c&gt;</code>
<code>inf</code>	Value: <code>inf</code>
<code>infty</code>	Value: <code>inf</code>
<code>invert</code>	Value: <code>&lt;ufunc 'invert'&gt;</code>
<code>isfinite</code>	Value: <code>&lt;ufunc 'isfinite'&gt;</code>
<code>isinf</code>	Value: <code>&lt;ufunc 'isinf'&gt;</code>
<code>isnan</code>	Value: <code>&lt;ufunc 'isnan'&gt;</code>
<code>ldexp</code>	Value: <code>&lt;ufunc 'ldexp'&gt;</code>
<code>left_shift</code>	Value: <code>&lt;ufunc 'left_shift'&gt;</code>
<code>less</code>	Value: <code>&lt;ufunc 'less'&gt;</code>
<code>less_equal</code>	Value: <code>&lt;ufunc 'less_equal'&gt;</code>
<code>little_endian</code>	Value: <code>True</code>
<code>log1p</code>	Value: <code>&lt;ufunc 'log1p'&gt;</code>
<code>logical_and</code>	Value: <code>&lt;ufunc 'logical_and'&gt;</code>
<code>logical_not</code>	Value: <code>&lt;ufunc 'logical_not'&gt;</code>
<code>logical_or</code>	Value: <code>&lt;ufunc 'logical_or'&gt;</code>
<code>logical_xor</code>	Value: <code>&lt;ufunc 'logical_xor'&gt;</code>
<code>maximum</code>	Value: <code>&lt;ufunc 'maximum'&gt;</code>
<code>mgrid</code>	Value: <code>&lt;numpy.lib.index.tricks.nd_grid object at 0x9ae7fec&gt;</code>
<code>minimum</code>	Value: <code>&lt;ufunc 'minimum'&gt;</code>
<code>mod</code>	Value: <code>&lt;ufunc 'remainder'&gt;</code>
<code>modf</code>	Value: <code>&lt;ufunc 'modf'&gt;</code>
<code>multiply</code>	Value: <code>&lt;ufunc 'multiply'&gt;</code>
<code>nan</code>	Value: <code>nan</code>
<code>nbytes</code>	Value: <code>{&lt;type 'numpy.int64'&gt;: 8, &lt;type 'numpy.int16'&gt;: 2, &lt;type ...</code>
<code>negative</code>	Value: <code>&lt;ufunc 'negative'&gt;</code>
<code>newaxis</code>	Value: <code>None</code>
<code>not_equal</code>	Value: <code>&lt;ufunc 'not_equal'&gt;</code>
<code>ogrid</code>	Value: <code>&lt;numpy.lib.index.tricks.nd_grid object at 0x9aef02c&gt;</code>
<code>ones_like</code>	Value: <code>&lt;ufunc 'ones_like'&gt;</code>
<code>oo</code>	Value: <code>oo</code>
<code>pi</code>	Value: <code>pi</code>
<code>pkgload</code>	Value: <code>&lt;numpy._import_tools.PackageLoader instance at 0x9c52b8c&gt;</code>
<code>r_</code>	Value: <code>&lt;numpy.lib.index.tricks.RClass object at 0x9aef0ac&gt;</code>
<code>radians</code>	Value: <code>&lt;ufunc 'radians'&gt;</code>
<code>reciprocal</code>	Value: <code>&lt;ufunc 'reciprocal'&gt;</code>
<code>remainder</code>	Value: <code>&lt;ufunc 'remainder'&gt;</code>

