

DotNetMatrix Namespace

Classes

Class	Description
CholeskyDecomposition	Cholesky Decomposition. For a symmetric, positive definite matrix A , the Cholesky decomposition is an lower triangular matrix L so that $A = L * L'$. If the matrix is not symmetric or positive definite, the constructor returns a partial decomposition and sets an internal flag that may be queried by the <code>isSPD()</code> method.
EigenvalueDecomposition	Eigenvalues and eigenvectors of a real matrix. If A is symmetric, then $A = V * D * V'$ where the eigenvalue matrix D is diagonal and the eigenvector matrix V is orthogonal. I.e. $A = V * Multiply(D * Multiply(V * Transpose()))$ and $V * Multiply(V * Transpose())$ equals the identity matrix. If A is not symmetric, then the eigenvalue matrix D is block diagonal with the real eigenvalues in 1-by-1 blocks and any complex eigenvalues, $\lambda + i * \mu$, in 2-by-2 blocks, $[\lambda, \mu; -\mu, \lambda]$. The columns of V represent the eigenvectors in the sense that $A * V = V * D$, i.e. $A * Multiply(V)$ equals $V * Multiply(D)$. The matrix V may be badly conditioned, or even singular, so the validity of the equation $A = V * D * Inverse(V)$ depends upon $V.cond()$.

GeneralMatrix	
LUDecomposition	
Maths	
QRDecomposition	QR Decomposition. For an m-by-n matrix A with $m \geq n$, the QR decomposition is an m-by-n orthogonal matrix Q and an n-by-n upper triangular matrix R so that $A = Q \cdot R$. The QR decomposition always exists, even if the matrix does not have full rank, so the constructor will never fail. The primary use of the QR decomposition is in the least squares solution of nonsquare systems of simultaneous linear equations. This will fail if <code>IsFullRank()</code> returns false.
SingularValueDecomposition	

CholeskyDecomposition Class

Cholesky Decomposition. For a symmetric, positive definite matrix A , the Cholesky decomposition is an lower triangular matrix L so that $A = L*L'$. If the matrix is not symmetric or positive definite, the constructor returns a partial decomposition and sets an internal flag that may be queried by the `isSPD()` method.

For a list of all members of this type, see [CholeskyDecomposition Members](#).

[System.Object](#) **CholeskyDecomposition**

```
[Visual Basic]
Public Class CholeskyDecomposition
Implements ISerializable
```

```
[C#]
public class CholeskyDecomposition :
    ISerializable
```

Requirements

Namespace: [DotNetMatrix](#)

Assembly: GeneralMatrix (in GeneralMatrix.dll)


See Also

[CholeskyDecomposition Members](#) | [DotNetMatrix Namespace](#)


CholeskyDecomposition Members

[CholeskyDecomposition overview](#)







Public Instance Constructors

 CholeskyDecomposition Constructor	Cholesky algorithm for symmetric and positive definite matrix.
---	--



Public Instance Properties


 SPD	Is the matrix symmetric and positive definite?
---	--

Public Instance Methods

 Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
 GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
 GetL	Return triangular factor.
 GetType (inherited from Object)	Gets the Type of the current instance.
 Solve	Solve $A * X = B$
 ToString (inherited from Object)	Returns a String that represents the current Object .

Private Instance Fields

 isspd	Symmetric and positive definite flag. @serial is symmetric and positive definite flag.
 L	Array for internal storage of

	decomposition. @serial internal array storage.
 n	Row and column dimension (square matrix). @serial matrix dimension.

Explicit Interface Implementations

 ISerializable.GetObjectData	
---	--

See Also

[CholeskyDecomposition Class](#) | [DotNetMatrix Namespace](#)

CholeskyDecomposition Constructor

Cholesky algorithm for symmetric and positive definite matrix.

```
[Visual Basic]Public Sub New( _  
    ByVal Arg As GeneralMatrix _  
)
```

```
[C#]  
public CholeskyDecomposition(  
    GeneralMatrix Arg  
)  
;
```

Parameters

Arg
Square, symmetric matrix.

Return Value

Structure to access L and isspd flag.




See Also

[CholeskyDecomposition Class](#) | [DotNetMatrix Namespace](#)

CholeskyDecomposition Fields

The fields of the **CholeskyDecomposition** class are listed below. For a complete list of **CholeskyDecomposition** class members, see the [CholeskyDecomposition Members](#) topic.

Private Instance Fields

 isspd	Symmetric and positive definite flag. @serial is symmetric and positive definite flag.
 L	Array for internal storage of decomposition. @serial internal array storage.
 n	Row and column dimension (square matrix). @serial matrix dimension.

See Also

[CholeskyDecomposition Class](#) | [DotNetMatrix Namespace](#)

CholeskyDecomposition.isspd Field

Symmetric and positive definite flag. @serial is symmetric and positive definite flag.

```
[Visual Basic]Private isspd As Boolean
```

```
[C#]  
private bool isspd;
```

See Also

[CholeskyDecomposition Class](#) | [DotNetMatrix Namespace](#)

CholeskyDecomposition.L Field

Array for internal storage of decomposition. @serial internal array storage.

```
[Visual Basic] Private L As Double()()
```

```
[C#]  
private double[][] L;
```

See Also

[CholeskyDecomposition Class](#) | [DotNetMatrix Namespace](#)

CholeskyDecomposition.n Field

Row and column dimension (square matrix). @serial matrix dimension.

```
[Visual Basic]Private n As Integer
```

```
[C#]  
private int n;
```


See Also

[CholeskyDecomposition Class](#) | [DotNetMatrix Namespace](#)

CholeskyDecomposition Properties

The properties of the **CholeskyDecomposition** class are listed below. For a complete list of **CholeskyDecomposition** class members, see the [CholeskyDecomposition Members](#) topic.

Public Instance Properties

 SPD	Is the matrix symmetric and positive definite?
---	--

See Also

[CholeskyDecomposition Class](#) | [DotNetMatrix Namespace](#)

CholeskyDecomposition.SPDP Property

Is the matrix symmetric and positive definite?

```
[Visual Basic]Overridable Public ReadOnly Prop
```

```
[C#]  
public virtual bool SPD {get;}
```

See Also

[CholeskyDecomposition Class](#) | [DotNetMatrix Namespace](#)

CholeskyDecomposition Methods

The methods of the **CholeskyDecomposition** class are listed below. For a complete list of **CholeskyDecomposition** class members, see the [CholeskyDecomposition Members](#) topic.

Public Instance Methods

Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
GetL	Return triangular factor.
GetType (inherited from Object)	Gets the Type of the current instance.
Solve	Solve $A * X = B$
ToString (inherited from Object)	Returns a String that represents the current Object .

Explicit Interface Implementations

ISerializable.GetObjectData	
---	--

See Also

[CholeskyDecomposition Class](#) | [DotNetMatrix Namespace](#)

CholeskyDecomposition.GetL Method

Return triangular factor.

```
[Visual Basic] Overridable Public Function GetL()
```

```
[C#]  
public virtual GeneralMatrix GetL();
```

Return Value

L

See Also

[CholeskyDecomposition Class](#) | [DotNetMatrix Namespace](#)

CholeskyDecomposition.Solve Method

Solve $A \cdot X = B$

```
[Visual Basic] Overridable Public Function Solve(  
    _  
    ByVal B As GeneralMatrix _  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix Solve(  
    GeneralMatrix B  
);
```

Parameters

B

A Matrix with as many rows as A and any number of columns.

Return Value

X so that $L \cdot L^T \cdot X = B$

Exceptions

Exception Type	Condition
ArgumentException	Matrix row dimensions must agree.
SystemException	Matrix is not symmetric positive definite.

See Also

[CholeskyDecomposition Class](#) | [DotNetMatrix Namespace](#)

CholeskyDecomposition.ISerializable.GetObjectData Metho

```
[Visual Basic]Sub GetObjectData( _  
    ByVal info As SerializationInfo, _  
    ByVal context As StreamingContext _  
) Implements _  
    ISerializable.GetObjectData
```

```
[C#]  
void ISerializable.GetObjectData(  
    SerializationInfo info,  
    StreamingContext context  
);
```

Implements

[ISerializable.GetObjectData](#)

See Also

[CholeskyDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition Class

Eigenvalues and eigenvectors of a real matrix. If A is symmetric, then $A = V \cdot D \cdot V'$ where the eigenvalue matrix D is diagonal and the eigenvector matrix V is orthogonal. I.e. $A = V \cdot \text{Multiply}(D \cdot \text{Multiply}(V \cdot \text{Transpose}()))$ and $V \cdot \text{Multiply}(V \cdot \text{Transpose}())$ equals the identity matrix. If A is not symmetric, then the eigenvalue matrix D is block diagonal with the real eigenvalues in 1-by-1 blocks and any complex eigenvalues, $\lambda + i \cdot \mu$, in 2-by-2 blocks, $[\lambda, \mu; -\mu, \lambda]$. The columns of V represent the eigenvectors in the sense that $A \cdot V = V \cdot D$, i.e. $A \cdot \text{Multiply}(V)$ equals $V \cdot \text{Multiply}(D)$. The matrix V may be badly conditioned, or even singular, so the validity of the equation $A = V \cdot D \cdot \text{Inverse}(V)$ depends upon $V \cdot \text{cond}()$.

For a list of all members of this type, see [EigenvalueDecomposition Members](#).

