Class LinearAlgebra

```
java.lang.Object
  └LinearAlgebra
```

```
public class LinearAlgebra
extends java.lang.Object
```

A library of linear algebra algorithms originally created in Python by Massimo Di Pierro and ported to Java. All code released under BSD licensing.

Version:
0.1
Author:
Ruthann Sudman
See Also:
Code Repository

Field Summary

private static double	**ap**
private static int	**ns**
private static int	**p**
private TestMatrix	**portfolio**
private double	**portfolio_return**
private double	**portfolio_risk**
private static double	**rp**

Constructor Summary

LinearAlgebra()

Method Summary

TestMatrix	**Cholesky**(TestMatrix A) Returns a TestMatrix object with the Cholesky algorithm applied.
TestMatrix	**exp**(TestMatrix x) Returns the exponent of a TestMatrix object.
TestMatrix	**getMarkovitzPortfolio**() Get method to return Markovitz portfolio value.
double	**getMarkovitzPortfolioReturn**() Get method to return Markovitz portfolio return.
double	**getMarkovitzPortfolioRisk**() Get method to return Markovitz portfolio risk.
boolean	**is_almost_symmetric**(TestMatrix x) Returns a boolean value indicating whether the matrix is almost symmetric.
boolean	**is_almost_zero**(TestMatrix A) Returns a boolean value indicating if a matrix is almost zero.
boolean	**is_positive_definite**(TestMatrix A) Returns a boolean value indicating if a TestMatrix is positive definite.
LinearAlgebra	**Markovitz**(TestMatrix mu, TestMatrix A, double r_free) Calculates the Markovitz portfolio, risk and return.
double	**norm**(double A) Returns the norm of a double value.
double	**norm**(TestMatrix A) Returns the norm of a TestMatrix.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

ap

```
private static double ap
```

---

rp

```
private static double rp
```

---

ns

```
private static int ns
```

---

p

```
private static int p
```

---

portfolio

```
private TestMatrix portfolio
```

---

portfolio_return

```
private double portfolio_return
```

---

portfolio_risk

```
private double portfolio_risk
```

Constructor Detail

LinearAlgebra

```
public LinearAlgebra()
```

Method Detail

is_almost_symmetric

```
public boolean is_almost_symmetric(TestMatrix x)
```

Returns a boolean value indicating whether the matrix is almost symmetric.

Parameters:

`x` - The TestMatrix object to be examined.

Returns:
The boolean result of the test.
Exception(s):
No known exceptions.
See Also:
TestMatrix

---

is_almost_zero

```
public boolean is_almost_zero(TestMatrix A)
```

Returns a boolean value indicating if a matrix is almost zero.

Parameters:
`A` - The TestMatrix object to be examined.
Returns:
Boolean result of the test.
Exception(s):
No known exceptions.
See Also:
TestMatrix

---

norm

```
public double norm(double A)
```

Returns the norm of a double value.

Parameters:
`A` - The value to be examined.
Returns:
The norm of A.
Exception(s):
No known sinces

---

norm

```
public double norm(TestMatrix A)
```

Returns the norm of a TestMatrix. Needs work. Not properly implemented.

Parameters:
`A` - The TestMatrix object to be examied.
Returns:
The norm of the matrix.

Exception(s):
Norm will always be zero. Not properly implemented.

---

exp

```
public TestMatrix exp(TestMatrix x)
```

Returns the exponent of a TestMatrix object.

Parameters:
`x` - The TestMatrix object to apply the function to.
Returns:
The exponent TestMatrix.
Exception(s):
Algorithm may fail to converge, division by zero errors.
See Also:
`TestMatrix`

Cholesky

```
public TestMatrix Cholesky(TestMatrix A)
```

Returns a TestMatrix object with the Cholesky algorithm applied.

Parameters:
`A` - The TestMatrix object to apply Cholesky to.
Returns:
A TestMatrix with Cholesky applied.
Exception(s):
Can't take a square root of a negative number.
See Also:
`TestMatrix`

---

is_positive_definite

```
public boolean is_positive_definite(TestMatrix A)
```

Returns a boolean value indicating if a TestMatrix is positive definite.

Parameters:
`A` - The TestMatrix to test for positive definite.
Returns:
The boolean result of the algorithm.
Exception(s):
Run time error possible.

See Also:
TestMatrix

Markovitz

```
public LinearAlgebra Markovitz(TestMatrix mu,
                               TestMatrix A,
                               double r_free)
```

Calculates the Markovitz portfolio, risk and return. Returns a reference to LinearAlgebra from which the Markovitz portfolio TestMatrix, risk and return can be obtained with get methods.

Parameters:
`mu` - Markovitz mu.
`A` - The TestMatrix object.
`r_free` - The risk free rate.
Returns:
LinearAlgebra reference to get portfolio, risk and return
Exception(s):
TestMatrix should be symmetric. Rows in mu should mirror columns in A
See Also:
getMarkovitzPortfolio(), getMarkovitzPortfolioRisk(), getMarkovitzPortfolioReturn()

---

getMarkovitzPortfolio

```
public TestMatrix getMarkovitzPortfolio()
```

Get method to return Markovitz portfolio value.

Returns:
Portfolio TestMatrix object
Exception(s):
Markovitz must be run and the value set prior to using this algorithm.
See Also:
Markovitz(TestMatrix, TestMatrix, double), TestMatrix

---

getMarkovitzPortfolioRisk

```
public double getMarkovitzPortfolioRisk()
```

Get method to return Markovitz portfolio risk.

Returns:
Markovitz portfolio risk.

Exception(s):

Markovitz must be run and the value set prior to using this algorithm.

See Also:

`Markovitz(TestMatrix, TestMatrix, double)`

---

getMarkovitzPortfolioReturn

```
public double getMarkovitzPortfolioReturn()
```

Get method to return Markovitz portfolio return.

Returns:

Markovitz portfolio return.

Exception(s):

Markovitz must be run and the value set prior to using this algorithm.

See Also:

`Markovitz(TestMatrix, TestMatrix, double)`