*continued on next page*

Name	Description
right_shift	Value: <ufunc 'right_shift'>
rint	Value: <ufunc 'rint'>
s_	Value: <numpy.lib.index_tricks.IndexExpression object at 0x9aef2cc>
sctypeDict	Value: {0: <type 'numpy.bool_'>, 1: <type 'numpy.int8'>, 2: <typ...
sctypeNA	Value: {'?': 'Bool', 'B': 'UInt8', 'Bool': <type 'numpy.bool_'>, ...
sctypes	Value: {'complex': [<type 'numpy.complex64'>, <type 'numpy.compl...
sieve	Value: <Sieve with 6 primes sieved: 2, 3, 5, ... 11, 13>
signbit	Value: <ufunc 'signbit'>
square	Value: <ufunc 'square'>
subtract	Value: <ufunc 'subtract'>
true_divide	Value: <ufunc 'true_divide'>
typeDict	Value: {0: <type 'numpy.bool_'>, 1: <type 'numpy.int8'>, 2: <typ...
typeNA	Value: {'?': 'Bool', 'B': 'UInt8', 'Bool': <type 'numpy.bool_'>, ...
typecodes	Value: {'All': '?bhilqpBHILQPfdgFDGSUV0', 'AllFloat': 'fdgFDG', ...
zoo	Value: zoo

## 1.3 Class Compiler

### 1.3.1 Methods

```
__init__(self, model)
```

```
build_substitution_list(self, for_matlab=False, for_c=True)
```

```
equations_to_function(self, eqs)
```

```
get_gaps_static_python(self)
```

```
tabify_expression(self, eq, for_matlab=False, for_c=True)
```

```
write_gaps_ccode_static(self)
```

### 1.3.2 Class Variables

Name	Description
model	Value: None

## 2 Module daredare

### 2.1 Functions

#### **rand(...)**

Return an array of the given dimensions which is initialized to random numbers from a uniform distribution in the range [0,1).

`rand(d0, d1, ..., dn) -> random values`

Note: This is a convenience function. If you want an interface that takes a tuple as the first argument use `numpy.random.random_sample(shape_tuple)`.

#### **randn(...)**

Returns zero-mean, unit-variance Gaussian random numbers in an array of shape (d0, d1, ..., dn).

`randn(d0, d1, ..., dn) -> random values`

Note: This is a convenience function. If you want an interface that takes a tuple as the first argument use `numpy.random.standard_normal(shape_tuple)`.

### 2.2 Variables

Name	Description
ALLOW_THREADS	<b>Value:</b> 1
BUFSIZE	<b>Value:</b> 10000
C	<b>Value:</b> <sympy.core.basic.ClassesRegistry instance at 0x96e56cc>
CLIP	<b>Value:</b> 0
Catalan	<b>Value:</b> Catalan
E	<b>Value:</b> E
ERR_CALL	<b>Value:</b> 3
ERR_DEFAULT	<b>Value:</b> 0
ERR_DEFAULT2	<b>Value:</b> 2084
ERR_IGNORE	<b>Value:</b> 0
ERR_LOG	<b>Value:</b> 5
ERR_PRINT	<b>Value:</b> 4
ERR_RAISE	<b>Value:</b> 2
ERR_WARN	<b>Value:</b> 1
EulerGamma	<b>Value:</b> EulerGamma
FLOATING_POINT_SUPPORT	<b>Value:</b> 1
FPE_DIVIDE_BY_ZERO	<b>Value:</b> 1

*continued on next page*

Name	Description
FPE_INVALID	Value: 8
FPE_OVERFLOW	Value: 2
FPE_UNDERFLOW	Value: 4
False_	Value: False
GoldenRatio	Value: GoldenRatio
I	Value: I
Inf	Value: inf
Infinity	Value: inf
MAXDIMS	Value: 32
NAN	Value: nan
NINF	Value: -inf
NZERO	Value: -0.0
NaN	Value: nan
PINF	Value: inf
PZERO	Value: 0.0
RAISE	Value: 2
S	Value: S
SHIFT_DIVIDEBYZERO	Value: 0
SHIFT_INVALID	Value: 9
SHIFT_OVERFLOW	Value: 3
SHIFT_UNDERFLOW	Value: 6
ScalarType	Value: (<type 'int'>, <type 'float'>, <type 'complex'>, <type 'l...
True_	Value: True
UFUNC_BUFSIZE_DEFAULT	Value: 10000
UFUNC_PYVALS_NAME	Value: 'UFUNC_PYVALS'
WRAP	Value: 1
absolute	Value: <ufunc 'absolute'>
arccosh	Value: <ufunc 'arccosh'>
arcsinh	Value: <ufunc 'arcsinh'>
arctan	Value: <ufunc 'arctan'>
arctan2	Value: <ufunc 'arctan2'>
bitwise_and	Value: <ufunc 'bitwise_and'>
bitwise_not	Value: <ufunc 'invert'>
bitwise_or	Value: <ufunc 'bitwise_or'>
bitwise_xor	Value: <ufunc 'bitwise_xor'>
c_	Value: <numpy.lib.index.tricks.CClass object at 0x9aef10c>
cast	Value: {<type 'numpy.int64'>: <function <lambda> at 0x99c5374>, ...
ceil	Value: <ufunc 'ceil'>
conj	Value: <ufunc 'conjugate'>
degrees	Value: <ufunc 'degrees'>
divide	Value: <ufunc 'divide'>
e	Value: 2.71828182846
equal	Value: <ufunc 'equal'>
expm1	Value: <ufunc 'expm1'>
fabs	Value: <ufunc 'fabs'>
floor_divide	Value: <ufunc 'floor_divide'>
fmod	Value: <ufunc 'fmod'>