[System.Object](#) **EigenvalueDecomposition**

```
[Visual Basic]
Public Class EigenvalueDecomposition
Implements ISerializable
```

```
[C#]
public class EigenvalueDecomposition :
    ISerializable
```

Requirements

Namespace: [DotNetMatrix](#)

Assembly: GeneralMatrix (in GeneralMatrix.dll)


See Also

[EigenvalueDecomposition Members](#) | [DotNetMatrix Namespace](#)




EigenvalueDecomposition Members

[EigenvalueDecomposition overview](#)






Public Instance Constructors

 EigenvalueDecomposition Constructor	Check for symmetry, then construct the eigenvalue decomposition
---	---

Public Instance Properties









 D	Return the block diagonal eigenvalue matrix
 ImagEigenvalues	Return the imaginary parts of the eigenvalues
 RealEigenvalues	Return the real parts of the eigenvalues

Public Instance Methods






 Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
 GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
 GetType (inherited from Object)	Gets the Type of the current instance.
 GetV	Return the eigenvector matrix
 ToString (inherited from Object)	Returns a String that represents the current Object .

Private Instance Fields

 cdivi	
---	--

 cdivr	
 d	Arrays for internal storage of eigenvalues. @serial internal storage of eigenvalues.
 e	Arrays for internal storage of eigenvalues. @serial internal storage of eigenvalues.
 H	Array for internal storage of nonsymmetric Hessenberg form. @serial internal storage of nonsymmetric Hessenberg form.
 issymmetric	Symmetry flag. @serial internal symmetry flag.
 n	Row and column dimension (square matrix). @serial matrix dimension.
 ort	Working storage for nonsymmetric algorithm. @serial working storage for nonsymmetric algorithm.
 V	Array for internal storage of eigenvectors. @serial internal storage of eigenvectors.

Private Instance Methods

 cdiv	
 hqr2	
 orthes	
 tql2	
 tred2	

Explicit Interface Implementations

 [Serializable.GetObjectData](#)

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition Constructor

Check for symmetry, then construct the eigenvalue decomposition

```
[Visual Basic]Public Sub New( _  
    ByVal Arg As GeneralMatrix _  
)
```

```
[C#]  
public EigenvalueDecomposition(  
    GeneralMatrix Arg  
)  
;
```

Parameters

Arg
Square matrix

Return Value

Structure to access D and V.










See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition Fields

The fields of the **EigenvalueDecomposition** class are listed below. For a complete list of **EigenvalueDecomposition** class members, see the [EigenvalueDecomposition Members](#) topic.

Private Instance Fields

 cdivi	
 cdivr	
 d	Arrays for internal storage of eigenvalues. @serial internal storage of eigenvalues.
 e	Arrays for internal storage of eigenvalues. @serial internal storage of eigenvalues.
 H	Array for internal storage of nonsymmetric Hessenberg form. @serial internal storage of nonsymmetric Hessenberg form.
 issymmetric	Symmetry flag. @serial internal symmetry flag.
 n	Row and column dimension (square matrix). @serial matrix dimension.
 ort	Working storage for nonsymmetric algorithm. @serial working storage for nonsymmetric algorithm.
 V	Array for internal storage of eigenvectors. @serial internal storage of eigenvectors.

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.cdivi Field

```
[Visual Basic]Private cdivi As Double
```

```
[C#]  
private double cdivi;
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.cdivr Field

```
[Visual Basic]Private cdivr As Double
```

```
[C#]  
private double cdivr;
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.D Property

Return the block diagonal eigenvalue matrix

```
[Visual Basic] Overridable Public ReadOnly Prop
```

```
[C#]  
public virtual GeneralMatrix D {get;}
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.e Field

Arrays for internal storage of eigenvalues. @serial internal storage of eigenvalues.

```
[Visual Basic]Private e As Double\(\)
```

```
[C#]  
private double\[\] e;
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.H Field

Array for internal storage of nonsymmetric Hessenberg form.
@serial internal storage of nonsymmetric Hessenberg form.

```
[Visual Basic] Private H As Double()
```

```
[C#]  
private double[][] H;
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.issymmetric Field

Symmetry flag. @serial internal symmetry flag.

```
[Visual Basic]Private issymmetric As Boolean
```

```
[C#]  
private bool issymmetric;
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.n Field

Row and column dimension (square matrix). @serial matrix dimension.

```
[Visual Basic]Private n As Integer
```

```
[C#]  
private int n;
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.ort Field

Working storage for nonsymmetric algorithm. @serial working storage for nonsymmetric algorithm.

```
[Visual Basic]Private ort As Double\(\)
```

```
[C#]  
private double\[\] ort;
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.V Field

Array for internal storage of eigenvectors. @serial internal storage of eigenvectors.

```
[Visual Basic]Private V As Double\(\)[]
```

```
[C#]  
private double[][] V;
```




See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition Properties

The properties of the **EigenvalueDecomposition** class are listed below. For a complete list of **EigenvalueDecomposition** class members, see the [EigenvalueDecomposition Members](#) topic.

Public Instance Properties

 D	Return the block diagonal eigenvalue matrix
 ImagEigenvalues	Return the imaginary parts of the eigenvalues
 RealEigenvalues	Return the real parts of the eigenvalues

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.ImagEigenvalues Property

Return the imaginary parts of the eigenvalues

```
[Visual Basic] Overridable Public ReadOnly Property
```

```
[C#]  
public virtual double[] ImagEigenvalues {get;
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.RealEigenvalues Property

Return the real parts of the eigenvalues

```
[Visual Basic] Overridable Public ReadOnly Property
```

```
[C#]  
public virtual double[] RealEigenvalues {get;
```






See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)






EigenvalueDecomposition Methods

The methods of the **EigenvalueDecomposition** class are listed below. For a complete list of **EigenvalueDecomposition** class members, see the [EigenvalueDecomposition Members](#) topic.

Public Instance Methods

 Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
 GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
 GetType (inherited from Object)	Gets the Type of the current instance.
 GetV	Return the eigenvector matrix
 ToString (inherited from Object)	Returns a String that represents the current Object .

Private Instance Methods

 cdiv	
 hqr2	
 orthes	
 tql2	
 tred2	

Explicit Interface Implementations

 ISerializable.GetObjectData	
---	--

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.cdiv Method

```
[Visual Basic]Private Sub cdiv( _  
    ByVal xr As Double, _  
    ByVal xi As Double, _  
    ByVal yr As Double, _  
    ByVal yi As Double _  
)
```

```
[C#]  
private void cdiv(  
    double xr,  
    double xi,  
    double yr,  
    double yi  
);
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.GetV Method

Return the eigenvector matrix

```
[Visual Basic] Overridable Public Function GetV()
```

```
[C#]  
public virtual GeneralMatrix GetV();
```

Return Value

V

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.hqr2 Method

```
[Visual Basic]Private Sub hqr2()
```

```
[C#]  
private void hqr2();
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.orthes Method

```
[Visual Basic]Private Sub orthes()
```

```
[C#]  
private void orthes();
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.ISerializable.GetObjectData Meth

```
[Visual Basic]Sub GetObjectData( _  
    ByVal info As SerializationInfo, _  
    ByVal context As StreamingContext _  
) Implements _  
    ISerializable.GetObjectData
```

```
[C#]  
void ISerializable.GetObjectData(  
    SerializationInfo info,  
    StreamingContext context  
);
```

Implements

[ISerializable.GetObjectData](#)

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.tql2 Method

```
[Visual Basic]Private Sub tql2()
```

```
[C#]  
private void tql2();
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

EigenvalueDecomposition.tred2 Method

```
[Visual Basic]Private Sub tred2()
```

```
[C#]  
private void tred2();
```

See Also

[EigenvalueDecomposition Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix Class

For a list of all members of this type, see [GeneralMatrix Members](#).

[System.Object](#) **GeneralMatrix**

```
[Visual Basic]
Public Class GeneralMatrix
Implements ICloneable, ISerializable,
    IDisposable
```