*continued on next page*

Name	Description
<code>frexp</code>	Value: <ufunc 'frexp'>
<code>greater</code>	Value: <ufunc 'greater'>
<code>greater_equal</code>	Value: <ufunc 'greater_equal'>
<code>hypot</code>	Value: <ufunc 'hypot'>
<code>index_exp</code>	Value: <numpy.lib.index.tricks.IndexExpression object at 0x9aef28c>
<code>inf</code>	Value: inf
<code>infty</code>	Value: inf
<code>invert</code>	Value: <ufunc 'invert'>
<code>isfinite</code>	Value: <ufunc 'isfinite'>
<code>isinf</code>	Value: <ufunc 'isinf'>
<code>isnan</code>	Value: <ufunc 'isnan'>
<code>ldexp</code>	Value: <ufunc 'ldexp'>
<code>left_shift</code>	Value: <ufunc 'left_shift'>
<code>less</code>	Value: <ufunc 'less'>
<code>less_equal</code>	Value: <ufunc 'less_equal'>
<code>little_endian</code>	Value: True
<code>log1p</code>	Value: <ufunc 'log1p'>
<code>logical_and</code>	Value: <ufunc 'logical_and'>
<code>logical_not</code>	Value: <ufunc 'logical_not'>
<code>logical_or</code>	Value: <ufunc 'logical_or'>
<code>logical_xor</code>	Value: <ufunc 'logical_xor'>
<code>maximum</code>	Value: <ufunc 'maximum'>
<code>mgrid</code>	Value: <numpy.lib.index.tricks.nd_grid object at 0x9ae7fec>
<code>minimum</code>	Value: <ufunc 'minimum'>
<code>mod</code>	Value: <ufunc 'remainder'>
<code>modf</code>	Value: <ufunc 'modf'>
<code>multiply</code>	Value: <ufunc 'multiply'>
<code>nan</code>	Value: nan
<code>nbytes</code>	Value: {<type 'numpy.int64'>: 8, <type 'numpy.int16'>: 2, <type ...
<code>negative</code>	Value: <ufunc 'negative'>
<code>newaxis</code>	Value: None
<code>not_equal</code>	Value: <ufunc 'not_equal'>
<code>ogrid</code>	Value: <numpy.lib.index.tricks.nd_grid object at 0x9aef02c>
<code>ones_like</code>	Value: <ufunc 'ones_like'>
<code>oo</code>	Value: oo
<code>pi</code>	Value: pi
<code>pkgload</code>	Value: <numpy._import.tools.PackageLoader instance at 0x9c52b8c>
<code>r_</code>	Value: <numpy.lib.index.tricks.RClass object at 0x9aef0ac>
<code>radians</code>	Value: <ufunc 'radians'>
<code>reciprocal</code>	Value: <ufunc 'reciprocal'>
<code>remainder</code>	Value: <ufunc 'remainder'>
<code>right_shift</code>	Value: <ufunc 'right_shift'>
<code>rint</code>	Value: <ufunc 'rint'>