```
[C#]
public class GeneralMatrix : ICloneable,
    ISerializable, IDisposable
```

Requirements

Namespace: [DotNetMatrix](#)

Assembly: GeneralMatrix (in GeneralMatrix.dll)




See Also

[GeneralMatrix Members](#) | [DotNetMatrix Namespace](#)




GeneralMatrix Members

[GeneralMatrix overview](#)


Public Static (Shared) Methods

 Create	Construct a matrix from a copy of a 2-D array.
 Identity	Generate identity matrix
 Random	Generate matrix with random elements







Public Static (Shared) Operators

 Addition Operator	Addition of matrices
 Multiplication Operator	Multiplication of matrices
 Subtraction Operator	Subtraction of matrices

Public Instance Constructors

 GeneralMatrix	Overloaded. Initializes a new instance of the GeneralMatrix class.
---	--

Public Instance Properties





 Array	Access the internal two-dimensional array.
 ArrayCopy	Copy the internal two-dimensional array.
 ColumnDimension	Get column dimension.
 ColumnPackedCopy	Make a one-dimensional column packed copy of the internal array.
 RowDimension	Get row dimension.
 RowPackedCopy	Make a one-dimensional row

packed copy of the internal array.




Public Instance Methods

◆Add	$C = A + B$
◆AddEquals	$A = A + B$
◆ArrayLeftDivide	Element-by-element left division, $C = A.\backslash B$
◆ArrayLeftDivideEquals	Element-by-element left division in place, $A = A.\backslash B$
◆ArrayMultiply	Element-by-element multiplication, $C = A.*B$
◆ArrayMultiplyEquals	Element-by-element multiplication in place, $A = A.*B$
◆ArrayRightDivide	Element-by-element right division, $C = A./B$
◆ArrayRightDivideEquals	Element-by-element right division in place, $A = A./B$
◆chol	Cholesky Decomposition
◆Clone	Clone the GeneralMatrix object.
◆Condition	Matrix condition (2 norm)
◆Copy	Make a deep copy of a matrix
◆Determinant	GeneralMatrix determinant
◆Dispose	Overloaded. Do not make this method virtual. A derived class should not be able to override this method.
◆Eigen	Eigenvalue Decomposition
◆Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .



◆ GetElement	Get a single element.
◆ GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
◆ GetMatrix	Overloaded. Get a submatrix.
◆ GetType (inherited from Object)	Gets the Type of the current instance.
◆ Inverse	Matrix inverse or pseudoinverse
◆ LUD	LU Decomposition
◆ Multiply	Overloaded. Linear algebraic matrix multiplication, $A * B$
◆ MultiplyEquals	Multiply a matrix by a scalar in place, $A = s*A$
◆ Norm1	One norm
◆ Norm2	Two norm
◆ NormF	Frobenius norm
◆ NormInf	Infinity norm
◆ QRD	QR Decomposition
◆ Rank	GeneralMatrix rank
◆ SetElement	Set a single element.
◆ SetMatrix	Overloaded. Set a submatrix.
◆ Solve	Solve $A*X = B$
◆ SolveTranspose	Solve $X*A = B$, which is also $A'*X' = B'$
◆ Subtract	$C = A - B$
◆ SubtractEquals	$A = A - B$
◆ SVD	Singular Value Decomposition

 ToString (inherited from Object)	Returns a String that represents the current Object .
 Trace	Matrix trace.
 Transpose	Matrix transpose.
 UnaryMinus	Unary minus

Private Instance Fields

 A	Array for internal storage of elements. @serial internal array storage.
 m	Row and column dimensions. @serial row dimension. @serial column dimension.
 n	Row and column dimensions. @serial row dimension. @serial column dimension.

Private Instance Methods

 CheckMatrixDimensions	Check if $\text{size}(A) == \text{size}(B) *$
 Dispose	Overloaded. <code>Dispose(bool disposing)</code> executes in two distinct scenarios. If disposing equals true, the method has been called directly or indirectly by a user's code. Managed and unmanaged resources can be disposed. If disposing equals false, the method has been called by the runtime from inside the finalizer and you should not reference other objects. Only unmanaged resources can be disposed.

Explicit Interface Implementations

 [ISerializable.GetObjectData](#)

A method called when serializing this class

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix Constructor

Construct an m-by-n matrix of zeros.

Overload List

Construct a matrix from a one-dimensional packed array

[`public GeneralMatrix\(double\[\],int\);`](#)

Construct a matrix from a 2-D array.

[`public GeneralMatrix\(double\[\]\[\]\);`](#)

Construct a matrix quickly without checking arguments.

[`public GeneralMatrix\(double\[\]\[\],int,int\);`](#)

Construct an m-by-n matrix of zeros.

[`public GeneralMatrix\(int,int\);`](#)

Construct an m-by-n constant matrix.

[`public GeneralMatrix\(int,int,double\);`](#)

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix Constructor (Int32, Int32)

Construct an m-by-n matrix of zeros.

```
[Visual Basic]Overloads Public Sub New( _  
    ByVal m As Integer, _  
    ByVal n As Integer _  
)
```

```
[C#]  
public GeneralMatrix(  
    int m,  
    int n  
);
```

Parameters

m
Number of rows.

n
Number of columns.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix Constructor Overload List](#)

GeneralMatrix Constructor (Int32, Int32, Double)

Construct an m-by-n constant matrix.

```
[Visual Basic]Overloads Public Sub New( _  
    ByVal m As Integer, _  
    ByVal n As Integer, _  
    ByVal s As Double _  
)
```

```
[C#]  
public GeneralMatrix(  
    int m,  
    int n,  
    double s  
) ;
```

Parameters

- m*
Number of rows.
- n*
Number of columns.
- s*
Fill the matrix with this scalar value.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix Constructor Overload List](#)

GeneralMatrix Constructor (Double[][])

Construct a matrix from a 2-D array.

```
[Visual Basic]Overloads Public Sub New( _  
    ByVal A As Double()[] _  
)
```

```
[C#]  
public GeneralMatrix(  
    double[][] A  
)  
;
```

Parameters

A
Two-dimensional array of doubles.

Exceptions

Exception Type	Condition
ArgumentException	All rows must have the same length

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix Constructor Overload List](#) | [Create](#)

GeneralMatrix Constructor (Double[[]], Int32, Int32)

Construct a matrix quickly without checking arguments.

```
[Visual Basic]Overloads Public Sub New( _  
    ByVal A As Double()[], _  
    ByVal m As Integer, _  
    ByVal n As Integer _  
)
```

```
[C#]  
public GeneralMatrix(  
    double[][] A,  
    int m,  
    int n  
) ;
```

Parameters

- A*
Two-dimensional array of doubles.
- m*
Number of rows.
- n*
Number of columns.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix Constructor Overload List](#)

GeneralMatrix Constructor (Double[], Int32)

Construct a matrix from a one-dimensional packed array

```
[Visual Basic]Overloads Public Sub New( _  
    ByVal vals As Double(), _  
    ByVal m As Integer _  
)
```

```
[C#]  
public GeneralMatrix(  
    double[] vals,  
    int m  
);
```

Parameters

vals

One-dimensional array of doubles, packed by columns (ala Fortran).

m

Number of rows.

Exceptions

Exception Type	Condition
ArgumentException	Array length must be a multiple of m.




See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix Constructor Overload List](#)

GeneralMatrix Fields

The fields of the **GeneralMatrix** class are listed below. For a complete list of **GeneralMatrix** class members, see the [GeneralMatrix Members](#) topic.

Private Instance Fields

 A	Array for internal storage of elements. @serial internal array storage.
 m	Row and column dimensions. @serial row dimension. @serial column dimension.
 n	Row and column dimensions. @serial row dimension. @serial column dimension.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.A Field

Array for internal storage of elements. @serial internal array storage.

```
[Visual Basic]Private A As Double\(\)[]
```

```
[C#]  
private double[][] A;
```

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.m Field

Row and column dimensions. @serial row dimension. @serial column dimension.

```
[Visual Basic]Private m As Integer
```

```
[C#]  
private int m;
```

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.n Field

Row and column dimensions. @serial row dimension. @serial column dimension.

```
[Visual Basic]Private n As Integer
```

```
[C#]  
private int n;
```







See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix Properties

The properties of the **GeneralMatrix** class are listed below. For a complete list of **GeneralMatrix** class members, see the [GeneralMatrix Members](#) topic.

Public Instance Properties

 Array	Access the internal two-dimensional array.
 ArrayCopy	Copy the internal two-dimensional array.
 ColumnDimension	Get column dimension.
 ColumnPackedCopy	Make a one-dimensional column packed copy of the internal array.
 RowDimension	Get row dimension.
 RowPackedCopy	Make a one-dimensional row packed copy of the internal array.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Array Property

Access the internal two-dimensional array.

```
[Visual Basic] Overridable Public ReadOnly Property  
Array
```

```
[C#]  
public virtual double\[\]\[\] Array {get;}
```

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.ArrayCopy Property

Copy the internal two-dimensional array.

```
[Visual Basic]Overridable Public ReadOnly Property  
ArrayCopy As Double[,]
```

```
[C#]  
public virtual double[,] ArrayCopy {get;}
```

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.ColumnDimension Property

Get column dimension.

```
[Visual Basic]Overridable Public ReadOnly Property
```

```
[C#]  
public virtual int ColumnDimension {get;}
```

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.ColumnPackedCopy Property

Make a one-dimensional column packed copy of the internal array.

```
[Visual Basic] Overridable Public ReadOnly Property
```

```
[C#]  
public virtual double\[\] ColumnPackedCopy {get
```

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.RowDimension Property

Get row dimension.

```
[Visual Basic]Overridable Public ReadOnly Property
```

```
[C#]  
public virtual int RowDimension {get;}
```

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.RowPackedCopy Property

Make a one-dimensional row packed copy of the internal array.

```
[Visual Basic] Overridable Public ReadOnly Property
```

```
[C#]  
public virtual double\[\] RowPackedCopy {get;}
```




See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)












GeneralMatrix Methods

The methods of the **GeneralMatrix** class are listed below. For a complete list of **GeneralMatrix** class members, see the [GeneralMatrix Members](#) topic.

Public Static (Shared) Methods

 Create	Construct a matrix from a copy of a 2-D array.
 Identity	Generate identity matrix
 Random	Generate matrix with random elements

Public Instance Methods

 Add	$C = A + B$
 AddEquals	$A = A + B$
 ArrayLeftDivide	Element-by-element left division, $C = A.\backslash B$
 ArrayLeftDivideEquals	Element-by-element left division in place, $A = A.\backslash B$
 ArrayMultiply	Element-by-element multiplication, $C = A.*B$
 ArrayMultiplyEquals	Element-by-element multiplication in place, $A = A.*B$
 ArrayRightDivide	Element-by-element right division, $C = A./B$
 ArrayRightDivideEquals	Element-by-element right division in place, $A = A./B$
 chol	Cholesky Decomposition
 Clone	Clone the GeneralMatrix object.
 Condition	Matrix condition (2 norm)

◆ Copy	Make a deep copy of a matrix
◆ Determinant	GeneralMatrix determinant
◆ Dispose	Overloaded. Do not make this method virtual. A derived class should not be able to override this method.
◆ Eigen	Eigenvalue Decomposition
◆ Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
◆ GetElement	Get a single element.
◆ GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
◆ GetMatrix	Overloaded. Get a submatrix.
◆ GetType (inherited from Object)	Gets the Type of the current instance.
◆ Inverse	Matrix inverse or pseudoinverse
◆ LUD	LU Decomposition
◆ Multiply	Overloaded. Linear algebraic matrix multiplication, $A * B$
◆ MultiplyEquals	Multiply a matrix by a scalar in place, $A = s*A$
◆ Norm1	One norm
◆ Norm2	Two norm
◆ NormF	Frobenius norm
◆ NormInf	Infinity norm
◆ QRD	QR Decomposition
◆ Rank	GeneralMatrix rank

SetElement	Set a single element.
SetMatrix	Overloaded. Set a submatrix.
Solve	Solve $A * X = B$
SolveTranspose	Solve $X * A = B$, which is also $A' * X' = B'$
Subtract	$C = A - B$
SubtractEquals	$A = A - B$
SVD	Singular Value Decomposition
ToString (inherited from Object)	Returns a String that represents the current Object .
Trace	Matrix trace.
Transpose	Matrix transpose.
UnaryMinus	Unary minus

Private Instance Methods

CheckMatrixDimensions	Check if $\text{size}(A) == \text{size}(B) *$
Dispose	Overloaded. <code>Dispose(bool disposing)</code> executes in two distinct scenarios. If disposing equals true, the method has been called directly or indirectly by a user's code. Managed and unmanaged resources can be disposed. If disposing equals false, the method has been called by the runtime from inside the finalizer and you should not reference other objects. Only unmanaged resources can be disposed.

Explicit Interface Implementations

--	--

 [Serializable.GetObjectData](#)

A method called when serializing this class

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Add Method

$C = A + B$

```
[Visual Basic]Overridable Public Function Add(  
    _  
    ByVal B As GeneralMatrix _  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix Add(  
    GeneralMatrix B  
);
```

Parameters

B
another matrix

Return Value

$A + B$

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.AddEquals Method

$A = A + B$

```
[Visual Basic] Overridable Public Function Add(  
    _  
    ByVal B As GeneralMatrix _  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix AddEquals(  
    GeneralMatrix B  
);
```

Parameters

B
another matrix

Return Value

$A + B$

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.ArrayLeftDivide Method

Element-by-element left division, $C = A \setminus B$

```
[Visual Basic] Overridable Public Function ArrayLeftDivide(  
    _  
    ByVal B As GeneralMatrix _  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix ArrayLeftDivide(  
    GeneralMatrix B  
);
```

Parameters

B
another matrix

Return Value

$A \setminus B$

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.ArrayLeftDivideEquals Method

Element-by-element left division in place, $A = A.\backslash B$

```
[Visual Basic] Overridable Public Function ArrayLeftDivideEquals(  
    _  
    ByVal B As GeneralMatrix _  
    ) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix ArrayLeftDivideEquals(  
    GeneralMatrix B  
    );
```

Parameters

B
another matrix

Return Value

$A.\backslash B$

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.ArrayMultiply Method

Element-by-element multiplication, $C = A * B$

```
[Visual Basic] Overridable Public Function ArrayMultiply(  
    _  
    ByVal B As GeneralMatrix _  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix ArrayMultiply(  
    GeneralMatrix B  
);
```

Parameters

B
another matrix

Return Value

$A * B$

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.ArrayMultiplyEquals Method

Element-by-element multiplication in place, $A = A * B$

```
[Visual Basic] Overridable Public Function ArrayMultiplyEquals(  
    _  
    ByVal B As GeneralMatrix _  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix ArrayMultiplyEquals(  
    GeneralMatrix B  
);
```

Parameters

B
another matrix

Return Value

$A * B$

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.ArrayRightDivide Method

Element-by-element right division, $C = A./B$

```
[Visual Basic] Overridable Public Function ArrayRightDivide(  
    A As GeneralMatrix,   
    ByVal B As GeneralMatrix  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix ArrayRightDivide(  
    GeneralMatrix A,  
    GeneralMatrix B  
);
```

Parameters

B
another matrix

Return Value

A./B

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.ArrayRightDivideEquals Method

Element-by-element right division in place, $A = A./B$

```
[Visual Basic] Overridable Public Function ArrayRightDivideEquals(  
    _  
    ByVal B As GeneralMatrix _  
    ) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix ArrayRightDivideEquals(  
    GeneralMatrix B  
    );
```

Parameters

B
another matrix

Return Value

A./B

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.CheckMatrixDimensions Method

Check if $\text{size}(A) == \text{size}(B) *$

```
[Visual Basic]Private Sub CheckMatrixDimensions(  
    _  
    ByVal B As GeneralMatrix _  
)
```

```
[C#]  
private void CheckMatrixDimensions(  
    GeneralMatrix B  
);
```

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.chol Method

Cholesky Decomposition

```
[Visual Basic] Overridable Public Function chol
```

```
[C#]  
public virtual CholeskyDecomposition chol();
```

Return Value

CholeskyDecomposition

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [CholeskyDecomposition](#)

GeneralMatrix.Clone Method

Clone the GeneralMatrix object.

```
[Visual Basic]NotOverridable Public Function  
Implements _  
    ICloneable.Clone
```

```
[C#]  
public object Clone();
```

Implements

[ICloneable.Clone](#)

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Condition Method

Matrix condition (2 norm)

```
[Visual Basic] Overridable Public Function Condition() As Double
```

```
[C#]  
public virtual double Condition();
```

Return Value

ratio of largest to smallest singular value.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Copy Method

Make a deep copy of a matrix

```
[Visual Basic] Overridable Public Function Copy()
```

```
[C#]  
public virtual GeneralMatrix Copy();
```

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Create Method

Construct a matrix from a copy of a 2-D array.

```
[Visual Basic]Public Shared Function Create(  
    _  
    ByVal A As Double()[] _  
) As GeneralMatrix
```

```
[C#]  
public static GeneralMatrix Create(  
    double[][] A  
);
```

Parameters

A
Two-dimensional array of doubles.

Exceptions

Exception Type	Condition
ArgumentException	All rows must have the same length

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Determinant Method

GeneralMatrix determinant

```
[Visual Basic]Overridable Public Function De:
```

```
[C#]  
public virtual double Determinant();
```

Return Value

determinant

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Dispose Method

Do not make this method virtual. A derived class should not be able to override this method.

Overload List

Do not make this method virtual. A derived class should not be able to override this method.

[public void Dispose\(\);](#)

Dispose(bool disposing) executes in two distinct scenarios. If disposing equals true, the method has been called directly or indirectly by a user's code. Managed and unmanaged resources can be disposed. If disposing equals false, the method has been called by the runtime from inside the finalizer and you should not reference other objects. Only unmanaged resources can be disposed.

[private void Dispose\(bool\);](#)

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Dispose Method ()

Do not make this method virtual. A derived class should not be able to override this method.

```
[Visual Basic]NotOverridable Overloads Public  
Implements _  
IDisposable.Dispose
```

```
[C#]  
public void Dispose();
```

Implements

[IDisposable.Dispose](#)

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix.Dispose Overload List](#)

GeneralMatrix.Dispose Method (Boolean)

Dispose(bool disposing) executes in two distinct scenarios. If disposing equals true, the method has been called directly or indirectly by a user's code. Managed and unmanaged resources can be disposed. If disposing equals false, the method has been called by the runtime from inside the finalizer and you should not reference other objects. Only unmanaged resources can be disposed.

```
[Visual Basic]Overloads Private Sub Dispose(  
    _  
    ByVal disposing As Boolean _  
)
```

```
[C#]  
private void Dispose(  
    bool disposing  
);
```

Parameters

disposing

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix.Dispose Overload List](#)

GeneralMatrix.Eigen Method

Eigenvalue Decomposition

```
[Visual Basic] Overridable Public Function Eigen()
```

```
[C#]  
public virtual EigenvalueDecomposition Eigen()
```

Return Value

EigenvalueDecomposition

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [EigenvalueDecomposition](#)

GeneralMatrix.GetElement Method

Get a single element.

```
[Visual Basic]Overridable Public Function GetElement(  
    ByVal i As Integer, _  
    ByVal j As Integer _  
) As Double
```

```
[C#]  
public virtual double GetElement(  
    int i,  
    int j  
);
```

Parameters

i
Row index.

j
Column index.

Return Value

A(i,j)

Exceptions

Exception Type	Condition
IndexOutOfRangeException	

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.GetMatrix Method

Get a submatrix.

Overload List

Get a submatrix.

[public virtual GeneralMatrix GetMatrix\(int,int,int,int\);](#)

Get a submatrix.

[public virtual GeneralMatrix GetMatrix\(int,int,int\[\]\);](#)

Get a submatrix.

[public virtual GeneralMatrix GetMatrix\(int\[\],int,int\);](#)

Get a submatrix.

[public virtual GeneralMatrix GetMatrix\(int\[\],int\[\]\);](#)

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.GetMatrix Method (Int32, Int32, Int32, Int32)

Get a submatrix.