*continued on next page*



Name	Description
s_	<b>Value:</b> <numpy.lib.index_tricks.IndexExpression object at 0x9aef2cc>
sctypeDict	<b>Value:</b> {0: <type 'numpy.bool_'>, 1: <type 'numpy.int8'>, 2: <typ...
sctypeNA	<b>Value:</b> {'?': 'Bool', 'B': 'UInt8', 'Bool': <type 'numpy.bool_'>, ...
sctypes	<b>Value:</b> {'complex': [<type 'numpy.complex64'>, <type 'numpy.compl...
sieve	<b>Value:</b> <Sieve with 6 primes sieved: 2, 3, 5, ... 11, 13>
signbit	<b>Value:</b> <ufunc 'signbit'>
square	<b>Value:</b> <ufunc 'square'>
subtract	<b>Value:</b> <ufunc 'subtract'>
true_divide	<b>Value:</b> <ufunc 'true_divide'>
typeDict	<b>Value:</b> {0: <type 'numpy.bool_'>, 1: <type 'numpy.int8'>, 2: <typ...
typeNA	<b>Value:</b> {'?': 'Bool', 'B': 'UInt8', 'Bool': <type 'numpy.bool_'>, ...
typecodes	<b>Value:</b> {'All': '?bhilqpBHILQPfdgFDGSUV0', 'AllFloat': 'fdgFDG', ...
zoo	<b>Value:</b> zoo

### 3 Module misc.calculus

Version: 0.6.0

#### 3.1 Functions

**construct\_4\_blocks\_matrix**(*blocks*)

construct block matrix line by line input : `[[A1,A2],[A3,A4]]`

**rand**(...)

Return an array of the given dimensions which is initialized to random numbers from a uniform distribution in the range `[0,1)`.

`rand(d0, d1, ..., dn) -> random values`

Note: This is a convenience function. If you want an interface that takes a tuple as the first argument use `numpy.random.random_sample(shape_tuple)`.

**randn**(...)

Returns zero-mean, unit-variance Gaussian random numbers in an array of shape `(d0, d1, ..., dn)`.

`randn(d0, d1, ..., dn) -> random values`

Note: This is a convenience function. If you want an interface that takes a tuple as the first argument use `numpy.random.standard_normal(shape_tuple)`.

**second\_order\_taylorisation**(*expr, shocks, covariances*)

#### 3.2 Variables

Name	Description
ALLOW_THREADS	Value: 1
BUFSIZE	Value: 10000
C	Value: <sympy.core.basic.ClassesRegistry instance at 0x96e56cc>
CLIP	Value: 0
Catalan	Value: Catalan
E	Value: E
ERR_CALL	Value: 3
ERR_DEFAULT	Value: 0
ERR_DEFAULT2	Value: 2084

*continued on next page*

Name	Description
ERR_IGNORE	Value: 0
ERR_LOG	Value: 5
ERR_PRINT	Value: 4
ERR_RAISE	Value: 2
ERR_WARN	Value: 1
EulerGamma	Value: EulerGamma
FLOATING_POINT_SUPPORT	Value: 1
FPE_DIVIDEBYZERO	Value: 1
FPE_INVALID	Value: 8
FPE_OVERFLOW	Value: 2
FPE_UNDERFLOW	Value: 4
False_	Value: False
GoldenRatio	Value: GoldenRatio
I	Value: I
Inf	Value: inf
Infinity	Value: inf
MAXDIMS	Value: 32
NAN	Value: nan
NINF	Value: -inf
NZERO	Value: -0.0
NaN	Value: nan
PINF	Value: inf
PZERO	Value: 0.0
RAISE	Value: 2
S	Value: S
SHIFT_DIVIDEBYZERO	Value: 0
SHIFT_INVALID	Value: 9
SHIFT_OVERFLOW	Value: 3
SHIFT_UNDERFLOW	Value: 6
ScalarType	Value: (<type 'int'>, <type 'float'>, <type 'complex'>, <type 'l...
True_	Value: True
UFUNC_BUFSIZE_DEFAULT	Value: 10000
UFUNC_PYVALS_NAME	Value: 'UFUNC_PYVALS'
WRAP	Value: 1
absolute	Value: <ufunc 'absolute'>
add	Value: <ufunc 'add'>
arccosh	Value: <ufunc 'arccosh'>
arcsinh	Value: <ufunc 'arcsinh'>
arctan	Value: <ufunc 'arctan'>
arctan2	Value: <ufunc 'arctan2'>
bitwise_and	Value: <ufunc 'bitwise_and'>
bitwise_not	Value: <ufunc 'invert'>
bitwise_or	Value: <ufunc 'bitwise_or'>
bitwise_xor	Value: <ufunc 'bitwise_xor'>
c_	Value: <numpy.lib.index.tricks.CClass object at 0x9aef10c>
cast	Value: {<type 'numpy.int64'>: <function <lambda> at 0x99c5374>, ...