```
[Visual Basic] Overridable Overloads Public Function  
-  
    ByVal i0 As Integer, -  
    ByVal i1 As Integer, -  
    ByVal j0 As Integer, -  
    ByVal j1 As Integer -  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix GetMatrix(  
    int i0,  
    int i1,  
    int j0,  
    int j1  
);
```

Parameters

- i0*
Initial row index
- i1*
Final row index
- j0*
Initial column index
- j1*
Final column index

Return Value

A(i0:i1,j0:j1)

Exceptions

Exception Type	Condition
----------------	-----------

[IndexOutOfRangeException](#)

Submatrix indices

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) |
[GeneralMatrix.GetMatrix Overload List](#)

GeneralMatrix.GetMatrix Method (Int32, Int32, Int32[])

Get a submatrix.

```
[Visual Basic] Overridable Overloads Public Function  
-  
    ByVal i0 As Integer, -  
    ByVal i1 As Integer, -  
    ByVal c As Integer() -  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix GetMatrix(  
    int i0,  
    int i1,  
    int[] c  
);
```

Parameters

- i0*
Initial row index
- i1*
Final row index
- c*
Array of column indices.

Return Value

A(i0:i1,c(:))

Exceptions

Exception Type	Condition
IndexOutOfRangeException	Submatrix indices

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) |

[GeneralMatrix.GetMatrix Overload List](#)

GeneralMatrix.GetMatrix Method (Int32[], Int32, Int32)

Get a submatrix.

```
[Visual Basic]Overridable Overloads Public Function  
-  
    ByVal r As Integer(), _  
    ByVal j0 As Integer, _  
    ByVal j1 As Integer _  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix GetMatrix(  
    int[] r,  
    int j0,  
    int j1  
);
```

Parameters

r
Array of row indices.

j0
Initial column index

j1
Final column index

Return Value

A(r(:),j0:j1)

Exceptions

Exception Type	Condition
IndexOutOfRangeException	Submatrix indices

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) |

[GeneralMatrix.GetMatrix Overload List](#)

GeneralMatrix.GetMatrix Method (Int32[], Int32[])

Get a submatrix.

```
[Visual Basic]Overridable Overloads Public Function  
-  
    ByVal r As Integer(), _  
    ByVal c As Integer() _  
    ) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix GetMatrix(  
    int[] r,  
    int[] c  
);
```

Parameters

- r*
Array of row indices.
- c*
Array of column indices.

Return Value

A(r(:),c(:))

Exceptions

Exception Type	Condition
IndexOutOfRangeException	Submatrix indices

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix.GetMatrix Overload List](#)

GeneralMatrix.Identity Method

Generate identity matrix

```
[Visual Basic]Public Shared Function Identity(  
    _  
    ByVal m As Integer, _  
    ByVal n As Integer _  
    ) As GeneralMatrix
```

```
[C#]  
public static GeneralMatrix Identity(  
    int m,  
    int n  
);
```

Parameters

m
Number of rows.

n
Number of columns.

Return Value

An m-by-n matrix with ones on the diagonal and zeros elsewhere.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Inverse Method

Matrix inverse or pseudoinverse

```
[Visual Basic] Overridable Public Function Inverse
```

```
[C#]  
public virtual GeneralMatrix Inverse();
```

Return Value

inverse(A) if A is square, pseudoinverse otherwise.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.LUD Method

LU Decomposition

```
[Visual Basic]Overridable Public Function LUD()
```

```
[C#]  
public virtual LUDecomposition LUD();
```

Return Value

LUDecomposition

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [LUDecomposition](#)

GeneralMatrix.Multiply Method

Linear algebraic matrix multiplication, $A * B$

Overload List

Linear algebraic matrix multiplication, $A * B$

[public virtual GeneralMatrix Multiply\(GeneralMatrix\);](#)

Multiply a matrix by a scalar, $C = s*A$

[public virtual GeneralMatrix Multiply\(double\);](#)

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Multiply Method (GeneralMatrix)

Linear algebraic matrix multiplication, $A * B$

```
[Visual Basic] Overridable Overloads Public Function  
    _  
    ByVal B As GeneralMatrix _  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix Multiply(  
    GeneralMatrix B  
);
```

Parameters

B

another matrix

Return Value

Matrix product, $A * B$

Exceptions

Exception Type	Condition
ArgumentException	Matrix inner dimensions must agree.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix.Multiply Overload List](#)

GeneralMatrix.Multiply Method (Double)

Multiply a matrix by a scalar, $C = s * A$

```
[Visual Basic] Overridable Overloads Public Function  
-  
    ByVal s As Double _  
    ) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix Multiply(  
    double s  
);
```

Parameters

s
scalar

Return Value

$s * A$

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) |
[GeneralMatrix.Multiply Overload List](#)

GeneralMatrix.MultiplyEquals Method

Multiply a matrix by a scalar in place, $A = s \cdot A$

```
[Visual Basic] Overridable Public Function MultiplyEquals(  
    ByVal s As Double  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix MultiplyEquals(  
    double s  
);
```

Parameters

s
scalar

Return Value

replace A by $s \cdot A$

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Norm1 Method

One norm

```
[Visual Basic] Overridable Public Function Norm1()
```

```
[C#]  
public virtual double Norm1();
```

Return Value

maximum column sum.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Norm2 Method

Two norm

```
[Visual Basic]Overridable Public Function Norm2()
```

```
[C#]  
public virtual double Norm2();
```

Return Value

maximum singular value.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.NormF Method

Frobenius norm

```
[Visual Basic] Overridable Public Function NormF()
```

```
[C#]  
public virtual double NormF();
```

Return Value

sqrt of sum of squares of all elements.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.NormInf Method

Infinity norm

```
[Visual Basic] Overridable Public Function NormInf()
```

```
[C#]  
public virtual double NormInf();
```

Return Value

maximum row sum.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.QRD Method

QR Decomposition

```
[Visual Basic] Overridable Public Function QRDecomposition()
```

```
[C#]  
public virtual QRDecomposition QRD();
```

Return Value

QRDecomposition

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [QRDecomposition](#)

GeneralMatrix.Random Method

Generate matrix with random elements

```
[Visual Basic]Public Shared Function Random(  
    _  
    ByVal m As Integer, _  
    ByVal n As Integer _  
    ) As GeneralMatrix
```

```
[C#]  
public static GeneralMatrix Random(  
    int m,  
    int n  
);
```

Parameters

m

Number of rows.

n

Number of columns.

Return Value

An m-by-n matrix with uniformly distributed random elements.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Rank Method

GeneralMatrix rank

```
[Visual Basic]Overridable Public Function Rank
```

```
[C#]  
public virtual int Rank();
```

Return Value

effective numerical rank, obtained from SVD.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.SetElement Method

Set a single element.

```
[Visual Basic]Overridable Public Sub SetElement(  
    _  
    ByVal i As Integer, _  
    ByVal j As Integer, _  
    ByVal s As Double _  
)
```

```
[C#]  
public virtual void SetElement(  
    int i,  
    int j,  
    double s  
);
```

Parameters

- i*
Row index.
- j*
Column index.
- s*
A(i,j).

Exceptions

Exception Type	Condition
IndexOutOfRangeException	

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.SetMatrix Method

Set a submatrix.

Overload List

Set a submatrix.

[public virtual void SetMatrix\(int,int,int,int,GeneralMatrix\);](#)

Set a submatrix.

[public virtual void SetMatrix\(int,int,int\[\],GeneralMatrix\);](#)

Set a submatrix.

[public virtual void SetMatrix\(int\[\],int,int,GeneralMatrix\);](#)

Set a submatrix.

[public virtual void SetMatrix\(int\[\],int\[\],GeneralMatrix\);](#)

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.SetMatrix Method (Int32, Int32, Int32, Int32, GeneralMatrix)

Set a submatrix.

```
[Visual Basic]Overridable Overloads Public Sub  
—  
    ByVal i0 As Integer, —  
    ByVal i1 As Integer, —  
    ByVal j0 As Integer, —  
    ByVal j1 As Integer, —  
    ByVal X As GeneralMatrix —  
)
```

```
[C#]  
public virtual void SetMatrix(  
    int i0,  
    int i1,  
    int j0,  
    int j1,  
    GeneralMatrix X  
);
```

Parameters

- i0*
Initial row index
- i1*
Final row index
- j0*
Initial column index
- j1*
Final column index
- X*
A(i0:i1,j0:j1)

Exceptions

Exception Type	Condition
IndexOutOfRangeException	Submatrix indices

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix.SetMatrix Overload List](#)

GeneralMatrix.SetMatrix Method (Int32, Int32, Int32[], GeneralMatrix)

Set a submatrix.

```
[Visual Basic]Overridable Overloads Public Sub  
-  
    ByVal i0 As Integer, _  
    ByVal i1 As Integer, _  
    ByVal c As Integer(), _  
    ByVal X As GeneralMatrix _  
)
```

```
[C#]  
public virtual void SetMatrix(  
    int i0,  
    int i1,  
    int[] c,  
    GeneralMatrix X  
);
```

Parameters

- i0*
Initial row index
- i1*
Final row index
- c*
Array of column indices.
- X*
A(i0:i1,c(:))

Exceptions

Exception Type	Condition
IndexOutOfRangeException	Submatrix indices

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix.SetMatrix Overload List](#)

GeneralMatrix.SetMatrix Method (Int32[], Int32, Int32, GeneralMatrix)

Set a submatrix.

```
[Visual Basic]Overridable Overloads Public Sub  
-  
    ByVal r As Integer(), _  
    ByVal j0 As Integer, _  
    ByVal j1 As Integer, _  
    ByVal X As GeneralMatrix _  
)
```

```
[C#]  
public virtual void SetMatrix(  
    int[] r,  
    int j0,  
    int j1,  
    GeneralMatrix X  
);
```

Parameters

- r*
Array of row indices.
- j0*
Initial column index
- j1*
Final column index
- X*
A(r(:),j0:j1)

Exceptions

Exception Type	Condition
IndexOutOfRangeException	Submatrix indices

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix.SetMatrix Overload List](#)

GeneralMatrix.SetMatrix Method (Int32[], Int32[], GeneralMatrix)

Set a submatrix.

```
[Visual Basic]Overridable Overloads Public Sub  
-  
    ByVal r As Integer(), _  
    ByVal c As Integer(), _  
    ByVal X As GeneralMatrix _  
)
```

```
[C#]  
public virtual void SetMatrix(  
    int[] r,  
    int[] c,  
    GeneralMatrix X  
);
```

Parameters

- r*
Array of row indices.
- c*
Array of column indices.
- X*
A(r(:),c(:))

Exceptions

Exception Type	Condition
IndexOutOfRangeException	Submatrix indices

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [GeneralMatrix.SetMatrix Overload List](#)

GeneralMatrix.Solve Method

Solve $A \cdot X = B$

```
[Visual Basic] Overridable Public Function Solve(  
    _  
    ByVal B As GeneralMatrix _  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix Solve(  
    GeneralMatrix B  
);
```

Parameters

B
right hand side

Return Value

solution if A is square, least squares solution otherwise

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.SolveTranspose Method

Solve $X \cdot A = B$, which is also $A' \cdot X' = B'$

```
[Visual Basic] Overridable Public Function SolveTranspose(  
    ByVal B As GeneralMatrix  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix SolveTranspose(  
    GeneralMatrix B  
);
```

Parameters

B
right hand side

Return Value

solution if A is square, least squares solution otherwise.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Subtract Method

$C = A - B$

```
[Visual Basic]Overridable Public Function Subtract(  
    _  
    ByVal B As GeneralMatrix _  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix Subtract(  
    GeneralMatrix B  
);
```

Parameters

B
another matrix

Return Value

A - B

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.SubtractEquals Method

$A = A - B$

```
[Visual Basic] Overridable Public Function SubtractEquals(  
    _  
    ByVal B As GeneralMatrix  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix SubtractEquals(  
    GeneralMatrix B  
);
```

Parameters

B
another matrix

Return Value

$A - B$

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.SVD Method

Singular Value Decomposition

```
[Visual Basic] Overridable Public Function SVD
```

```
[C#]  
public virtual SingularValueDecomposition SVD
```

Return Value

SingularValueDecomposition

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#) | [SingularValueDecomposition](#)

GeneralMatrix.ISerializable.GetObjectData Method

A method called when serializing this class

```
[Visual Basic]Sub GetObjectData( _  
    ByVal info As SerializationInfo, _  
    ByVal context As StreamingContext _  
) Implements _  
    ISerializable.GetObjectData
```

```
[C#]  
void ISerializable.GetObjectData(  
    SerializationInfo info,  
    StreamingContext context  
);
```

Parameters

info

context

Implements

[ISerializable.GetObjectData](#)

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Trace Method

Matrix trace.

```
[Visual Basic]Overridable Public Function Trace
```

```
[C#]  
public virtual double Trace();
```

Return Value

sum of the diagonal elements.

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.Transpose Method

Matrix transpose.

```
[Visual Basic] Overridable Public Function Transpose()
```

```
[C#]  
public virtual GeneralMatrix Transpose();
```

Return Value

A'

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix.UnaryMinus Method

Unary minus

```
[Visual Basic] Overridable Public Function UnaryMinus()
```

```
[C#]  
public virtual GeneralMatrix UnaryMinus();
```

Return Value

-A




See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix Operators

The operators of the **GeneralMatrix** class are listed below. For a complete list of **GeneralMatrix** class members, see the [GeneralMatrix Members](#) topic.

Public Static (Shared) Operators

 Addition Operator	Addition of matrices
 Multiplication Operator	Multiplication of matrices
 Subtraction Operator	Subtraction of matrices

See Also

[GeneralMatrix Class](#) | [GeneralMatrix Members](#) | [DotNetMatrix Namespace](#)

GeneralMatrix Addition Operator

Addition of matrices

```
[Visual Basic]returnValue =  
GeneralMatrix.op_Addition(m1, m2)
```

```
[C#]  
public static GeneralMatrix operator +(  
    GeneralMatrix m1,  
    GeneralMatrix m2  
);
```

Parameters

m1

m2

Return Value

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix Multiplication Operator

Multiplication of matrices

```
[Visual Basic]returnValue =  
GeneralMatrix.op_Multiply(m1, m2)
```

```
[C#]  
public static GeneralMatrix operator *(  
    GeneralMatrix m1,  
    GeneralMatrix m2  
);
```

Parameters

m1

m2

Return Value

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

GeneralMatrix Subtraction Operator

Subtraction of matrices

```
[Visual Basic]returnValue =  
GeneralMatrix.op_Subtraction(m1, m2)
```

```
[C#]  
public static GeneralMatrix operator -(  
    GeneralMatrix m1,  
    GeneralMatrix m2  
);
```

Parameters

m1

m2

Return Value

See Also

[GeneralMatrix Class](#) | [DotNetMatrix Namespace](#)

LUDecomposition Class

For a list of all members of this type, see [LUDecomposition Members](#).

[System.Object](#) **LUDecomposition**

```
[Visual Basic]  
Public Class LUDecomposition  
Implements ISerializable
```

```
[C#]  
public class LUDecomposition :  
ISerializable
```

Requirements

Namespace: [DotNetMatrix](#)

Assembly: GeneralMatrix (in GeneralMatrix.dll)

See Also

[LUDecomposition Members](#) | [DotNetMatrix Namespace](#)






LUDecomposition Members

[LUDecomposition overview](#)







Public Instance Constructors

 LUDecomposition Constructor	LU Decomposition
---	------------------

Public Instance Properties






 DoublePivot	Return pivot permutation vector as a one-dimensional double array
 IsNonSingular	Is the matrix nonsingular?
 L	Return lower triangular factor
 Pivot	Return pivot permutation vector
 U	Return upper triangular factor

Public Instance Methods

 Determinant	Determinant
 Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
 GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
 GetType (inherited from Object)	Gets the Type of the current instance.
 Solve	Solve $A * X = B$
 ToString (inherited from Object)	Returns a String that represents the current Object .

Private Instance Fields

--	--

 LU	Array for internal storage of decomposition. @serial internal array storage.
 m	Row and column dimensions, and pivot sign. @serial column dimension. @serial row dimension. @serial pivot sign.
 n	Row and column dimensions, and pivot sign. @serial column dimension. @serial row dimension. @serial pivot sign.
 piv	Internal storage of pivot vector. @serial pivot vector.
 pivsign	Row and column dimensions, and pivot sign. @serial column dimension. @serial row dimension. @serial pivot sign.

Explicit Interface Implementations

 IQueryable.GetObjectData	
--	--

See Also

[LUdecomposition Class](#) | [DotNetMatrix Namespace](#)

LU Decomposition Constructor

LU Decomposition

```
[Visual Basic]Public Sub New( _  
    ByVal A As GeneralMatrix _  
)
```

```
[C#]  
public LUdecomposition(  
    GeneralMatrix A  
)  
;
```

Parameters

A
Rectangular matrix

Return Value

Structure to access L, U and piv.






See Also

[LUdecomposition Class](#) | [DotNetMatrix Namespace](#)

LUdecomposition Fields

The fields of the **LUdecomposition** class are listed below. For a complete list of **LUdecomposition** class members, see the [LUdecomposition Members](#) topic.

Private Instance Fields

 LU	Array for internal storage of decomposition. @serial internal array storage.
 m	Row and column dimensions, and pivot sign. @serial column dimension. @serial row dimension. @serial pivot sign.
 n	Row and column dimensions, and pivot sign. @serial column dimension. @serial row dimension. @serial pivot sign.
 piv	Internal storage of pivot vector. @serial pivot vector.
 pivsign	Row and column dimensions, and pivot sign. @serial column dimension. @serial row dimension. @serial pivot sign.

See Also

[LUdecomposition Class](#) | [DotNetMatrix Namespace](#)

LUDecomposition.LU Field

Array for internal storage of decomposition. @serial internal array storage.

```
[Visual Basic]Private LU As Double()[ ]
```

```
[C#]  
private double[][] LU;
```

See Also

[LUDecomposition Class](#) | [DotNetMatrix Namespace](#)

LUDecomposition.m Field

Row and column dimensions, and pivot sign. @serial column dimension. @serial row dimension. @serial pivot sign.

```
[Visual Basic]Private m As Integer
```

```
[C#]  
private int m;
```

See Also

[LUDecomposition Class](#) | [DotNetMatrix Namespace](#)

LUDecomposition.n Field

Row and column dimensions, and pivot sign. @serial column dimension. @serial row dimension. @serial pivot sign.

```
[Visual Basic] Private n As Integer
```

```
[C#]  
private int n;
```

See Also

[LUDecomposition Class](#) | [DotNetMatrix Namespace](#)

LUdecomposition.piv Field

Internal storage of pivot vector. @serial pivot vector.

```
[Visual Basic]Private piv As Integer()
```

```
[C#]  
private int[] piv;
```

See Also

[LUdecomposition Class](#) | [DotNetMatrix Namespace](#)

LUDecomposition.pivsign Field

Row and column dimensions, and pivot sign. @serial column dimension. @serial row dimension. @serial pivot sign.