*continued on next page*

Name	Description
ceil	Value: <ufunc 'ceil'>
conj	Value: <ufunc 'conjugate'>
conjugate	Value: <ufunc 'conjugate'>
cos	Value: <ufunc 'cos'>
cosh	Value: <ufunc 'cosh'>
degrees	Value: <ufunc 'degrees'>
divide	Value: <ufunc 'divide'>
e	Value: 2.71828182846
equal	Value: <ufunc 'equal'>
exp	Value: <ufunc 'exp'>
expm1	Value: <ufunc 'expm1'>
fabs	Value: <ufunc 'fabs'>
floor	Value: <ufunc 'floor'>
floor_divide	Value: <ufunc 'floor_divide'>
fmod	Value: <ufunc 'fmod'>
frexp	Value: <ufunc 'frexp'>
greater	Value: <ufunc 'greater'>
greater_equal	Value: <ufunc 'greater_equal'>
hypot	Value: <ufunc 'hypot'>
index_exp	Value: <numpy.lib.index.tricks.IndexExpression object at 0x9aef28c>
inf	Value: inf
infty	Value: inf
invert	Value: <ufunc 'invert'>
isfinite	Value: <ufunc 'isfinite'>
isinf	Value: <ufunc 'isinf'>
isnan	Value: <ufunc 'isnan'>
ldexp	Value: <ufunc 'ldexp'>
left_shift	Value: <ufunc 'left_shift'>
less	Value: <ufunc 'less'>
less_equal	Value: <ufunc 'less_equal'>
little_endian	Value: True
log1p	Value: <ufunc 'log1p'>
logical_and	Value: <ufunc 'logical.and'>
logical_not	Value: <ufunc 'logical.not'>
logical_or	Value: <ufunc 'logical.or'>
logical_xor	Value: <ufunc 'logical.xor'>
maximum	Value: <ufunc 'maximum'>
mgrid	Value: <numpy.lib.index.tricks.nd_grid object at 0x9ae7fec>
minimum	Value: <ufunc 'minimum'>
mod	Value: <ufunc 'remainder'>
modf	Value: <ufunc 'modf'>
multiply	Value: <ufunc 'multiply'>
nan	Value: nan
nbytes	Value: {<type 'numpy.int64'>: 8, <type 'numpy.int16'>: 2, <type ...
negative	Value: <ufunc 'negative'>
newaxis	Value: None
not_equal	Value: <ufunc 'not_equal'>