```
[Visual Basic]Private pivsign As Integer
```

```
[C#]  
private int pivsign;
```






See Also

[LUDecomposition Class](#) | [DotNetMatrix Namespace](#)

LUdecomposition Properties

The properties of the **LUdecomposition** class are listed below. For a complete list of **LUdecomposition** class members, see the [LUdecomposition Members](#) topic.

Public Instance Properties

 DoublePivot	Return pivot permutation vector as a one-dimensional double array
 IsNonSingular	Is the matrix nonsingular?
 L	Return lower triangular factor
 Pivot	Return pivot permutation vector
 U	Return upper triangular factor

See Also

[LUdecomposition Class](#) | [DotNetMatrix Namespace](#)

LUDecomposition.DoublePivot Property

Return pivot permutation vector as a one-dimensional double array

```
[Visual Basic] Overridable Public ReadOnly Property
```

```
[C#]  
public virtual double\[\] DoublePivot {get;}
```

See Also

[LUDecomposition Class](#) | [DotNetMatrix Namespace](#)

LUdecomposition.IsNonSingular Property

Is the matrix nonsingular?

```
[Visual Basic]Overridable Public ReadOnly Property
```

```
[C#]  
public virtual bool IsNonSingular {get;}
```

See Also

[LUdecomposition Class](#) | [DotNetMatrix Namespace](#)

LUdecomposition.L Property

Return lower triangular factor

```
[Visual Basic] Overridable Public ReadOnly Property
```

```
[C#]  
public virtual GeneralMatrix L {get;}
```

See Also

[LUdecomposition Class](#) | [DotNetMatrix Namespace](#)

LUdecomposition.Pivot Property

Return pivot permutation vector

```
[Visual Basic]Overridable Public ReadOnly Property
```

```
[C#]  
public virtual int\[\] Pivot {get;}
```

See Also

[LUdecomposition Class](#) | [DotNetMatrix Namespace](#)

LUdecomposition.U Property

Return upper triangular factor

```
[Visual Basic]Overridable Public ReadOnly Property
```

```
[C#]  
public virtual GeneralMatrix U {get;}
```







See Also

[LUdecomposition Class](#) | [DotNetMatrix Namespace](#)

LUdecomposition Methods

The methods of the **LUdecomposition** class are listed below. For a complete list of **LUdecomposition** class members, see the [LUdecomposition Members](#) topic.

Public Instance Methods

 Determinant	Determinant
 Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
 GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
 GetType (inherited from Object)	Gets the Type of the current instance.
 Solve	Solve $A * X = B$
 ToString (inherited from Object)	Returns a String that represents the current Object .

Explicit Interface Implementations

 ISerializable.GetObjectData	
---	--

See Also

[LUdecomposition Class](#) | [DotNetMatrix Namespace](#)

LUDecomposition.Determinant Method

Determinant

```
[Visual Basic]Overridable Public Function De
```

```
[C#]  
public virtual double Determinant();
```

Return Value

$\det(A)$

Exceptions

Exception Type	Condition
ArgumentException	Matrix must be square

See Also

[LUDecomposition Class](#) | [DotNetMatrix Namespace](#)

LUDecomposition.Solve Method

Solve $A \cdot X = B$

```
[Visual Basic] Overridable Public Function Solve(  
    _  
    ByVal B As GeneralMatrix _  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix Solve(  
    GeneralMatrix B  
);
```

Parameters

B

A Matrix with as many rows as A and any number of columns.

Return Value

X so that $L \cdot U \cdot X = B(\text{piv},:)$

Exceptions

Exception Type	Condition
ArgumentException	Matrix row dimensions must agree.
SystemException	Matrix is singular.

See Also

[LUDecomposition Class](#) | [DotNetMatrix Namespace](#)

LUDecomposition.ISerializable.GetObjectData Method

```
[Visual Basic]Sub GetObjectData( _  
    ByVal info As SerializationInfo, _  
    ByVal context As StreamingContext _  
) Implements _  
    ISerializable.GetObjectData
```

```
[C#]  
void ISerializable.GetObjectData(  
    SerializationInfo info,  
    StreamingContext context  
);
```

Implements

[ISerializable.GetObjectData](#)

See Also

[LUDecomposition Class](#) | [DotNetMatrix Namespace](#)

Maths Class

For a list of all members of this type, see [Maths Members](#).

[System.Object](#) **Maths**

```
[Visual Basic]  
Private Class Maths
```

```
[C#]  
private class Maths
```

Requirements

Namespace: [DotNetMatrix](#)

Assembly: GeneralMatrix (in GeneralMatrix.dll)

See Also

[Maths Members](#) | [DotNetMatrix Namespace](#)

Maths Members

[Maths overview](#)

Public Static (Shared) Methods

Hypot	$\text{sqrt}(a^2 + b^2)$ without under/overflow.
-----------------------	--

Public Instance Constructors

Maths Constructor	Initializes a new instance of the Maths class.
-----------------------------------	--

Public Instance Methods

Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
GetType (inherited from Object)	Gets the Type of the current instance.
ToString (inherited from Object)	Returns a String that represents the current Object .

See Also

[Maths Class](#) | [DotNetMatrix Namespace](#)

Maths Constructor

Initializes a new instance of the Maths class.

```
[Visual Basic]Public Sub New()
```

```
[C#]  
public Maths();
```


See Also

[Maths Class](#) | [DotNetMatrix Namespace](#)





Maths Methods

The methods of the **Maths** class are listed below. For a complete list of **Maths** class members, see the [Maths Members](#) topic.

Public Static (Shared) Methods

 Hypot	$\text{sqrt}(a^2 + b^2)$ without under/overflow.
---	--

Public Instance Methods

 Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
 GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
 GetType (inherited from Object)	Gets the Type of the current instance.
 ToString (inherited from Object)	Returns a String that represents the current Object .

See Also

[Maths Class](#) | [DotNetMatrix Namespace](#)

Maths.Hypot Method

$\sqrt{a^2 + b^2}$ without under/overflow.

```
[Visual Basic]Public Shared Function Hypot(  
    _  
    ByVal a As Double, _  
    ByVal b As Double _  
    ) As Double
```

```
[C#]  
public static double Hypot(  
    double a,  
    double b  
);
```

Parameters

a

b

Return Value

See Also

[Maths Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition Class

QR Decomposition. For an m-by-n matrix A with $m \geq n$, the QR decomposition is an m-by-n orthogonal matrix Q and an n-by-n upper triangular matrix R so that $A = Q \cdot R$. The QR decomposition always exists, even if the matrix does not have full rank, so the constructor will never fail. The primary use of the QR decomposition is in the least squares solution of nonsquare systems of simultaneous linear equations. This will fail if `IsFullRank()` returns false.

For a list of all members of this type, see [QRDecomposition Members](#).

[System.Object](#) **QRDecomposition**

```
[Visual Basic]
Public Class QRDecomposition
Implements ISerializable
```

```
[C#]
public class QRDecomposition :
    ISerializable
```

Requirements

Namespace: [DotNetMatrix](#)

Assembly: GeneralMatrix (in GeneralMatrix.dll)


See Also

[QRDecomposition Members](#) | [DotNetMatrix Namespace](#)





QRDecomposition Members

[QRDecomposition overview](#)






Public Instance Constructors

 QRDecomposition Constructor	QR Decomposition, computed by Householder reflections.
---	--

Public Instance Properties





 FullRank	Is the matrix full rank?
 H	Return the Householder vectors
 Q	Generate and return the (economy-sized) orthogonal factor
 R	Return the upper triangular factor

Public Instance Methods

 Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
 GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
 GetType (inherited from Object)	Gets the Type of the current instance.
 Solve	Least squares solution of $A \cdot X = B$
 ToString (inherited from Object)	Returns a String that represents the current Object .

Private Instance Fields

--	--

 m	Row and column dimensions. @serial column dimension. @serial row dimension.
 n	Row and column dimensions. @serial column dimension. @serial row dimension.
 QR	Array for internal storage of decomposition. @serial internal array storage.
 Rdiag	Array for internal storage of diagonal of R. @serial diagonal of R.

Explicit Interface Implementations

 ISerializable.GetObjectData	
---	--

See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition Constructor

QR Decomposition, computed by Householder reflections.

```
[Visual Basic]Public Sub New( _  
    ByVal A As GeneralMatrix _  
)
```

```
[C#]  
public QRDecomposition(  
    GeneralMatrix A  
);
```

Parameters

A
Rectangular matrix

Return Value

Structure to access R and the Householder vectors and compute Q.





See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition Fields

The fields of the **QRDecomposition** class are listed below. For a complete list of **QRDecomposition** class members, see the [QRDecomposition Members](#) topic.

Private Instance Fields

 m	Row and column dimensions. @serial column dimension. @serial row dimension.
 n	Row and column dimensions. @serial column dimension. @serial row dimension.
 QR	Array for internal storage of decomposition. @serial internal array storage.
 Rdiag	Array for internal storage of diagonal of R. @serial diagonal of R.

See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition.m Field

Row and column dimensions. @serial column dimension. @serial row dimension.

```
[Visual Basic]Private m As Integer
```

```
[C#]  
private int m;
```

See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition.n Field

Row and column dimensions. @serial column dimension. @serial row dimension.

```
[Visual Basic]Private n As Integer
```

```
[C#]  
private int n;
```

See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition.QR Field

Array for internal storage of decomposition. @serial internal array storage.

```
[Visual Basic]Private QR As Double()[ ]
```

```
[C#]  
private double[][] QR;
```

See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition.Rdiag Field

Array for internal storage of diagonal of R. @serial diagonal of R.

```
[Visual Basic]Private Rdiag As Double\(\)
```

```
[C#]  
private double\[\] Rdiag;
```





See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition Properties

The properties of the **QRDecomposition** class are listed below. For a complete list of **QRDecomposition** class members, see the [QRDecomposition Members](#) topic.

Public Instance Properties

 FullRank	Is the matrix full rank?
 H	Return the Householder vectors
 Q	Generate and return the (economy-sized) orthogonal factor
 R	Return the upper triangular factor

See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition.FullRank Property

Is the matrix full rank?

```
[Visual Basic]Overridable Public ReadOnly Prop
```

```
[C#]  
public virtual bool FullRank {get;}
```

See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition.H Property

Return the Householder vectors

```
[Visual Basic] Overridable Public ReadOnly Property
```

```
[C#]  
public virtual GeneralMatrix H {get;}
```

See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition.Q Property

Generate and return the (economy-sized) orthogonal factor

```
[Visual Basic] Overridable Public ReadOnly Prop
```

```
[C#]  
public virtual GeneralMatrix Q {get;}
```

See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition.R Property

Return the upper triangular factor

```
[Visual Basic] Overridable Public ReadOnly Prop
```

```
[C#]  
public virtual GeneralMatrix R {get;}
```






See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition Methods

The methods of the **QRDecomposition** class are listed below. For a complete list of **QRDecomposition** class members, see the [QRDecomposition Members](#) topic.

Public Instance Methods

 Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
 GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
 GetType (inherited from Object)	Gets the Type of the current instance.
 Solve	Least squares solution of $A \cdot X = B$
 ToString (inherited from Object)	Returns a String that represents the current Object .

Explicit Interface Implementations

 IQueryable.GetObjectData	
--	--

See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition.Solve Method

Least squares solution of $A \cdot X = B$

```
[Visual Basic]Overridable Public Function Solve(  
    _  
    ByVal B As GeneralMatrix _  
) As GeneralMatrix
```

```
[C#]  
public virtual GeneralMatrix Solve(  
    GeneralMatrix B  
);
```

Parameters

B

A Matrix with as many rows as A and any number of columns.

Return Value

X that minimizes the two norm of $Q \cdot R \cdot X - B$.

Exceptions

Exception Type	Condition
ArgumentException	Matrix row dimensions must agree.
SystemException	Matrix is rank deficient.

See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

QRDecomposition.ISerializable.GetObjectData Method

```
[Visual Basic]Sub GetObjectData( _  
    ByVal info As SerializationInfo, _  
    ByVal context As StreamingContext _  
) Implements _  
    ISerializable.GetObjectData
```

```
[C#]  
void ISerializable.GetObjectData(  
    SerializationInfo info,  
    StreamingContext context  
);
```

Implements

[ISerializable.GetObjectData](#)

See Also

[QRDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition Class

For a list of all members of this type, see [SingularValueDecomposition Members](#).

[System.Object](#) **SingularValueDecomposition**

```
[Visual Basic]  
Public Class SingularValueDecomposition  
Implements ISerializable
```

```
[C#]  
public class SingularValueDecomposition :  
    ISerializable
```

Requirements

Namespace: [DotNetMatrix](#)

Assembly: GeneralMatrix (in GeneralMatrix.dll)


See Also

[SingularValueDecomposition Members](#) | [DotNetMatrix Namespace](#)



SingularValueDecomposition Members

[SingularValueDecomposition overview](#)










Public Instance Constructors

 SingularValueDecomposition Constructor	Construct the singular value decomposition
--	--






Public Instance Properties

 S	Return the diagonal matrix of singular values
 SingularValues	Return the one-dimensional array of singular values

Public Instance Methods

 Condition	Two norm condition number
 Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
 GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
 GetType (inherited from Object)	Gets the Type of the current instance.
 GetU	Return the left singular vectors
 GetV	Return the right singular vectors
 Norm2	Two norm
 Rank	Effective numerical matrix rank
 ToString (inherited from Object)	Returns a String that represents the current Object .

Private Instance Fields

 m	Row and column dimensions. @serial row dimension. @serial column dimension.
 n	Row and column dimensions. @serial row dimension. @serial column dimension.
 s	Array for internal storage of singular values. @serial internal storage of singular values.
 U	Arrays for internal storage of U and V. @serial internal storage of U. @serial internal storage of V.
 V	Arrays for internal storage of U and V. @serial internal storage of U. @serial internal storage of V.

Explicit Interface Implementations

 ISerializable.GetObjectData	
---	--

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition Constructor

Construct the singular value decomposition

```
[Visual Basic]Public Sub New( _  
    ByVal Arg As GeneralMatrix _  
)
```

```
[C#]  
public SingularValueDecomposition(  
    GeneralMatrix Arg  
)  
;
```

Parameters

Arg
Rectangular matrix

Return Value

Structure to access U, S and V.






See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition Fields

The fields of the **SingularValueDecomposition** class are listed below. For a complete list of **SingularValueDecomposition** class members, see the [SingularValueDecomposition Members](#) topic.

Private Instance Fields

 m	Row and column dimensions. @serial row dimension. @serial column dimension.
 n	Row and column dimensions. @serial row dimension. @serial column dimension.
 s	Array for internal storage of singular values. @serial internal storage of singular values.
 U	Arrays for internal storage of U and V. @serial internal storage of U. @serial internal storage of V.
 V	Arrays for internal storage of U and V. @serial internal storage of U. @serial internal storage of V.

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition.m Field

Row and column dimensions. @serial row dimension. @serial column dimension.

```
[Visual Basic]Private m As Integer
```

```
[C#]  
private int m;
```

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition.n Field

Row and column dimensions. @serial row dimension. @serial column dimension.

```
[Visual Basic]Private n As Integer
```

```
[C#]  
private int n;
```

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition.S Property

Return the diagonal matrix of singular values

```
[Visual Basic] Overridable Public ReadOnly Property
```

```
[C#]  
public virtual GeneralMatrix S {get;}
```

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition.U Field

Arrays for internal storage of U and V. @serial internal storage of U.
@serial internal storage of V.

```
[Visual Basic]Private U As Double[][]
```

```
[C#]  
private double[][] U;
```

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition.V Field

Arrays for internal storage of U and V. @serial internal storage of U.
@serial internal storage of V.

```
[Visual Basic]Private V As Double()[ ]
```

```
[C#]  
private double[][] V;
```



See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition Properties

The properties of the **SingularValueDecomposition** class are listed below. For a complete list of **SingularValueDecomposition** class members, see the [SingularValueDecomposition Members](#) topic.

Public Instance Properties

 S	Return the diagonal matrix of singular values
 SingularValues	Return the one-dimensional array of singular values

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition.SingularValues Property

Return the one-dimensional array of singular values

```
[Visual Basic] Overridable Public ReadOnly Property
```

```
[C#]  
public virtual double[] SingularValues {get;}
```

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition Methods

The methods of the **SingularValueDecomposition** class are listed below. For a complete list of **SingularValueDecomposition** class members, see the [SingularValueDecomposition Members](#) topic.

Public Instance Methods

Condition	Two norm condition number
Equals (inherited from Object)	Determines whether the specified Object is equal to the current Object .
GetHashCode (inherited from Object)	Serves as a hash function for a particular type, suitable for use in hashing algorithms and data structures like a hash table.
GetType (inherited from Object)	Gets the Type of the current instance.
GetU	Return the left singular vectors
GetV	Return the right singular vectors
Norm2	Two norm
Rank	Effective numerical matrix rank
ToString (inherited from Object)	Returns a String that represents the current Object .

Explicit Interface Implementations

 ISerializable.GetObjectData	
---	--

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition.Condition Method

Two norm condition number

```
[Visual Basic]Overridable Public Function Condition()
```

```
[C#]  
public virtual double Condition();
```

Return Value

$\max(S)/\min(S)$

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition.GetU Method

Return the left singular vectors

```
[Visual Basic] Overridable Public Function GetU()
```

```
[C#]  
public virtual GeneralMatrix GetU();
```

Return Value

U

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition.GetV Method

Return the right singular vectors

```
[Visual Basic] Overridable Public Function GetV()
```

```
[C#]  
public virtual GeneralMatrix GetV();
```

Return Value

V

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition.Norm2 Method

Two norm

```
[Visual Basic] Overridable Public Function Norm2()
```

```
[C#]  
public virtual double Norm2();
```

Return Value

$\max(S)$

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition.Rank Method

Effective numerical matrix rank

```
[Visual Basic] Overridable Public Function Rank
```

```
[C#]  
public virtual int Rank();
```

Return Value

Number of nonnegligible singular values.

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)

SingularValueDecomposition.ISerializable.GetObjectData M

```
[Visual Basic]Sub GetObjectData( _  
    ByVal info As SerializationInfo, _  
    ByVal context As StreamingContext _  
) Implements _  
    ISerializable.GetObjectData
```

```
[C#]  
void ISerializable.GetObjectData(  
    SerializationInfo info,  
    StreamingContext context  
);
```

Implements

[ISerializable.GetObjectData](#)

See Also

[SingularValueDecomposition Class](#) | [DotNetMatrix Namespace](#)