*continued on next page*

Name	Description
ogrid	Value: <numpy.lib.index_tricks.nd_grid object at 0x9aef02c>
ones_like	Value: <ufunc 'ones_like'>
oo	Value: oo
pi	Value: 3.14159265359
pkgload	Value: <numpy._import_tools.PackageLoader instance at 0x9c52b8c>
r_	Value: <numpy.lib.index_tricks.RClass object at 0x9aef0ac>
radians	Value: <ufunc 'radians'>
reciprocal	Value: <ufunc 'reciprocal'>
remainder	Value: <ufunc 'remainder'>
right_shift	Value: <ufunc 'right_shift'>
rint	Value: <ufunc 'rint'>
s_	Value: <numpy.lib.index_tricks.IndexExpression object at 0x9aef2cc>
sctypeDict	Value: {0: <type 'numpy.bool_'>, 1: <type 'numpy.int8'>, 2: <typ...
sctypeNA	Value: {'?': 'Bool', 'B': 'UInt8', 'Bool': <type 'numpy.bool_'>, ...
sctypes	Value: {'complex': [<type 'numpy.complex64'>, <type 'numpy.compl...
sieve	Value: <Sieve with 6 primes sieved: 2, 3, 5, ... 11, 13>
sign	Value: <ufunc 'sign'>
signbit	Value: <ufunc 'signbit'>
sin	Value: <ufunc 'sin'>
sinh	Value: <ufunc 'sinh'>
square	Value: <ufunc 'square'>
subtract	Value: <ufunc 'subtract'>
tan	Value: <ufunc 'tan'>
tanh	Value: <ufunc 'tanh'>
true_divide	Value: <ufunc 'true_divide'>
typeDict	Value: {0: <type 'numpy.bool_'>, 1: <type 'numpy.int8'>, 2: <typ...
typeNA	Value: {'?': 'Bool', 'B': 'UInt8', 'Bool': <type 'numpy.bool_'>, ...
typecodes	Value: {'All': '?bhilqpBHILQPfdgFDGSUV0', 'AllFloat': 'fdgFDG', ...
zoo	Value: zoo

## 4 Module model.model

### 4.1 Variables

Name	Description
C	<b>Value:</b> <sympy.core.basic.ClassesRegistry instance at 0x96e56cc>
Catalan	<b>Value:</b> Catalan
E	<b>Value:</b> E
EulerGamma	<b>Value:</b> EulerGamma
GoldenRatio	<b>Value:</b> GoldenRatio
I	<b>Value:</b> I
S	<b>Value:</b> S
nan	<b>Value:</b> nan
oo	<b>Value:</b> oo
pi	<b>Value:</b> pi
sieve	<b>Value:</b> <Sieve with 6 primes sieved: 2, 3, 5, ... 11, 13>
zoo	<b>Value:</b> zoo

### 4.2 Class Model

#### 4.2.1 Methods

```
__init__(self, fname, lookup=False)
```

```
check_all(self, print_info=False, print_eq_info=False)
```

```
copy(self)
```

```
current_equations(self)
```

```
future_variables(self)
```

```
get_jacobian(self)
```

returns the jacobian of the equations with respect to  
all\_variables.f,all\_variables,all\_variables\_b where all\_variables == variables + exovariables

```
is_matlab_compatible(self)
```

```
is_uhlig_compatible(self)
```

```
map_function_to_expression(self, f, expr)
```

```
process_portfolio_model(self)
```

```
set_info(self, eq)
```

```
to_uhlig_model(self)
```

#### 4.2.2 Class Variables

Name	Description
<code>commands</code>	<b>Value:</b> <code>None</code>
<code>covariances</code>	<b>Value:</b> <code>[]</code>
<code>equations</code>	<b>Value:</b> <code>[]</code>
<code>exovariables</code>	<b>Value:</b> <code>[]</code>
<code>fname</code>	<b>Value:</b> <code>None</code>
<code>init_values</code>	<b>Value:</b> <code>{}</code>
<code>parameters</code>	<b>Value:</b> <code>[]</code>
<code>shocks</code>	<b>Value:</b> <code>[]</code>
<code>variables</code>	<b>Value:</b> <code>[]</code>
<code>variables_order</code>	<b>Value:</b> <code>{}</code>

## Index

- compiler (*package*)
  - compiler.compiler (*module*), 2–5
  - compiler.compiler.Compiler (*class*), 5
- daredare (*module*), 6–9
  - daredare.rand (*function*), 2, 6, 10
- misc (*package*)
  - misc.calculus (*module*), 10–13
    - misc.calculus.construct\_4\_blocks\_matrix (*function*), 10
    - misc.calculus.randn (*function*), 2, 6, 10
    - misc.calculus.second\_order\_taylorisation (*function*), 10
- model (*package*)
  - model.model (*module*), 14–15
    - model.model.Model (*class*), 14–15